Buku Ajar Logika

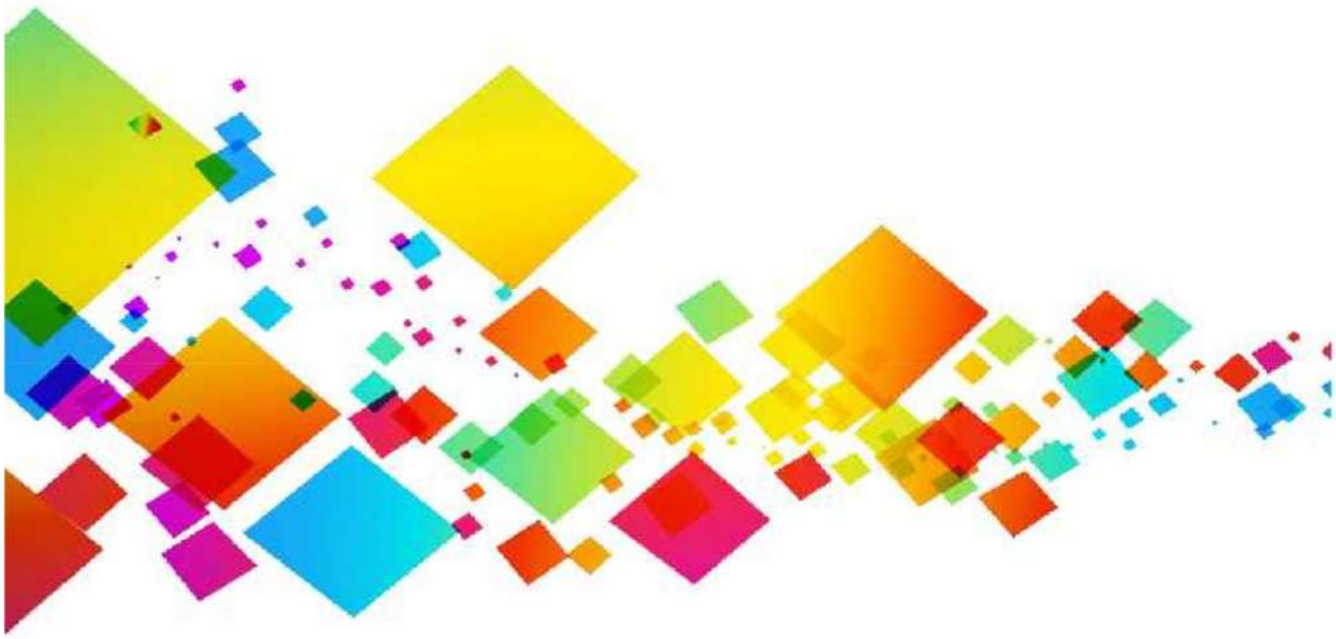
by Endra Rahmawati

Submission date: 07-Apr-2020 10:12AM (UTC+0700)
Submission ID: 1291601571
File name: C.6.g.3-Scan_Buku_Ajar_Logika.pdf (1.61M)
Word count: 32068
Character count: 193116

BUKU AJAR

IMPLEMENTASI DASAR LOGIKA & ALGORITMA

(Lengkap dengan Flowchart dan Pseudocode Menggunakan Notasi C/Java)



Pengarang:
Endra Rahmawati, M.Kom.

INSTITUT BISNIS DAN INFORMATIKA STIKOM SURABAYA

Heart & Mind Towards Excellence

Kata Pengantar

Puji Syukur kehadirat Tuhan Yang Maha Esa atas limpahan rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan pembuatan Buku Ajar Implementasi Dasar Logika & Algoritma dengan Notasi C/Java. **Buku Ajar ini disusun untuk membantu para mahasiswa menguasai konsep dan implementasi dasar logika algoritma dengan menggunakan flowchart dan pseudocode.**

Logika Algoritma merupakan mata kuliah dasar yang harus dikuasai oleh mahasiswa, terutama mahasiswa jurusan Sistem Informasi/Teknik Informatika. Mata kuliah ini biasanya diberikan di tahun pertama perkuliahan, yaitu di semester 1. Tidak jarang mahasiswa mengalami kesulitan dalam mempelajari dan menguasai konsep Logika Algoritma dengan baik dan benar.

Adapun materi-materi yang disampaikan pada **Buku Ajar ini terdiri dari 12 Bab**, yaitu sebagai berikut :

Bab 1 : Membahas tentang **Konsep Dasar Logika Algoritma**. Mahasiswa dikenalkan pada elemen-elemen algoritma dan metode pembuatan algoritma.

Bab 2 : Membahas tentang **Pengenalan Tipe Data**. Dengan mempelajari materi pada Bab 2 ini, diharapkan mahasiswa dapat mengetahui perbedaan penggunaan tipe data dengan mudah.

Bab 3 : Membahas tentang **Flowchart dan Pseudocode**. Mahasiswa akan diajarkan bagaimana membuat algoritma menggunakan 2 metode yang berbeda, yaitu menggunakan flowchart dan menggunakan pseudocode.

Bab 4 : Membahas tentang **Proses Sekuensial**. Proses Sekuensial merupakan proses paling sederhana pada sebuah algoritma. Proses ini berjalan secara urut, sesuai dengan tahapan yang ditentukan, tanpa melibatkan proses percabangan atau perulangan.

Bab 5 : Membahas tentang **Proses Percabangan (Selection/Decision)**. Proses ini merupakan kelanjutan dari Proses Sekuensial. Bab ini mengajarkan bagaimana mahasiswa berpikir secara kondisional.

Bab 6 : Membahas tentang **Proses Perulangan (Looping).** Proses ini merupakan kelanjutan dari Proses Sekunesial dan Percabangan. Bab ini mengajarkan bagaimana mahasiswa berpikir secara berulang. Penggunaan proses perulangan banyak dimanfaatkan untuk mencetak deret angka dan pengoperasian variabel array.

Bab 7 : Membahas tentang **Array 1D.** Array merupakan salah satu struktur data yang menggunakan konsep tabel. Array 1D biasanya dimanfaatkan untuk menyimpan nilai yang dengan konsep tabel satu baris atau satu kolom.

Bab 8 : Membahas tentang **Array 2D.** Konsep array 2D merupakan lanjutan dari array 1D, dimana penyimpanan nilai dapat dilakukan pada banyak baris dan banyak kolom.

Bab 9 : Membahas tentang **Sub Program (Metode/Modular).** Sub Program mempunyai 2 macam bentuk, yaitu Fungsi dan Prosedur.

Bab 10 : Membahas tentang **Stack dan Queue.** Konsep stack dan queue dipelajari terlebih dahulu sebelum belajar tentang konsep Rekursif. Pemanfaatan stack dan queue biasanya digunakan mengetahui urutan dan antrian.

Bab 11 : Membahas tentang **Rekursif.** Konsep rekursif merupakan fungsi yang memanggil dirinya sendiri. Prosesnya dapat menggunakan percabangan dan perulangan.

Bab 12 : Membahas tentang **Kumpulan Soal Kombinasi dan Pembahasan**. Pada bab ini akan disediakan berbagai macam soal menggunakan kombinasi konsep yang sudah dipelajari pada Bab 1 sampai Bab 11. Lengkap dengan pembahasan gambar flowchart dan pseudocodenya.

Semoga semua materi yang dibahas pada buku ajar ini dapat bermanfaat bagi para mahasiswa jurusan Sistem Informasi/Teknik Informatika. Penulis sangat berharap ada masukan berupa kritik dan saran yang membangun terhadap buku ajar ini, demi terciptanya kemajuan dan kualitas pendidikan yang lebih baik.

"Let Your Knowledge Grow More and More…"

Surabaya, Juni 2016

Endra Rahmawati

Daftar Isi

Daftar Gambar

Daftar Tabel

Deskripsi Mata Kuliah
Logika Algoritma

Mata Kuliah : Logika Algoritma

SKS/Semester : 3 SKS/ 1 (satu)

Kompetensi Lulusan : Memberikan kemampuan kepada mahasiswa untuk berpikir secara divergen dan runtun sebagai bagian untuk berpikir kritis, sehingga dapat **mengidentifikasi** masalah serta memberikan solusi secara sistematis.

Tujuan Pembelajaran (Learning Objective) :

Mahasiswa mampu menyusun algoritma berupa flowchart dan pseudocode sebagai solusi pemecahan masalah pada proses bisnis.

Baseline : Mahasiswa memiliki pengetahuan tentang operasi aritmatika.

Kompetensi Hardskill :

1. Mahasiswa dapat **menyebutkan** Konsep Dasar Algoritma, Penamaan Variabel, dan Jenis Penyajian Algoritma. **[K-1, C1]**
2. Mahasiswa dapat **menjelaskan dan memberikan contoh** penggunaan Struktur Kontrol dan Struktur Data dalam algoritma. **[K-2, C2]**
3. Mahasiswa dapat **mengembangkan** penggunaan Abstraksi Kontrol dan Penerapan Algoritma pada program bisnis. **[K-3, C2]**

Kompetensi Softskill :

a. Bekerja sama dalam kelompok
b. Tangguh
c. Jujur
d. Inisiatif
e. Tanggung jawab
f. Tepat waktu.

Pokok Bahasan :

Definisi dan Konsep Dasar Algoritma, Penamaan Variabel (Variabel dan Konstanta, Tipe Data dan Operator), Jenis Penyajian Algoritma (Flowchart dan Pseudocode). Struktur Kontrol (Percabangan dan Perulangan), Struktur Data (Array 1D dan 2D), dan Abstraksi Kontrol (Sub Program, Stack, Queue, Rekursif).

Media Belajar :

1. **Software :** Windows, Power Point, Aplikasi Algoritma
2. **Hardware :** Personal Computer/Laptop, LCD Projector, Sound System

Jenis Assesment :

1. **Tes :**
   a. Membuat Algoritma (Flowchart/Pseudocode) Kompetensi 1-2 (UTS)
   b. Membuat Algoritma (Flowchart/Pseudocode) sampai Kompetensi 3 (UAS)
2. **Non Tes :** Presentasi dan Diskusi, Dokumen/Paper.

Bab 1

Pengantar Logika Algoritma

Tujuan :

- ✓ Mahasiswa dapat menyebutkan definisi dan konsep dasar algoritma.
- ✓ Mahasiswa dapat menjelaskan metode pembuatan algoritma.
- ✓ Mahasiswa dapat menjelaskan perbedaan istilah pada pemrograman.

1.1 Pendahuluan

Dalam dunia komputer, algoritma menjadi hal yang sangat penting yang harus dikuasai sebelum melakukan kegiatan pemrograman. Algoritma merupakan kumpulan langkah atau tahapan untuk menyelesaikan suatu masalah. Perintah-perintah tersebut dapat dilakukan secara bertahap mulai dari awal hingga akhir. Permasalahan yang dapat diselesaikan menggunakan algoritma adalah tidak terbatas, dengan catatan untuk setiap masalah terdapat kondisi awal yang harus dipenuhi sebelum menjalankan algoritma. Sebuah algoritma dapat dikatakan selesai melakukan proses, dengan syarat semua kondisi awal terpenuhi. Beberapa algoritma sering menggunakan pengulangan (iterasi) atau menggunakan proses pengambilan keputusan (perbandingan atau percabangan).

Logika Algoritma adalah suatu cabang khusus dalam ilmu komputer yang mempelajari bagaimana para peserta didik dapat berpikir secara logis dalam pembuatan sebuah bentuk algoritma, baik menggunakan flowchart maupun pseudocode. Dalam cabang disiplin ini algoritma dipelajari melalui konsep dasar, terlepas dari sistem komputer atau bahasa pemrograman apa yang digunakan. Algoritma yang berbeda dapat diterapkan pada suatu masalah dengan kriteria yang sama. Hal ini dikarenakan adanya perbedaan cara berpikir setiap individu, namun tetap dalam satu tujuan yang sama.

Kompleksitas dari suatu algoritma merupakan ukuran banyak tidaknya jumlah langkah atau tahapan yang dibuat untuk menyelesaikan satu proses permasalahan. Secara rasional, algoritma yang dapat menyelesaikan suatu permasalahan dalam waktu singkat memiliki kompleksitas yang lebih sederhana, dibandingkan dengan algoritma yang membutuhkan waktu lama. Sehingga algoritma tersebut digolongkan sebagai algoritma dengan kompleksitas yang tinggi.

Ditinjau dari asal-usul katanya, kata Algoritma mempunyai cerita sejarah yang cukup panjang. Pada awalnya, kata *algorism* diartikan sebagai proses menghitung dengan angka arab. Apabila ada sesorang yang disebut sebagai *algorist* maka orang tersebut adalah ahli dalam perhitungan menggunakan angka arab. Namun, akhirnya para ahli sejarah matematika menemukan asal kata algoritma, yakni berasal dari nama seorang penulis buku arab yang terkenal yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarizmi. Penggalan kata Al-Khuwarizmi dibaca orang barat menjadi *Algorism*.

Al-Khuwarizmi menulis buku yang berjudul *Kitab Al Jabar Wal-Muqabala* yang artinya "Buku pemugaran dan pengurangan" (*The book of restoration and reduction*). Dari judul buku tersebut juga dapat diperoleh asal kata "Aljabar" (*Algebra*). Perubahan kata dari *algorism* menjadi *algorithm* muncul karena kata *algorism* sering disebut dengan *arithmetic*, sehingga akhiran *–sm* berubah menjadi *–thm*. Seiring berjalannya waktu, perhitungan dengan angka Arab sudah menjadi hal yang biasa. Secara berangsur-angsur kata *algorithm* digunakan sebagai metode perhitungan (komputasi) secara umum, sehingga makna kata aslinya mulai pudar. Dalam bahasa Indonesia, kata *algorithm* diserap menjadi *algoritma*.

1.2 Definisi Algoritma

Logika dapat diartikan sebagai penalaran atau masuk akal. Penalaran adalah salah satu bentuk pemikiran. Berdasarkan arti kata tersebut, logika didefinisikan sebagai ilmu yang harus menggunakan prinsip-prinsip tertentu agar dapat berfikir secara logis (tepat dan benar) menurut kaidah/aturan tertentu. Materi logika dapat membiasakan para peserta didik untuk berfikir secara logis dan sistematis. Cara berfikir logis dan sistematis inilah yang menjadi dasar penting pembuatan algoritma.

Berdasarkan *Merriam – Webster's Collegiate Dictionary*, istilah Algoritma diartikan sebagai prosedur langkah demi langkah untuk memecahkan masalah atau menyelesaikan suatu tugas. Kamus Besar Bahasa Indonesia mendefinisikan algoritma sebagai urutan logis pengambilan keputusan untuk pemecahan masalah.

Menurut Donald E.Knuth, sebuah algoritma harus memenuhi persyaratan sebagai berikut :

1. *Finiteness*.
   Algoritma harus berakhir (terminate) setelah melakukan sejumlah langkah proses.
2. *Definiteness*.
   Setiap langkah algoritma harus didefinisikan dengan tepat dan tidak menimbulkan makna ambigu.
3. *Input*.
   Setiap algoritma memerlukan data masukan untuk diolah.
4. *Output*.
   Setiap algoritma memberikan suatu keluaran (output) setelah mengerjakan proses.
5. *Effectiveness*.
   Langkah-langkah algoritma dikerjakan dalam waktu yang wajar.

Adapun pembagian alur proses dalam membentuk sebuah algoritma yang dibagi menjadi 3 proses besar, yaitu sebagai berikut :

1. **Proses Sekuensial *(Sequence Process).***
   Pada proses sekuensial, semua perintah/instruksi dijalankan secara berurutan, satu persatu, mulai dari yang pertama sampai yang terakhir.
   Contoh : Pembuatan Jus Buah, Pembuatan Mie Instan, Pembuatan Pisang Goreng.
   - **Pembuatan Jus Buah :**
     1) Siapkan buah yang akan dijus (Alpokat/Pepaya/Strawberry).
     2) Potong buah dalam bentuk dadu kecil.
     3) Siapkan Blender, pasang gelas blender dengan tepat.
     4) Masukkan potongan buah.
     5) Tambahkan gula dan susu sesuai selera.
     6) Blender potongan buah, gula, dan susu hingga tercampur rata.
     7) Siapkan gelas jus.
     8) Tuangkan hasil blender buah pada gelas jus.
     9) Jus siap dihidangkan.

   Dari contoh studi kasus di atas, langkah-langkah pembuatan jus buah harus dilakukan secara berurutan, mulai dari no.1 sampai no.9. Tidak boleh terbalik

ataupun terlewatkan satu langkah sekalipun. Apabila hal tersebut terjadi, maka hasil dari proses algoritma tidak dapat menghasilkan output yang benar.

2. **Proses Percabangan *(Selection/Decision Process)*.**

   Pada proses percabangan, sebuah perintah/instruksi dapat dijalankan dengan syarat/kondisi tertentu harus dipenuhi terlebih dahulu. Apabila syarat/kondisi belum terpenuhi, maka perintah/instruksi tersebut tidak akan pernah dijalankan. Syarat atau kondisi yang digunakan boleh lebih dari satu kondisi (tidak terbatas).

   Contoh : Pembelian baju di sebuah mall. Jika harga baju yang dibeli diatas Rp 300.000, maka pembeli akan mendapatkan 1 potong baju dengan jenis yang sama. Namun, apabila harga baju yang dibeli tidak melebihi harga Rp 300.000, maka pembeli tidak mendapatkan baju gratis.

   Dari contoh studi kasus di atas, syarat atau kondisi yang harus dipenuhi adalah pembelian dengan nominal diatas Rp 300.000.

3. **Proses Perulangan (Iteration/Looping Process).**

   Pada proses perulangan, sebuah perintah/instruksi dapat dijalankan secara berulang sampai kondisi tertentu. Kondisi pada proses perulangan ini biasanya digunakan sebagai batas awal dan batas akhir perulangan. Apabila tidak ada kondisi sebagai batas perulangan, maka proses pada algoritma tersebut akan berjalan terus-menerus, sehingga mengakibatkan adanya *out of space memory*.

   Contoh : Mencetak deret angka mulai dari angka 1 sampai 10, dengan kelipatan 1.

   Dari contoh tersebut, kondisinya adalah berupa batas awal dengan nilai = 1 dan batas akhir dengan nilai = 10. Proses perulangan akan menambahkan angka 1 sebagai kelipatannya. Sehingga pada akhir program, dapat menghasil output angka 1,2,3,4,5,6,7,8,9,10.

1.3 Elemen-Elemen Algoritma

Secara sederhana, algoritma dapat didefinisikan sebagai urutan langkah-langkah logis untuk menyelesaikan suatu masalah yang disusun secara sistematis. Berdasarkan definisi tersebut, algoritma dapat dibentuk ketika terjadi sebuah permasalahan. Permasalahan akan terselesaikan, jika dapat ditemukan solusi yang tepat. Hubungan antara algoritma, masalah dan solusi dapat digambarkan pada Gambar 1.1.

Gambar 1.1 Bagan Hubungan antara Algoritma, Masalah, dan Solusi.

Tahap Permasalahan merupakan proses terbentuknya permasalahan. Permasalahan dapat berupa masalah aritmatika, logika, atau sebuah proses bisnis. Tahap Algoritma merupakan proses pembuatan algoritma menggunakan bentuk penyajian tertentu. Sedangkan Tahap Solusi merupakan suatu program yang mengimplementasikan dari algoritma yang disusun. Adapun ciri-ciri dari sebuah algoritma yang baik adalah sebagai berikut :

a. Algoritma yang dibuat harus mengandung logika perhitungan atau metode yang tepat sesuai dengan permasalahan yang akan diselesaikan.
b. Nilai yang diinputkan harus sesuai, sehingga dapat menghasilkan output yang tepat dan benar dalam waktu yang singkat.
c. Algoritma dapat disajikan menggunakan flowchart atau pseudocode yang secara sistematis dan rapi, sehingga tidak menimbulkan arti ganda *(ambiguous)*.
d. Algoritma ditulis dengan format yang mudah dipahami dan mudah diimplementasikan ke dalam bahasa pemrograman.
e. Semua variabel yang dibutuhkan harus dideklarasikan dengan jelas.
f. Semua proses dalam algoritma harus berakhir (End/Stop) setelah melalui sejumlah langkah/proses.
g. Sebuah algoritma hanya dapat digunakan untuk menyelesaikan satu jenis permasalahan saja.

Seperti halnya komputer, yang mempunyai sejumlah elemen pendukung agar dapat melakukan sejumlah proses dengan tepat dan benar, pada algoritma juga mempunyai 3 elemen utama sebagai penentu keberhasilan seseorang membuat algoritma yaitu input, proses, dan output. Hubungan elemen utama algoritma antara input, proses, dan output dapat digambarkan pada Gambar 1.2.

Gambar 1.2. Hubungan elemen utama algoritma antara input, proses, dan output.

Keterangan :

- Input : Data inputan (masukan), baik berupa deklarasi variabel yang akan digunakan selama proses berjalan maupun variabel yang nilainya berasal dari inputan user.
- Proses : Tahap pengolahan data, baik menggunakan logika proses maupun model matematika yang sesuai.
- Output : Data output (keluaran) yang dihasilkan berdasarkan inputan dan proses yang terjadi pada sebuah algoritma.

Sebuah proses komputasi yang berjalan dengan baik, dapat menghasilkan (beberapa) nilai output dari (beberapa) nilai input yang diberikan. Dengan adanya 3 elemen utama algoritma tersebut, pembuatan algoritma harus melalui 3 tahap yang harus dilakukan, yaitu sebagai berikut :

a. Identifikasi semua variabel yang dibutuhkan sebagai data input.
b. Tentukan proses yang tepat untuk menyelesaikan permasalahan.
c. Tampilkan hasil output yang dibutuhkan sebagai solusi dari permasalahan.

Sebagai contoh mudah dalam membedakan jenis data input, proses dan data output dapat dilihat pada Gambar 1.3. Pada gambar tersebut, menampilkan bagaimana mengubah logika seseorang dari Proses Pembuatan Jus Buah dengan Pembuatan Algoritma menggunakan 3 elemen utama.

Pembuatan Jus Buah

Alat & Bahan :
- Blender 1 buah.
- Gelas Besar Jus 2 buah.
- Strawberry ¼ kg (10 buah).
- Es Batu 5 potong kecil.
- Gula 3 sendok makan.
- Susu 2 sendok makan.

Cara Pembuatan :
- Potong strawberry menjadi bentuk dadu kecil.
- Masukkan potongan strawberry ke dalam blender.
- Tambahkan gula dan susu.
- Masukkan es batu, aduk hingga tercampur rata.
- Bagi dan Tuangkan pada 2 gelas besar jus.

Hasil :
- 2 Gelas Jus Strawbery siap dihidangkan.

Pembuatan Algoritma Penjumlahan dan Pengurangan

Input (Variabel yang dibutuhkan) :
- Variabel Bilangan1
- Variabel Bilangan2
- Variabel HasilJumlah
- Variabel HasilKurang

Proses :
- Masukkan nilai Bilangan1
- Masukkan nilai Bilangan2
- Tambahkan Bilangan1 dengan Bilangan2, simpan hasil penjumlahan pada HasilJumlah.
- Kurangkan Bilangan1 dengan Bilangan2, simpan hasil pengurangan pada HasilKurang.

Output (Hasil akhir) :
- Tampilkan nilai variabel HasilJumlah.
- Tampilkan nilai variabel HasilKurang.

Gambar 1.3. Contoh Pembuatan Algoritma

1.4 Metode Pembuatan Algoritma

Algoritma dapat dibuat atau disajikan menggunakan dua teknik yaitu teknik tulisan dan teknik gambar. Teknik tulisan biasanya menggunakan metode *structure english* dan *pseudocode*, sedangkan teknik gambar biasanya menggunakan diagram alir (flowchart).

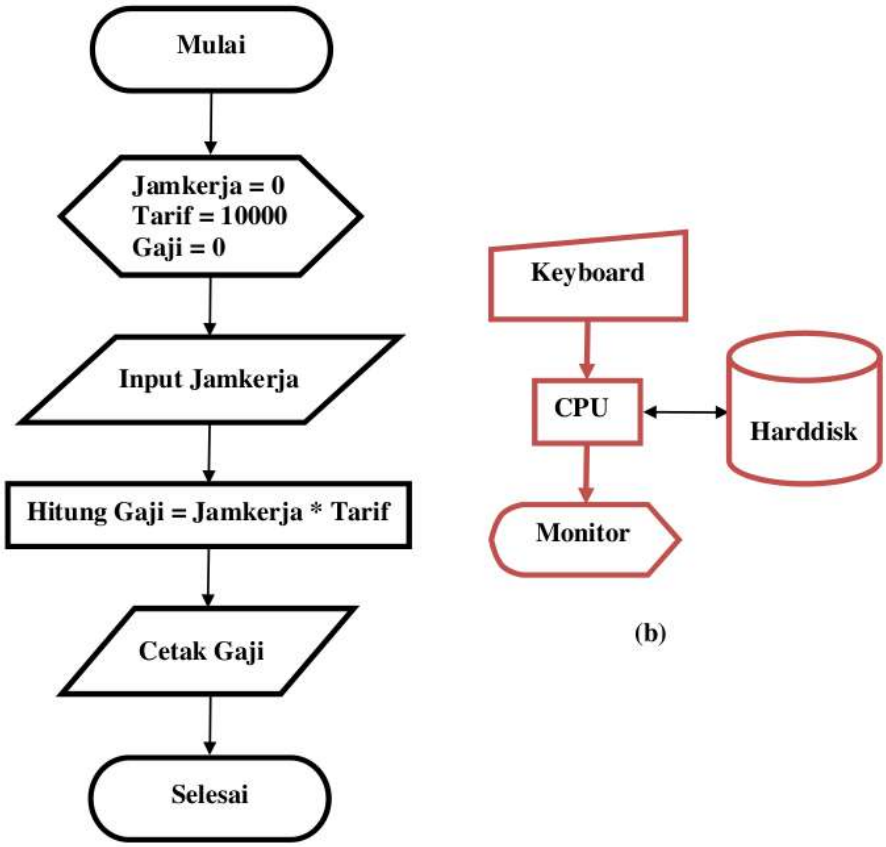
Tips:

> Dalam membuat algoritma (contoh: menggunakan flowchart).
> Sebelum membuat flowchart, lakukan identifikasi terlebih dahulu, berapa banyak variabel/peubah yang dibutuhkan atau digunakan dalam proses pembuatan algoritma.
> *Contoh :* Bila terdapat sebuah rumus
>
> **luas = panjang x lebar**
>
> Maka pada flowchart tersebut terdapat 3 buah variabel selama proses algoritma dijalankan. Pada inputan membutuhkan 2 variabel, yaitu variabel panjang dan lebar yang dapat diinputkan user. Sedangkan variabel luas sebagai variabel output.

1.4.1 Flowchart

Flowchart didefinisikan sebagai suatu bagan terurut untuk menggambarkan alur yang terjadi pada suatu proses, dengan menggunakan simbol – simbol tertentu. Flowchart terdiri dari 2 macam, yaitu flowchart program dan flowchart sistem. Namun, yang paling sering digunakan sebagai dasar pemrograman adalah flowchart program.

- **Flowchart program** merupakan bagan yang menggambarkan arus logika dari data yang akan diproses dalam suatu program dari awal sampai akhir. Flowchart program sangat berguna bagi programmer untuk mempersiapkan program yang rumit. Flowchart yang dimiliki terdiri dari simbol-simbol yang mewakili langkah program dan garis alir menunjukkan urutan yang akan dikerjakan. Contoh Flowchart Program dapat dilihat pada Gambar 1.4 (a).
- **Flowchart sistem** berbeda dengan flowchart program. Flowchart program sifatnya lebih terperinci tentang langkah-langkah proses di dalam program dari awal sampai akhir. Flowchart sistem hanya menggambarkan arus data dari sistem. Simbol-simbol yang digunakan pada flowchart sistem ada yang berbeda dengan simbol-simbol yang digunakan pada flowchart program.
  Contoh Flowchart Program dapat dilihat pada Gambar 1.4 (b).



Gambar 1.4. Contoh Flowchart Program (a) dan Flowchart Sistem (b)

1.4.2 Pseudocode

Pseudocode didefinisikan sebagai bentuk penyajian algoritma dengan menggunakan struktur bahasa tertentu. Pseudocode berasal dari kata *pseudo* yang berarti imitasi/mirip/menyerupai. Sedangkan *code* berarti kode program. Dari asal kata tersebut, kode program yang paling sering digunakan dalam penulisan pseudocode adalah struktur bahasa pascal dan C. Namun, dalam buku ajar ini pseudocode akan lebih banyak ditulis menggunakan struktur bahasa C/ Java.

Contoh algoritma dalam struktur Bahasa Indonesia :

```
Baca data jam_kerja
Hitung gaji adalah jam_kerja dikalikan tarif
Tampilkan gaji
```

Pseudocode dengan Pascal :

```
Read jam_kerja
Gaji := jam_kerja * tarif
Write gaji
```

Pseudocode dengan C/Java :

```
Input jam_kerja
Gaji = jam_kerja * tarif
Print gaji
```

Dengan adanya flowchart dan pseudocode tersebut dalam menyajikan algoritma, ada beberapa manfaat yang dapat diperoleh, yaitu sebagai berikut :

a. Memudahkan penelusuran alur proses.
b. Mempercepat proses pencarian lokasi kesalahan dalam pemrograman.
c. Dokumentasi.

1.5 Program dan Pemrograman

Hal terpenting dalam menjalankan komputer adalah program. Dalam pemrograman dikenal beberapa bahasa pemrograman, seperti juga manusia mengenal bahasa-bahasa yang digunakan untuk berkomunikasi. Manusia dalam berkomunikasi menggunakan kata atau karakter sedangkan komputer dengan kode 0 dan 1. Untuk mempermudah manusia berkomunikasi dengan komputer, maka diciptakan bahasa pemrograman. Dengan adanya bahasa pemrograman ini, bila manusia ingin berkomunikasi dengan komputer tidak harus menerjemahkan ke dalam 0 dan 1.

Adapun beberapa istilah pada pemrograman, yaitu :

- **Program** merupakan susunan instruksi (kata, ekspresi, pernyataan atau kombinasinya) yang dirangkai dan disusun manjadi satu kesatuan prosedur, berupa urutan langkah untuk menyelesaikan masalah, dan mengimplementasikan dengan menggunakan bahasa pemrograman, sehingga dapat dieksekusi oleh komputer.
- **Bahasa pemrograman** merupakan prosedur atau tata cara penulisan program. Dalam bahasa pemrograman, terdapat dua faktor penting yaitu sintaksis dan semantik. Sintak adalah aturan-aturan gramatikal yang mengatur tata cara penulisan kata, ekspresi dan pernyataan sedangkan semantik adalah aturan-aturan untuk menyatakan suatu arti.

Bahasa Pemrograman dibagi menjadi 3 level, yaitu sebagai berikut :

1. **Bahasa Pemrograman Tingkat Rendah *(Low Level Language)***

   Merupakan bahasa yang berorientasi pada mesin. Pemrograman menggunakan bahasa ini harus berpikir berdasarkan logika mesin berpikir, sehingga bahasa ini kurang fleksibel dan sulit dipahami.

   Contoh : Bahasa Mesin, Assembly.

2. **Bahasa Pemrograman Tingkat Menengah *(Middle Level Language)***

   Merupakan bahasa pemrograman yang menggunakan aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar yang mudah dipahami manusia serta memiliki instruksi-instruksi tertentu yang langsung bisa diakses oleh komputer.

   Contoh : C.

3. **Bahasa Pemrograman Tingkat Tinggi *(High Level Language)***

   Merupakan bahasa pemrograman yang menggunakan aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar bahasa yang langsung dapat dipahami oleh manusia.

   Contoh : Pascal, Cobol, Power Basic.

- **Pemrograman** merupakan proses mengimplementasikan urutan langkah untuk menyelesaikan suatu masalah dengan menggunakan suatu bahasa pemrograman.
- **Pemrograman Terstruktur** merupakan proses mengimplementasikan urutan langkah-langkah untuk menyelesaikan suatu masalah dalam bentuk program yang memiliki rancang bangun yang terstruktur dan tidak berbelit-belit sehingga mudah ditelusuri, dipahami dan dikembangkan oleh siapa saja.
- **Pemrograman Berorientasi Objek** merupakan proses mengimplementasikan urutan langkah-langkah untuk menyelesaikan suatu masalah dalam bentuk program yang berorientasi pada objek dan class. Contoh bahasa pemrograman yang berorientasi objek adalah Visual Basic, C++, Java.

Adapun **Fungsi Pemrograman** adalah sebagai berikut :

a. Sebagai media untuk menyusun dan memahami pemrograman dalam komputer.
b. Sebagai alat komunikasi antara pembuat program dengan komputer.
c. Sebagai alat komunikasi antar pembuat program.

1.6 Tahapan Pembuatan Algoritma

Tahapan pembuatan algoritma terdiri dari 3 tahapan besar, yaitu tahapan persiapan, tahapan identifikasi masalah, dan tahapan pembuatan.

1. **Tahapan Persiapan**
   - Identifikasi masalah (menentukan input, proses, output).
   - Menentukan variabel yang terlibat beserta typenya.
   - Membuat flowchart dan atau pseudocode.
   - Simulasi terhadap flowchart atau pseudocode yang sudah disusun.
2. **Tahapan Identifikasi Masalah**
   - Menentukan Input dan Output : menentukan variabel yang digunakan untuk menyimpan nilai masukan dan nilai keluaran.
   - Proses : menyusun model matematis dari permasalahan yang diselesaikan. Dalam proses harus kelihatan hubungan antara variabel masukan dan keluarannya.
3. **Tahapan Pembuatan**
   - Berdasarkan pseudocode, mencari *statement/reserved word* yang dibutuhkan.
   - Menyusun program sesuai dengan aturan yang ditetapkan.
   - Melakukan kompilasi.

1.7 Rangkuman

- Algoritma adalah urutan instruksi spesifik secara bertahap yang harus dilakukan dalam menyelesaikan masalah.
- Pembagian alur proses dalam algoritma dibagi menjadi 3 proses, yaitu :
  - Proses Sekuensial *(Sequence Process)*
  - Proses Percabangan *(Selection/Decision Process)*
  - Proses Perulangan *(Iteration/Looping Process)*
- Elemen-elemen algoritma terdiri dari 3 elemen utama, yaitu :
  - Input
  - Proses
  - Output
- Bentuk penyajian algoritma dibagi menjadi 2, yaitu :
  - Flowchart
  - Pseudocode

1.8 Latihan Soal

1. Buat algoritma untuk menghitung rata-rata dari nilai 3 orang mahasiswa. Nilai mahasiswa boleh ditentukan atau berasal dari inputan user. Tampilkan hasil perhitungan nilai rata-rata. **(Gunakan 3 elemen utama, yaitu input, proses, dan output).**
2. Buat algoritma untuk mengurutkan 3 orang berdasarkan tinggi badan. Tinggi badan orang boleh ditentukan langsung atau berasal dari inputan user. Tampilkan hasil urutan mulai dari yang paling pendek sampai yang paling tinggi. **(Gunakan 3 elemen utama, yaitu input, proses, dan output).**
3. Buat algoritma untuk menghitung keliling dan luas persegi. Panjang dan lebar persegi boleh ditentukan atau berasal dari inputan user. Tampilkan hasil perhitungan keliling dan luas persegi. **(Gunakan 3 elemen utama, yaitu input, proses, dan output).**

Bab 2

Pengenalan Tipe Data

Tujuan :

- ✓ Mahasiswa dapat menguraikan berbagai macam tipe data dan operator.
- ✓ Mahasiswa dapat menunjukkan penggunaan tipe data, operator, dan ekspresi.

2.1 Pengantar Tipe Data

Sebuah Program yang dibuat pada komputer bekerja dengan memanipulasi atau mengolah objek (data) untuk diubah menjadi sebuah informasi yang bermanfaat. Hasil dari manipulasi data tersebut akan disimpan dalam memory komputer. Penyimpanan hasil manipulasi data dilakukan dalam bentuk variabel atau konstanta dengan nama tertentu dan tipe data tertentu. Suatu tipe data menyatakan pola penyajian data dalam memori komputer, seperti alamat yang dapat menunjukkan dimana objek data disimpan.

Data yang akan diprogram memiliki tipe yang bermacam-macam. Tipe data pada algoritma dapat dikelompokkan menjadi dua, yaitu: tipe data dasar dan tipe data bentukan. Tipe dasar adalah tipe yang dapat langsung dipakai, sedangkan tipe bentukan merupakan tipe yang dibentuk dari tipe dasar, atau dari tipe bentukan lain yang sudah didefinisikan.

2.2 Pembagian Tipe Data

Berdasarkan ulasan pada sub bab sebelumnya, tipe data pada algoritma dibagi menjadi 2 jenis, yaitu sebagai berikut :

1. **Tipe Data Dasar**

   Tipe dasar adalah tipe yang dapat langsung dipakai. Tipe data dasar dibagi menjadi 5 jenis tipe data, diantaranya sebagai berikut :

a. Integer (Bilangan Bulat)

Tipe data integer digunakan untuk bilangan atau angka yang bulat (bukan pecahan/desimal). Artinya angka yang tidak mengandung koma.

Contoh : **1**, **7**, **25**, **1000**, **352387**, dst

b. Real (Bilangan Riil/Pecahan)

Tipe data real digunakan untuk bilangan atau angka yang sifatnya pecahan/desimal (bukan bilangan bulat). Artinya angka yang mengandung koma.

Contoh : **0.8**, **3.14**, **27.5698**, **250.3749**, dst.

Notes :

Pada bahasa pemrograman Java, penulisan bilangan real yang menggunakan tanda koma digantikan dengan tanda titik.

c. String (Kata/Kalimat)

Tipe data string digunakan untuk penulisan kata/kalimat, dimana kata tersebut berasal dari sederetan karakter (huruf alphabet dan/atau angka). Penulisan tipe data string harus diapit dengan tanda petik ganda (“ ”).

Contoh :

- Untuk Penulisan Nama → **“Endra Rahmawati”**
- Untuk Penulisan NIM → **“1241010121”**
- Untuk Penulisan Alamat → **“Jl. Mangga No. 8 Surabaya”**

Notes :

- ✓ Penulisan kata atau kalimat pada tipe data string dapat menggunakan spasi, titik, dan koma di dalamnya.
- ✓ Tipe data string dapat diisi dengan angka dengan syarat bahwa angka tersebut tidak digunakan untuk proses operasi aritmatika.
- ✓ Penggabungan 2 tipe data string atau lebih dapat dilakukan dengan menggunakan operator plus (+).
  Contoh : “Endra” + “ Rahmawati” = “Endra Rahmawati”

d. Char (Karakter)

Tipe data char digunakan untuk penulisan satu huruf alphabet, satu angka (bulat/desimal), satu tanda baca, dll. Penulisan tipe data char harus diapit dengan tanda petik tunggal (‘ ‘).

Contoh :

- Untuk Penulisan Huruf Alphabet → **‘A’, ‘C’, ‘S’, ‘Z’**

- Untuk Penulisan Angka → **'6', '45', '2014', '3.24'**
- Untuk Penulisan Tanda Baca → **'@', '#', '?'**

e. Boolean (Logika)

Tipe data boolean merupakan tipe data yang hanya mempunyai 2 nilai saja, yaitu benar (true) dan salah (false). Tipe data ini biasanya digunakan untuk mengetahui nilai kondisi pada proses percabangan.

Contoh : (3<10) : **true**, (12==15) : **false**.

2. Tipe Data Bentukan

Tipe bentukan merupakan tipe yang dibentuk dari tipe data dasar, atau dari tipe bentukan lain yang sudah didefinisikan oleh seorang programmer. Terdapat 2 tipe data bentukan, yaitu :

a. Tipe Data Dasar yang diberi nama Tipe Data Baru

Penggunaan tipe data baru dapat dilakukan apabila seorang programmer menginginkan nama yang lebih mudah dipahami oleh orang lain yang membaca teks algoritma. Pemberian nama baru untuk tipe data dasar dapat dilakukan dengan menambahkan kata kunci *type*.

Contoh: *type* BilanganBulat : integer.

b. Tipe Data Terstruktur

Tipe data terstruktur merupakan tipe yang berbentuk rekaman (*record*). Rekaman tersusun atas satu atau lebih field dengan nama tipe tertentu.

Contoh : Penulisan koordinat titik A (x1, y1) dan B (x2, y2).

type TitikA : *record* <x1 : real, y1 : real>
type TitikB : *record* <x2 : real, y2 : real>

atau

type TitikA : *record* <x1 , y1 : real>
type TitikB : *record* <x2 , y2 : real>

2.3 Variabel

Variabel merupakan suatu lokasi memori komputer yang digunakan untuk menampung dan menyimpan data yang akan diolah. **Ada beberapa ciri-ciri variabel, yaitu sebagai berikut :**

a. Penamaannya bersifat UNIK, artinya harus berbeda dengan variabel lainnya yang digunakan pada algoritma.
b. Satu variabel hanya diperuntukkan menyimpan satu jenis data saja.

c. Tidak tergantung pada besarnya data.

d. Nilainya bisa berubah-ubah.

Selain ciri-ciri variabel yang telah dituliskan diatas, terdapat juga aturan-aturan yang harus dipenuhi dalam penulisan sebuah variabel pada algoritma. **Aturan penulisan variabel diantaranya yaitu :**

a. Harus diawali dengan huruf, **biasanya menggunakan huruf kecil semua** dalam penulisannya.

Contoh :

- **bilangan**
- **jumlah**
- **hasil**
- **angka**
- **jual**
- **beli**

b. Tidak boleh diawali dengan angka. Namun, apabila angka ditulis setelah penulisan huruf, boleh dilakukan.

Contoh :

- **1angka → (tidak boleh)**
- **10bilangan → (tidak boleh)**
- **5buah → (tidak boleh)**
- **angka1 → (boleh)**
- **nimmhske1 → (boleh)**

c. Panjangnya terukur, artinya pemberian nama variabel tidak boleh terlalu panjang. Cukup gunakan 1 atau 2 kata saja.

Contoh :

- **hasilbagi → (boleh)**
- **namamhs → (boleh)**
- **namamhsstikomsurabaya → (tidak disarankan/tidak boleh)**
- **hasilpenjualantokoserbaguna → (tidak disarankan/tidak boleh)**

d. Tidak boleh menggunakan spasi.

- **hasilkali → (boleh)**
- **namabuah → (boleh)**
- **hasil kali → (tidak boleh, karena mengandung spasi)**

- **nama buah → (tidak boleh, karena mengandung spasi)**

e. Apabila nama variabel terdiri dari 2 kata, **sebaiknya gunakan aturan penulisan *camel case*** (Huruf besar dibagian tengah, seperti punuk unta).

Contoh :

- **hasilPenjumlahan**
- **namaMhs**
- **almtRumah**

f. **Penulisan nama variabel harus diikuti dengan tipe data.**

Contoh :

- **namaKaryawan : String**
- **bilangan1 : Integer**
- **diskon : Real**
- **karakter : Char**

2.4 Konstanta

Selain variabel yang selalu berubah, ada juga variabel yang nilainya tetap, yang biasa disebut sebagai Konstanta. Konstanta hampir sama dengan variabel**. Konstanta merupakan variabel yang mempunyai besaran atau nilai yang tetap selama program dijalankan**. Seperti halnya variabel, terdapat beberapa **aturan yang harus diperhatikan saat penulisan konstanta pada algoritma,** diantaranya sebagai berikut :

a. Penulisan konstanta selalu **diawali dengan kata *Const*.**

Contoh :

- **Const phi**
- **Const panjang**
- **Const password**

b. Nama konstanta boleh ditulis menggunakan huruf besar (jarang digunakan). Namun biasanya hampir sama dengan penulisan nama variabel, yaitu **menggunakan huruf kecil semua.**

Contoh :

- **Const Nmaks (jarang digunakan)**
- **Const User (jarang digunakan)**
- **Const password (sering digunakan)**
- **Const user (sering digunakan)**

c. Tambahkan besaran atau **nilai konstantanya.**

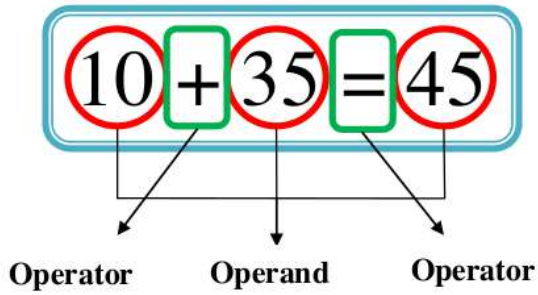
Contoh :

- **Const phi = 3.14**
- **Const panjang = 100**
- **Const password = "logikaku"**

Notes : *Konstanta dapat dibuat sendiri, tidak harus berasal dari konstanta aritmatika yang dikenal selama ini. Contoh : phi yang selalu bernilai 3.14.*

2.5 Ekspresi

Ekspresi adalah pernyataan atau sebuah proses yang mentransformasikan nilai menjadi hasil (keluaran) yang diinginkan melalui proses perhitungan (komputasi). Ekspresi terdiri dari **Operator dan Operand.** Operand adalah nilai yang diberikan, dapat berupa variabel, konstanta, nilai, dan nilai balik dari fungsi. Sedangkan operator merupakan penghubung antar operand.



Gambar 2.1. Perbedaan Operator dan Operand

2.6 Macam-macam Ekspresi

Ekspresi dapat digunakan di berbagai macam kebutuhan, sesuai proses yang telah ditentukan. Salah satunya adalah dibutuhkan untuk operasi aritmatika, penjumlahan, pengurangan, perkalian, atau pembagian. Adapun pembagian ekspresi pada logika lagoritma dibagi menjadi 3, yaitu :

1. **Ekspresi Aritmatika**

   Ekspresi aritmatika merupakan ekspresi yang digunakan untuk proses perhitungan. Misalnya untuk penjumlahan, pengurangan, perkalian, pembagian, termasuk menampung hasil (keluaran) dari proses perhitungan tersebut.

- **Operator yang digunakan :** ^, *, /, DIV, MOD, +, - , = , ←

  Untuk lebih jelas mengenai prioritas dalam penggunaan operator, lihat pada Tabel 2.1.
- **Operand yang digunakan :** numerik (angka)
- **Hasil (Keluaran) berupa :** numerik (angka)
- **Contoh :**
  - A*B
  - x = (k*i) mod 2
  - hasil = bil1 – bil2

2. Ekspresi Relasi

Ekspresi relasi merupakan ekspresi yang digunakan pada algoritma yang mengandung proses percabangan, tepatnya pada kondisi yang berlaku sebagai syarat/aturan dari sebuah permasalahan.

- **Operator yang digunakan :** <, >, <>, = , >=, <=, NOT, AND, OR

  Operator Relasi digunakan untuk membandingkan hubungan antara 2 buah operand dan hasil yang didapatkan merupakan keluaran bertipe boolean, true n false.
- **Operand yang digunakan :** numerik, string
- **Hasil (Keluaran) berupa :** Boolean (true, false)
- **Contoh :**

  A = 10

  B = 30

  C = 10

  A <> C = false

  Untuk lebih jelasnya, perhatikan hasil output penggunaan operator relasi untuk ketiga bilangan tersebut pada Tabel 2.2

3. Ekspresi String

Ekspresi String digunakan untuk menggabungkan 2 buah string atau lebih. Penggabungan string dilakukan menggunakan operator +.

- **Operator yang digunakan : +**
- **Operand yang digunakan :** string
- **Hasil (Keluaran) berupa :** string

- **Contoh :**

  kata1 ← "STIKOM"

  kata2 ← "Surabaya Indonesia"

  kata1 + kata2 = "STIKOMSurabaya Indonesia"

2.7 Operator Aritmatika

Operator Aritmatika merupakan operator yang digunakan pada ekspresi aritmatika. Operand yang digunakan dapar berupa numerik (integer atau real). Adapun prioritas penggunaan operator aritmatika adalah sebagai berikut :

Tabel 2.1. Prioritas Penggunaan Operator Aritmatika

Prioritas	Operator	Operasi	Tipe Operand	Tipe Hasil
1	^	Pangkat	Real, real Integer, real Real, integer	Real Integer real
2	*	Perkalian	Real,real Integer, integer Real, integer	Real Integer real
	/	Pembagian real	Real, real Integer, integer Real, integer	Real Real Real
	DIV	Pembagian bulat	Integer, integer	integer
	MOD	Sisa Pembagian	Integer, integer	Integer
3	+	Penjumlahan	Real,real Integer, integer Real, integer	Real Integer real
	-	Pengurangan	Real,real Integer, integer Real, integer	Real Integer Real
4	= atau ←	Pemuatan Nilai	Integer Real	Integer Real

2.8 Operator Relasi

Operator Relasi merupakan operator yang digunakan pada ekspresi relasi. Operand yang digunakan dapat berupa numerik ataupun string. Hasil output hanya mempunyai 2 nilai saja, yaitu true dan false. Adapun contoh penggunaan operator relasi adalah sebagai berikut :

Tabel 2.2. Penggunaan Operator Relasi dan Hasil Output

Operator	Arti	Contoh	Hasil Output
=	Sama dengan	A= B	false
>	Lebih dari	B > A	true
<	Kurang dari	B < C	false
>=	Lebih dari atau sama dengan	A >=C	true
<=	Kurang dari atau sama dengan	A <=B	true
<>	Tidak sama dengan	A<>C	false

2.9 Operator Logika

Selain operator relasi, operator logika juga merupakan operator yang digunakan pada ekspresi relasi. Operator ini diterapkan pada kondisi yang berlaku sebagai syarat/aturan dari sebuah permasalahan. Operator logika berfungsi untuk menghubungkan 2 buah nilai dan akan menghasilkan output nilai true dan false. Adapun prioritas penggunaan operator logika adalah sebagai berikut :

Tabel 2.3. Prioritas Penggunaan Operator Logika

Prioritas	Operator	Arti
1	NOT	Komplemen Logika
2	AND	Perbandingan secara DAN
3	OR	Perbandingan secara OR

Contoh :

- totalPembelian = 200000
- jenisbarang = “kemeja”
- (totalPembelian == 200000) **AND** (jenisbarang==”kemeja”)

2.10 Rangkuman

- Sebuah tipe data menyatakan pola penyajian data dalam memori komputer, seperti alamat yang dapat menunjukkan dimana objek data disimpan.
- Pembagian tipe data pada algoritma dibagi menjadi 2 bentuk, yaitu :
  - Tipe Data Dasar, mencakup Integer, Real, String, Char, dan Boolean.
  - Tipe Data Bentukan, mencakup Tipe data dasar yang diberi nama tipe data baru dan tipe data terstruktur.
- Variabel merupakan suatu lokasi memori komputer yang digunakan untuk menampung dan menyimpan data yang akan diolah.
- Konstanta merupakan variabel yang mempunyai besaran atau nilai yang tetap selama program dijalankan.
- Ekpresi pada algoritma dibagi menjadi 3 bagian, yaitu :
  - Ekspresi Aritmatika
  - Ekspresi Relasi
  - Ekspresi String
- Operator pada algoritma juga dibagi menjadi 3 bagian, yaitu :
  - Operator Aritmatika
  - Operator Relasi
  - Operator Logika

2.11 Latihan Soal

Diketahui : A = 10 B = 15 C = 20 D = 2.5	Diketahui : A = Benar B = Benar C = Salah D = Salah
Tentukan nilai ekspresi berikut ini : a. A*2+B b. (A*2)+C c. A*(2+D) d. C+B^2 e. A + (C-B^3)/7 f. D*2 + A*B g. (C-D)* 4 + C Mod A h. B / D * 5 i. 7 Mod 2 + C Mod B j. A * D / C	Tentukan nilai ekspresi berikut ini : a. A and B b. B or C c. A or B and C d. Not A or B and C e. Not C and D f. A or B or C and D g. A and C or (Not D) h. (Not B or D) and (A or B) i. B or (C and D) and A j. Not (D and B) or Not (A and C) k. Not (A or B) and Not (C or D)

Notes : *Perhatikan prioritas penggunaan operator, baik operator aritmatika, operator relasi maupun operator logika.*

Perhatikan penulisan pencatatan untuk Data Pemeriksaan Balita di bawah ini. **Tentukan Tipe Data** yang digunakan dan **tuliskan kembali ke dalam bentuk/notasi algoritma yang benar.**

Pencatatan Data Pemeriksaan Balita di Puskesmas Sumber Waras.

Tanggal	: 17 Agustus 2021
Pukul	: 09.00 WIB
Nama Balita	: Putri Permadi
Tempat Lahir	: Surabaya
Tanggal Lahir	: 12 Oktober 2017
Usia (tahun)	: 4
Berat Badan (kg)	: 16,5
Tinggi Badan (cm)	: 100,3
Hasil Pemeriksaan	: Sehat dan BB Ideal

Bab 3

Flowchart dan Pseudocode

Tujuan :

- ✓ Mahasiswa dapat mendemonstrasikan jenis penyajian algoritma menggunakan flowchart.
- ✓ Mahasiswa dapat mendemonstrasikan jenis penyajian algoritma menggunakan pseudocode.

3.1 Flowchart

Dalam Bahasa Inggris, flowchart berasal dari 2 buah kata yaitu *flow* dan *chart*. Apabila diterjemahkan ke dalam Bahasa Indonesia, flowchart dapat diartikan sebagai diagram alir. Berdasarkan dua kata tersebut, dapat digambarkan bahwa flowchart pasti berbentuk diagram dan dapat mengalirkan sesuatu. Hal tersebut memang benar, karena pada dasarnya flowchart memang menggambarkan suatu aliran kegiatan dari awal hingga akhir sesuai tahapan atau langkah-langkah dalam menyelesaikan suatu masalah.

Permasalahan tersebut dapat beraneka ragam, mulai dari masalah yang paling sederhana hingga masalah yang paling kompleks. Dalam hal ini, permasalahan yang dihadapi peserta didik merupakan masalah pemrograman yang dapat diselesaikan menggunakan komputer. Namun, secara logika dapat diawali dengan mengamati permasalahan dalam kehidupan sehari-hari kita. Contoh masalah paling sederhana adalah masalah pembuatan kue tart. Dalam membuat kue tart, diperlukan tahapan atau langkah-langkah yang berurutan supaya hasilnya dapat sesuai dengan apa yang diinginkan. Sama halnya dalam kegiatan pemrograman, diperlukan suatu algoritma (urutan tahapan atau langkah-langkah logis untuk menyelesaikan masalah dan disusun secara sistematis) agar program dapat berjalan dengan baik dan memberikan hasil output yang benar (valid). Untuk menyajikan atau merepresentasikan algoritma tersebut, dibutuhkan sebuah flowchart.

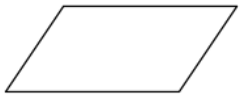
Flowchart biasanya dipelajari pada awal mempelajari pemrograman. Mengapa demikian? Hal ini tak lain karena dengan mempelajari flowchart, diharapkan dapat berfikir secara logis, dapat menentukan komponen program (input dan output), serta memahami alur proses program. Flowchart merupakan teknik yang memudahkan programmer, dengan tujuan memudahkan dan meminimalisir kesalahan, agar tidak ada komponen program yang tertinggal.

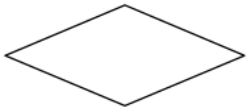
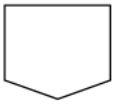
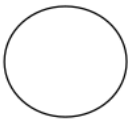
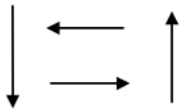
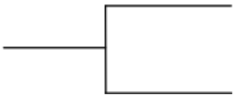
3.1.1 Definisi Flowchart

Flowchart didefinisikan sebagai suatu bagan terurut untuk menggambarkan alur yang terjadi pada suatu proses, dengan menggunakan simbol – simbol tertentu. Flowchart diawali dengan penerimaan input, pemrosesan input, dan diakhiri dengan penampilan output. Flowchart terdiri dari 2 macam, yaitu flowchart program dan flowchart sistem. Namun, yang paling sering digunakan sebagai dasar pemrograman adalah flowchart program.

- **Flowchart program** merupakan bagan yang menggambarkan arus logika dari data yang akan diproses dalam suatu program dari awal sampai akhir. Flowchart program sangat berguna bagi programmer untuk mempersiapkan program yang rumit. Flowchart yang dimiliki terdiri dari simbol-simbol yang mewakili langkah program dan garis alir menunjukan urutan yang akan dikerjakan. Adapun simbol-simbol flowchart yang termasuk dalam flowchart program adalah sebagai berikut :

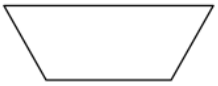
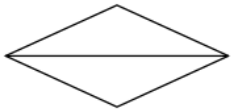
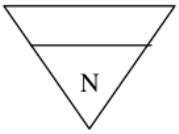
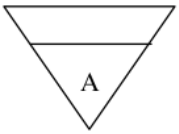
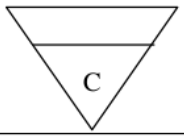
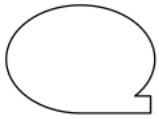
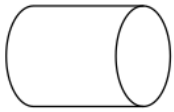
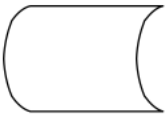
Tabel 3.1. Simbol Flowchart Program

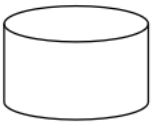
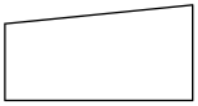
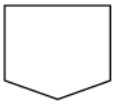
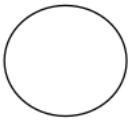
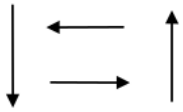
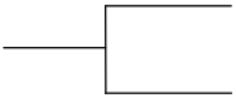
1.		**SIMBOL TERMINAL** Simbol yang digunakan untuk menyatakan awal atau akhir sebuah algoritma/program.
2.		**SIMBOL PREPARATION/LOOPING** Simbol yang digunakan untuk memberikan nilai awal pada suatu variabel atau digunakan untuk proses perulangan.
3.		**SIMBOL INPUT/OUTPUT** Simbol yang digunakan untuk menunjukkan operasi masukan (input) atau keluaran (ouput).

4.		**SIMBOL PROSES** Simbol yang digunakan untuk menggambarkan proses pengolahan data.
5.		**SIMBOL SELECTION/DECISION** Simbol yang digunakan untuk menyatakan suatu pilihan berdasarkan suatu kondisi tertentu.
6.		**SIMBOL PREDEFINED PROCESS** Simbol yang digunakan untuk proses yang detilnya dijelaskan terpisah, misal dalam bentuk *subroutine* (fungsi/prosedur).
7.		**SIMBOL PENGHUBUNG KE HALAMAN LAIN** Simbol yang digunakan untuk menghubungkan bagian diagram alir pada halaman yang berbeda.
8.		**SIMBOL PENGHUBUNG KE HALAMAN YANG SAMA** Simbol yang digunakan untuk menghubungkan bagian diagram alir pada halaman yang sama.
9.		**SIMBOL ARAH ALIRAN** Simbol yang digunakan untuk menunjukkan arah aliran proses.
10.		**SIMBOL ANOTASI** Simbol yang digunakan untuk memberikan keterangan-keterangan untuk memperjelas simbol-simbol lain

- **Flowchart sistem** berbeda dengan flowchart program. Flowchart program sifatnya lebih terperinci tentang langkah-langkah proses di dalam program dari awal sampai akhir. Flowchart sistem hanya menggambarkan arus data dari sistem. Simbol-simbol yang digunakan pada flowchart sistem ada yang berbeda dengan simbol-simbol yang digunakan pada flowchart program. Adapun simbol-simbol flowchart yang termasuk dalam flowchart program adalah sebagai berikut :

Tabel 3.2. Simbol Flowchart Sistem

1.		**SIMBOL DOKUMEN** Simbol yang menunjukkan dokumen yang digunakan untuk input dan output baik secara manual maupun komputerisasi.
2.		**SIMBOL OPERASI MANUAL** Simbol yang menunjukkan pekerjaan yang dilakukan secara manual.
3.		**SIMBOL PROSES** Simbol yang menunjukkan kegiatan proses operasi program komputer.
4.		**SIMBOL PENGURUTAN** Simbol yang menunjukkan proses pengurutan dokumen di luar komputer.
5.		**SIMBOL OFFLINE STORAGE** Simbol yang yang menunjukkan file non komputer yang diarsip urut angka (Numeric).
6.		**SIMBOL OFFLINE STORAGE** Simbol yang yang menunjukkan file non komputer yang diarsip urut huruf (Alphabet).
7.		**SIMBOL OFFLINE STORAGE** Simbol yang yang menunjukkan file non komputer yang diarsip urut tanggal (Chronological).
8.		**SIMBOL MAGNETIC TAPE** Simbol yang menunjukkan Input Output yang menggunakan pita magnetic.
9.		**SIMBOL MAGNETIC DRUM** Simbol yang menunjukkan Input Output yang menggunakan drum magnetic.
10.		**SIMBOL MAGNETIC STORAGE** Simbol yang menunjukkan Input Output yang menggunakan disket.

11.		**SIMBOL HARDDISK STORAGE** Simbol yang menunjukkan Input Output yang menggunakan hard disk.
12.		**SIMBOL PUNCHED CARD** Simbol yang menunjukkan Input Output yang menggunakan kartu plong.
13.		**SIMBOL PUNCHED TAPE** Simbol yang menunjukkan Input Output yang menggunakan kertas berlubang.
14.		**SIMBOL PUNCHED TAPE** Simbol yang menunjukkan Input Output yang menggunakan online keyboard.
15.		**SIMBOL DISPLAY** Simbol yang menunjukkan Output yang Ditampilkan di layar terminal.
16.		**SIMBOL PENGHUBUNG KE HALAMAN LAIN** Simbol yang digunakan untuk menghubungkan bagian diagram alir pada halaman yang berbeda.
17.		**SIMBOL PENGHUBUNG KE HALAMAN YANG SAMA** Simbol yang digunakan untuk menghubungkan bagian diagram alir pada halaman yang sama.
18.		**SIMBOL ARAH ALIRAN** Simbol yang digunakan untuk menunjukkan arah aliran proses.
19.		**SIMBOL ANOTASI** Simbol yang digunakan untuk memberikan keterangan-keterangan untuk memperjelas simbol-simbol lain

Notes : *Contoh penerapan penggunaan flowchart program dan flowchart sistem dapat dilihat pada Gambar 1.4 (a) dan 1.4 (b).*

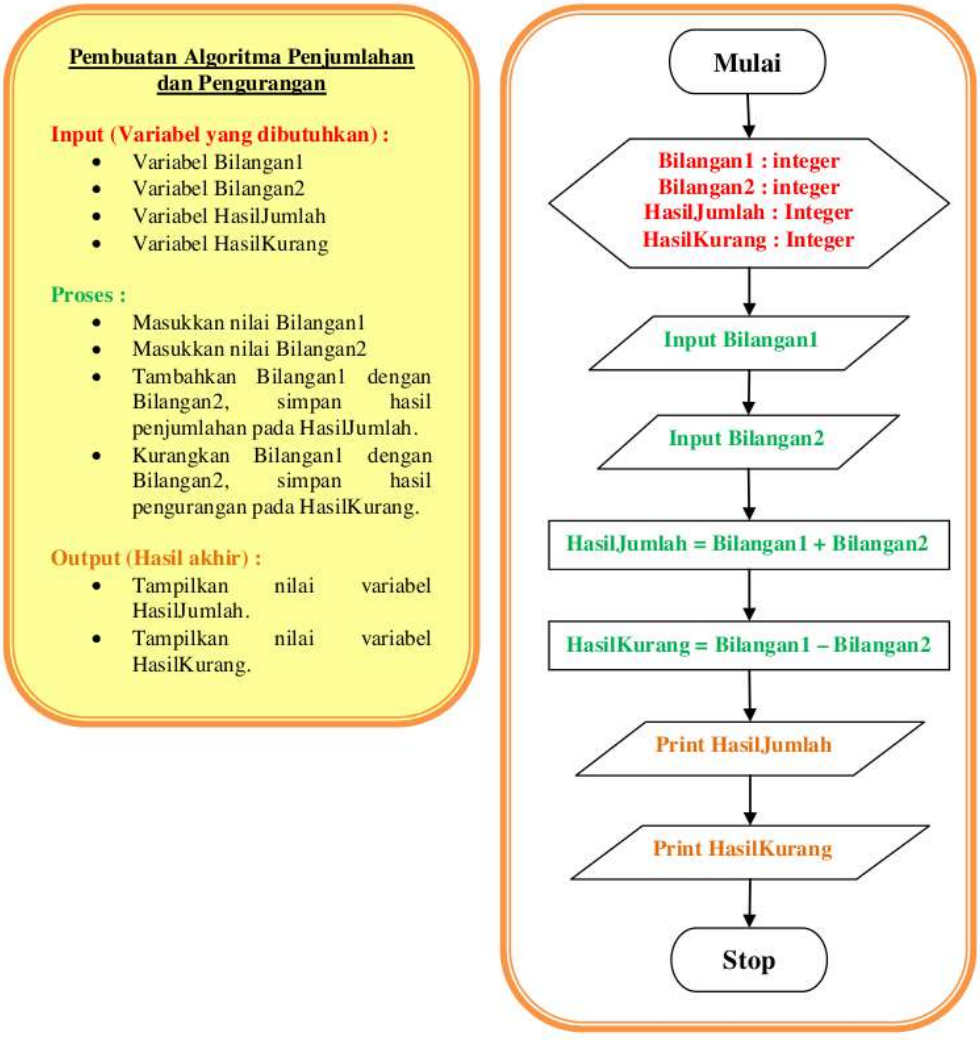
3.1.2 Macam-macam Alur Proses

Adapun pembagian alur proses dalam membentuk sebuah algoritma yang dibagi menjadi 3 proses besar, yaitu sebagai berikut :

1. **Proses Sekuensial *(Sequence Process)*.**

   Pada proses sekuensial, semua perintah/instruksi dijalankan secara berurutan, satu persatu, mulai dari yang pertama sampai yang terakhir.

Contoh : Penjumlahan dan Pengurangan 2 buah bilangan.



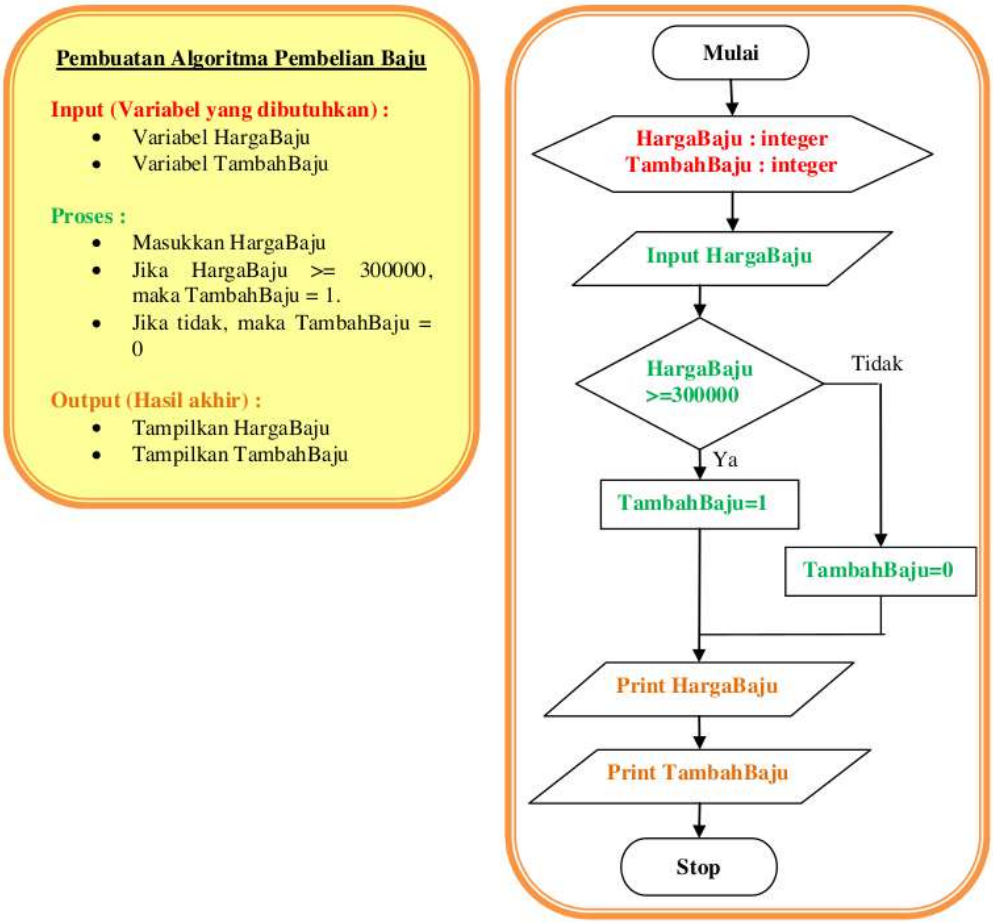
Gambar 3.1. Flowchart Penjumlahan dan Pengurangan 2 buah Bilangan.

Dari contoh studi kasus diatas, langkah-langkah penjumlahan dan pengurangan 2 buah bilangan harus dilakukan secara berurutan, mulai dari awal sampai akhir (stop). Tidak boleh terbalik ataupun terlewatkan satu langkah sekalipun. Apabila hal tersebut terjadi, maka hasil dari proses algoritma tidak dapat menghasilkan output yang benar.

2. Proses Percabangan *(Selection/Decision Process).*

Pada proses percabangan, sebuah perintah/instruksi dapat dijalankan dengan syarat/kondisi tertentu harus dipenuhi terlebih dahulu. Apabila syarat/kondisi belum terpenuhi, maka perintah/instruksi tersebut tidak akan pernah dijalankan. Syarat atau kondisi yang digunakan boleh lebih dari satu kondisi (tidak terbatas).

Contoh : Pembelian baju di sebuah mall. Jika harga baju diatas Rp 300.000, maka pembeli akan mendapatkan 1 potong baju dengan jenis yang sama. Namun, apabila harga baju tidak melebihi harga Rp 300.000, maka pembeli tidak mendapatkan baju gratis.



Gambar 3.2. Flowchart Pembelian Baju.

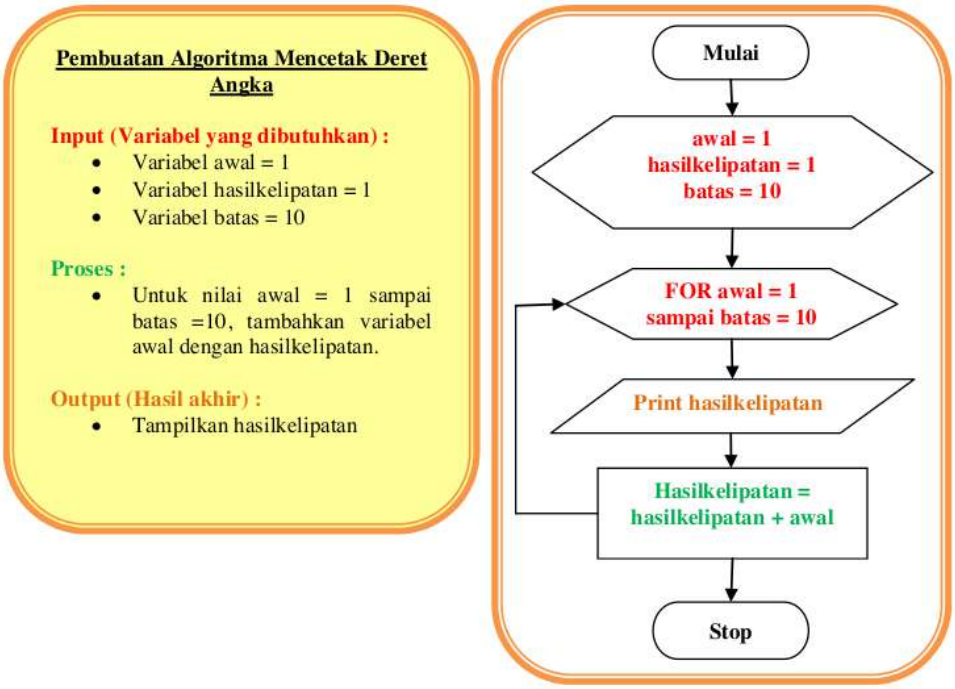
Dari contoh studi kasus diatas, syarat atau kondisi yang harus dipenuhi adalah pembelian dengan nominal diatas Rp 300.000.

3. Proses Perulangan (Iteration/Looping Process).

Pada proses perulangan, sebuah perintah/instruksi dapat dijalankan secara berulang sampai kondisi tertentu. Kondisi pada proses perulangan ini biasanya digunakan sebagai batas awal dan batas akhir perulangan. Apabila tidak ada kondisi sebagai batas perulangan, maka proses pada algoritma tersebut akan berjalan terus-menerus, sehingga mengakibatkan adanya *out of space memory*.

Contoh : Mencetak deret angka mulai dari angka 1 sampai 10, dengan kelipatan 1.

Dari contoh diatas, kondisinya adalah berupa batas awal dengan nilai = 1 dan batas akhir dengan nilai = 10. Proses perulangan akan menambahkan angka 1 sebagai kelipatannya. Sehingga pada akhir program, dapat menghasil output angka 1,2,3,4,5,6,7,8,9,10.



Gambar 3.3. Flowchart Mencetak Deret Angka.

3.2 Pseudocode

Selain flowchart, algoritma juga dapat disajikan dalam bentuk struktur english dan pseudocode. Struktur English merupakan alat yang cukup efisien untuk menggambarkan suatu algoritma. Basis dari struktur inggris english adalah bahasa inggris, tetapi juga bisa

digunakan bahasa Indonesia. Sedangkan pseudocode berarti kode yang mirip dengan kode pemrograman sebenarnya. Namun, dalam buku ajar ini pseudocode akan lebih banyak ditulis menggunakan struktur bahasa C/ Java.

3.2.1 Definisi Pseudocode

Pseudocode didefinisikan sebagai bentuk penyajian algoritma dengan menggunakan struktur bahasa tertentu. Pseudocode berasal dari kata *pseudo* yang berarti imitasi/mirip/menyerupai. Sedangkan *code* berarti kode program. Dari asal kata tersebut, dapat diartikan kode yang mirip dengan kode pemrograman sebenarnya. Kode program yang paling sering digunakan dalam penulisan pseudocode adalah struktur bahasa pascal dan C.

Contoh algoritma dalam struktur Bahasa Indonesia :

```
Baca data nilai_tugas dan nilai_ujian
Hitung total adalah nilai_tugas ditambah nilai_ujian
Tampilkan total
```

Pseudocode dengan Pascal :

```
Read (nilai_tugas, nilai_ujian)
total := nilai_tugas + nilai_ujian
Write total
```

Pseudocode dengan C/Java :

```
Input (nilai_tugas, nilai_ujian)
total = nilai_tugas + nilai_ujian
Print total
```

3.2.2 Notasi Pseudocode

Seperti halnya bahasa pemrograman yang mempunyai struktur kode program yang berbeda-beda, terdapat juga beberapa notasi dalam penulisan teks menggunakan Pseudocode. Aturan Notasi Penulisan Pseudocode adalah sebagai berikut :

1. **Memberikan nilai awal**

   Notasi : ← atau =

   Contoh :

   - Variabel namaMhs diisi dengan nilai awal “Endra”.

     namaMhs = “Endra”
   - Variabel bilangan diisi dengan nilai awal 100.

     Bilangan ← 100

2. **Membaca inputan user**

   Notasi : Read atau Input

   Contoh :

   - Variabel nim dan namaMhs diisi dengan meminta inputan dari user.

     Read (nim, namaMhs)

   - Variabel bilangan diisi dengan meminta inputan dari user

     Input (bilangan)

3. **Mencetak hasil output**

   Notasi : Write atau Print

   Contoh :

   - Mencetak isi dari variabel nim dan namaMhs

     Write (nim, namaMhs)

   - Mencetak isi dari variabel bilangan.

     Print (bilangan)

4. **Menghitung operasi aritmatika**

   Notasi : ← atau =

   Contoh : Menghitung penjumlahan antara isi variabel bil1 dan bil2 yang disimpan pada variabel hasilJumlah.

   hasilJumlah ← bil1 + bil2 atau **hasilJumlah = bil1 + bil2**

5. **Melakukan proses penyeleksian kondisi**

   Notasi : If (kondisi) then

   ```
   Pernyataan jika kondisi bernilai benar
   Else
   Pernyataan jika kondisi bernilai salah
   End If
   ```

   Contoh : Pembelian baju diatas Rp 300.000, akan mendapat tambahan baju 1 potong.

   ```
   If (HargaBaju >= 300000) then
   TambahBaju = 1;
   Else
   TambahBaju = 0;
   End If
   ```

6. **Melakukan proses perulangan**

Notasi :
```
For (variabel_ulang = nilai_awal To nilai_akhir) do
        Pernyataan yang akan diulang
        Increment/Decrement (Penaikan/Penurunan)
    End For
```

Contoh : Mencetak kata "STIKOM Surabaya" sebanyak 10 kali

```
For (awal = 1 To 10) do
        Write ("STIKOM Surabaya")
        awal = awal+1;
End For
```

Hal yang paling mudah dalam penulisan pseudocode adalah sama dengan apa yang dituliskan di dalam simbol flowchart. Namun, apabila ingin mengikuti aturan penulisan pseudocode akan lebih baik. Hal ini dikarenakan, apabila pseudocode telah ditulis sesuai dengan notasinya, maka lebih memudahkan programmer atau orang lain untuk memahami proses yang terjadi di dalamnya.

Struktur penulisan pseudocode pada umunya terdiri dari 3 bagian, yaitu bagian judul, bagian deklarasi, dan bagian deskripsi. Adapun Struktur penulisan pseudocode dapat dilihat pada Gambar 3.4. Contoh penulisan pseudocode dengan notasi algoritmik pada proses sekuensial untuk proses penjumlahan dan pengurangan 2 buah bilangan, dapat dilihat pada Gambar 3.5.

Judul : ***(Judul_Algoritma)***

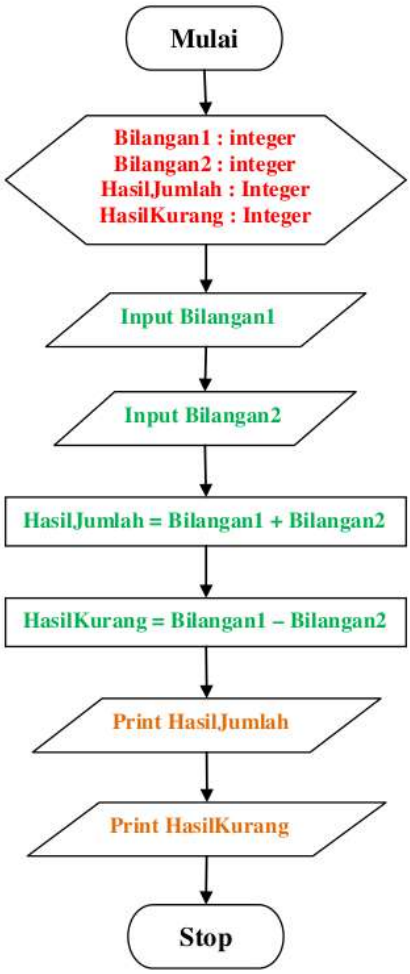
Deklarasi :

(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :

(Tulis setiap tahapan/proses yang dilakukan di dalam algoritma, mulai dari awal sampai akhir. Sesuaikan penulisan deskripsi dengan apa yang ditulis di dalam symbol flowchart atau sesuaikan dengan penulisan notasi algoritmik)

Gambar 3.4. Struktur Penulisan Pseudocode.



Judul : Algoritma Penjumlahan dan Pengurangan 2 buah bilangan.

Deklarasi :
Bilangan1 : integer;
Bilangan2 : integer;
HasilJumlah : integer;
HasilKurang : integer;

Deskripsi :
Input (Bilangan1)
Input (Bilangan2)
HasilJumlah = Bilangan1 + Bilangan2
HasilKurang = Bilangan1 – Bilangan2
Print (HasilJumlah)
Print (HasilKurang)

ATAU

Judul : Algoritma Penjumlahan dan Pengurangan 2 buah bilangan.

Deklarasi :
Bilangan1 : integer;
Bilangan2 : integer;
HasilJumlah : integer;
HasilKurang : integer;

Deskripsi :
Input (Bilangan1, Bilangan2)
HasilJumlah ← Bilangan1 + Bilangan2
HasilKurang ← Bilangan1 – Bilangan2
Print (HasilJumlah, HasilKurang)

Gambar 3.5. Penulisan Pseudocode dengan Notasi Algoritmik pada Proses Sekuensial untuk Proses Penjumlahan dan Pengurangan 2 buah bilangan.

Notes :

a. Untuk memudahkan Anda membuat pseudocode, buat flowchartnya terlebih dahulu.
b. Dari flowchart yang telah dibuat, dapat langsung diubah ke dalam struktur penulisan pseudocode dan sesuaikan dengan notasi algoritmik.
c. Anda dapat melakukan hal yang sama dalam pembuatan pseudocode untuk proses percabangan dan perulangan.

3.3 Rangkuman

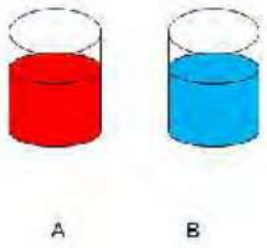
- Flowchart didefinisikan sebagai suatu bagan terurut untuk menggambarkan alur yang terjadi pada suatu proses, dengan menggunakan simbol – simbol tertentu.
- Flowchart pada algoritma dibagi menjadi 2 bentuk, yaitu :
  - Flowchart Program
  - Flowchart Sistem
- Simbol-simbol yang digunakan pada flowchart sistem ada yang berbeda dengan simbol-simbol yang digunakan pada flowchart program.
- Pembagian alur proses dalam membentuk sebuah algoritma yang dibagi menjadi 3 proses besar, yaitu :
  - Proses Sekuensial (Sequnce Process)
  - Proses Percabangan (Selection/Decision Process)
  - Proses Perulangan (Iteration/Looping Process)
- Di samping Flowchart, algoritma juga dapat disajikan dalam bentuk struktur english dan pseudocode.
- Pseudocode berasal dari kata *pseudo* yang berarti imitasi/mirip/menyerupai. Sedangkan *code* berarti kode program. Pseudocode berarti kode yang mirip dengan kode pemrograman sebenarnya.

3.4 Latihan Soal

Buatlah Flowchart dan Pseudocode untuk permasalahan berikut ini :

1. Titi, seorang siswa yang masih duduk di bangku Sekolah Dasar sedang belajar Matematika dengan kakaknya, Mas Didi, untuk menghadapi ujian tengah semester. Dengan senang hati Mas Didi memberikan latihan soal kepada Titi. Latihan soal yang diberikan adalah bagaimana melakukan proses perhitungan berupa penjumlahan, pengurangan, perkalian dan pembagian dari 3 buah bilangan yang nilainya berbeda.

2. Menukar isi dua buah gelas



30

Diberikan dua buah bejana A dan B,
bejana A berisi larutan berwarna merah,
bejana B berisi larutan berwarna biru.
Pertukarkan isi kedua bejana itu sedemikian sehingga bejana A berisi larutan berwarna biru dan bejana B berisi larutan berwarna merah.

3. Seorang ibu menyuruh anaknya bernama Titi untuk membeli 2 kilogram beras dan 3 liter minyak goreng di Toko Serbaguna Didi. Setiap kilogram beras dijual dengan harga Rp 9.000, sedangkan setiap liter minyak goreng dijual dengan harga Rp 11.000. Berapa jumlah uang kembalian yang Titi terima, apabila ibu memberikan uang sebesar Rp 200.000 untuk belanja ?

Bab 4

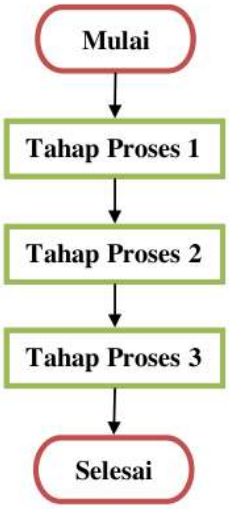
Proses Sekuensial

Tujuan :

- ✓ Mahasiswa dapat menjelaskan alur proses sekuensial.
- ✓ Mahasiswa dapat mendemonstrasikan pembuatan struktur proses sekuensial.

4.1 Flowchart Proses Sekuensial

Dalam sebuah algoritma, tahapan penyelesaian sebuah permasalahan dapat berupa struktur berurutan *(sequence)*, struktur pemilihan atau pengambilan keputusan *(selection)*, dan struktur pengulangan *(iteration)*. Ketiga jenis langkah tersebut membentuk struktur konstruktif suatu algoritma. Proses sekuensial adalah sebuah proses program dimana setiap baris program akan dikerjakan secara urut dari atas ke bawah, mulai dari yang pertama sampai yang terakhir, sesuai dengan urutan penulisannya.



Gambar 4.1. Gambar Flowchart Proses Sekuensial.

Berdasarkan gambar 4.1, flowchart berjalan pada awal tahapan mulai, kemudian berjalan pada tahap Proses 1. Jika tahap proses 1 telah selesai, maka dilanjutkan dengan menjalankan tahap proses 2. Tahap proses 2 tidak akan dijalankan, apabila tahap proses 1 belum selesai dikerjakan. Setelah tahap proses 2 selesai dikerjakan, flowchart akan melangkah ke tahap proses 3. Jika tahap proses 3 selesai, maka flowchart akan berhenti melakukan proses, artinya sudah selesai, tidak ada tahap proses yang harus dikerjakan lagi. Tahap proses 1, tahap proses 2, dan tahap proses 3 harus dikerjakan secara urut. Tidak boleh meloncati 1 tahap ataupun ada tahap proses yang dibalik. Jika hal tersebut terjadi, maka hasil output tidak akan sesuai dengan yang diinginkan, bahkan bisa jadi tidak tepat.

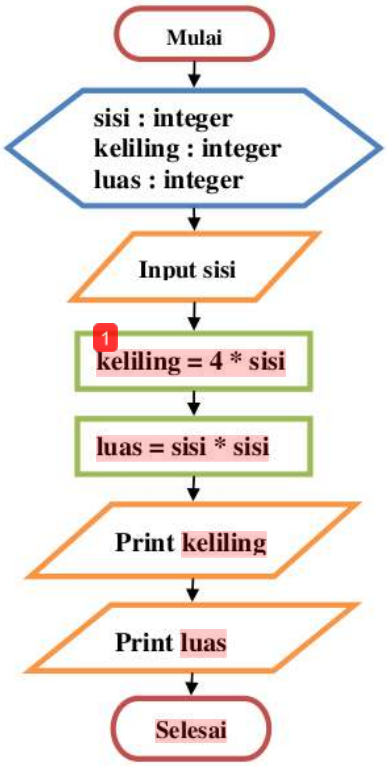
Contoh 1 : Flowchart menghitung Keliling dan Luas Persegi

Input : panjang sisi

Rumus Keliling persegi = 4 x sisi

Rumus Luas Persegi = sisi x sisi

Output : Hasil Keliling dan Luas Persegi



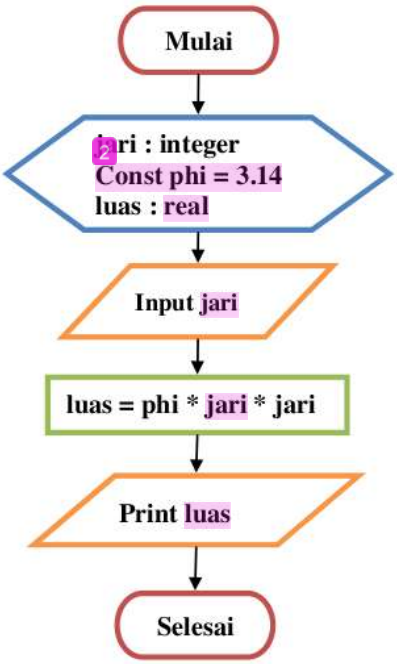
Gambar 4.2. Flowchart Menghitung Keliling dan Luas Persegi.

Contoh 2 : Flowchart menghitung Luas Lingkaran

Input : panjang jari-jari

Rumus Luas Lingkaran = phi x jari-jari x jari-jari

Output : Hasil Luas Lingkaran



Gambar 4.3. Flowchart Menghitung Luas Lingkaran.

Contoh 3 : Flowchart menghitung Total Pembelian

Menghitung Total Pembelian yang harus dibayar untuk sejumlah barang sembako di Toko Sembako Didi. Daftar yang dibeli adalah sebagai berikut :

- 10 kilogram beras @ Rp 10.000
- 2 liter minyak goreng @ Rp 15.000
- 5 buah sabun @ Rp 3.000

Variabel : jberas, jminyak, jsabun, hberas, hminyak, hsabun, tberas, tminyak, tsabun, total

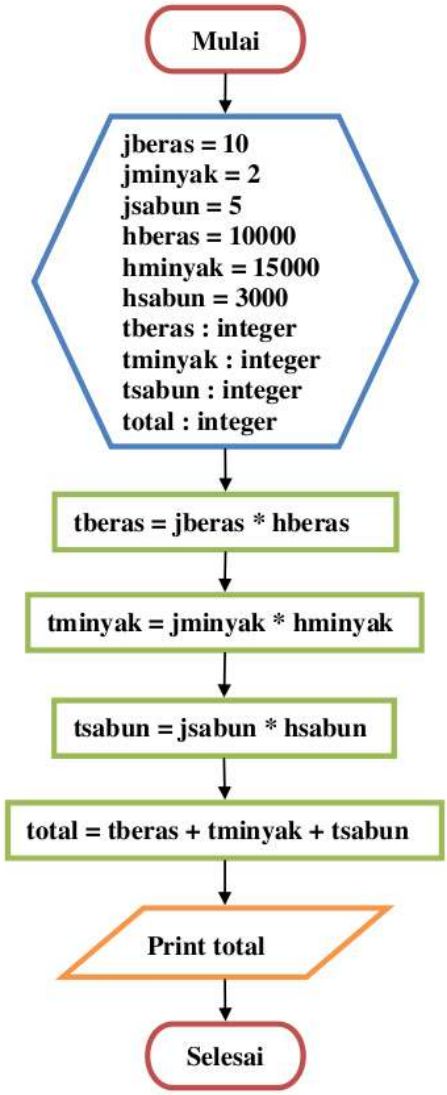
tberas = jberas x hberas

tminyak = jminyak x hminyak

tsabun = jsabun x hsabun

Hitung Total Pembelian = tberas + tminyak + tsabun

Output : Hasil Total Pembelian



Gambar 4.4. Flowchart Menghitung Total Pembelian

4.2 Pseudocode Proses Sekuensial

Hal yang paling mudah dalam penulisan pseudocode untuk proses sekuensial adalah sama dengan apa yang dituliskan di dalam simbol flowchart. Namun, apabila ingin mengikuti aturan penulisan pseudocode akan lebih baik. Hal ini dikarenakan, apabila pseudocode telah ditulis sesuai dengan notasinya, maka lebih memudahkan programmer atau orang lain untuk memahami proses yang terjadi di dalamnya. Struktur penulisan pseudocode untuk Proses Sekuensial dapat dilihat pada Gambar 4.5.

Judul : ***(Judul_Algoritma)***

Deklarasi :

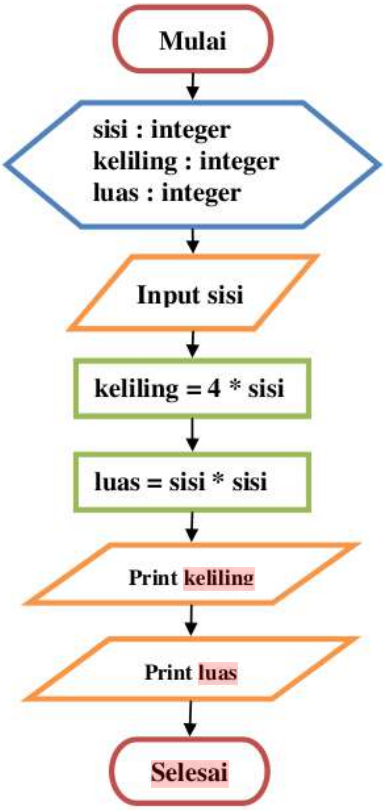
(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :

Tahap Proses 1
Tahap Proses 2
Tahap Proses 3

Gambar 4.5. Struktur Penulisan Pseudocode untuk Proses Sekuensial.

Contoh 1 : Pseudocode menghitung Keliling dan Luas Persegi



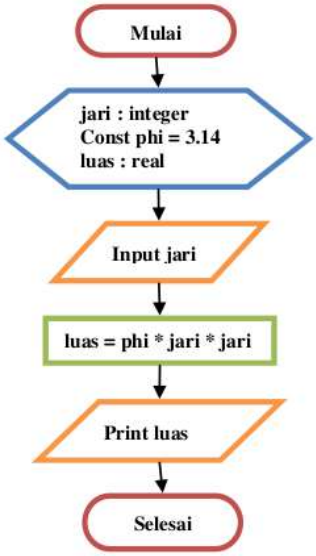
Judul : Algoritma Menghitung Keliling dan Luas Persegi.

Deklarasi :
sisi : integer;
keliling : integer;
luas : integer;

Deskripsi :
Input (sisi)
keliling ← 4 * sisi
luas ← sisi * sisi
Print (keliling, luas)

Gambar 4.6. Pseudocode menghitung Keliling dan Luas Persegi.

Contoh 2 : Pseudocode menghitung Luas Lingkaran



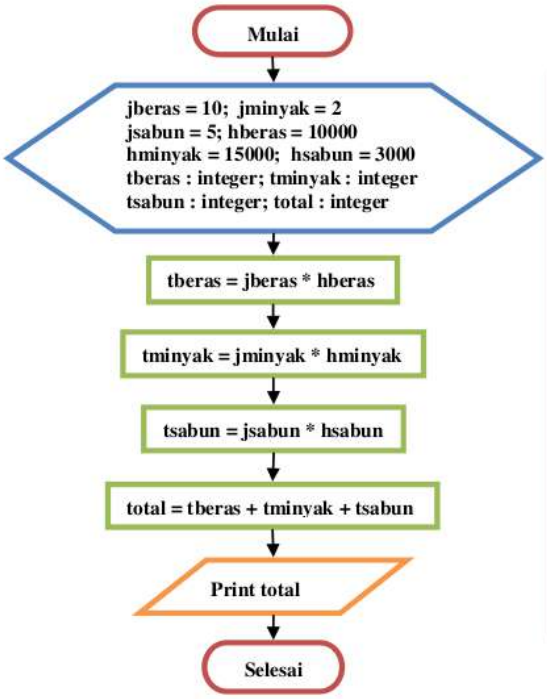
Judul : Algoritma Menghitung Luas Lingkaran.

Deklarasi :
jari : integer;
Const phi = 3.14;
luas : real;

Deskripsi :
Input (jari)
luas ← phi * jari * jari
Print (luas)

Gambar 4.7. Pseudocode menghitung Luas Lingkaran.

Contoh 3 : Pseudocode menghitung Total Pembelian



Judul : Algoritma Menghitung Total Pembelian.

Deklarasi :
jberas = integer;
jminyak = integer;
jsabun = integer;
hberas = integer;
hminyak = integer;
hsabun = integer;
tberas : integer;
tminyak : integer;
tsabun : integer;
total : integer;

Deskripsi :
jberas ← 10
jminyak ← 2
jsabun ← 5
hberas ← 10000
hminyak ← 15000
hsabun ← 3000
tberas ← jberas * hberas
tminyak ← jminyak * hminyak
tsabun ← jsabun * hsabun
total ← tberas + tminyak + tsabun
print (total)

Gambar 4.8. Pseudocode menghitung Total Pembelian.

4.3 Rangkuman

- Proses sekuensial adalah sebuah proses program dimana setiap baris program akan dikerjakan secara urut dari atas ke bawah, mulai dari yang pertama sampai yang terakhir, sesuai dengan urutan penulisannya.
- Pembuatan Flowchart pada Proses Sekuensial dilakukan secara urut, yaitu :
  - Tahap Proses 1
  - Tahap Proses 2
  - Tahap Proses 3
  - Tahap Proses ke-N
- Struktur penulisan pseudocode untuk Proses Sekuensial, yaitu terdiri dari :
  - Judul
  - Deklarasi
  - Deskripsi
    - Tahap Proses 1
    - Tahap Proses 2
    - Tahap Proses 3

4.4 Latihan Soal

Buatlah Flowchart dan Pseudocode untuk permasalahan Proses Sekuensial berikut ini :

1. Pada saat menjelang kenaikan kelas, Ibu Titi sibuk menghitung nilai para siswa untuk dapat ditulis pada raport sebagai laporan nilai akhir siswa. Cara perhitungan nilai akhir pun memiliki rumus tersendiri, yaitu sebagai berikut :

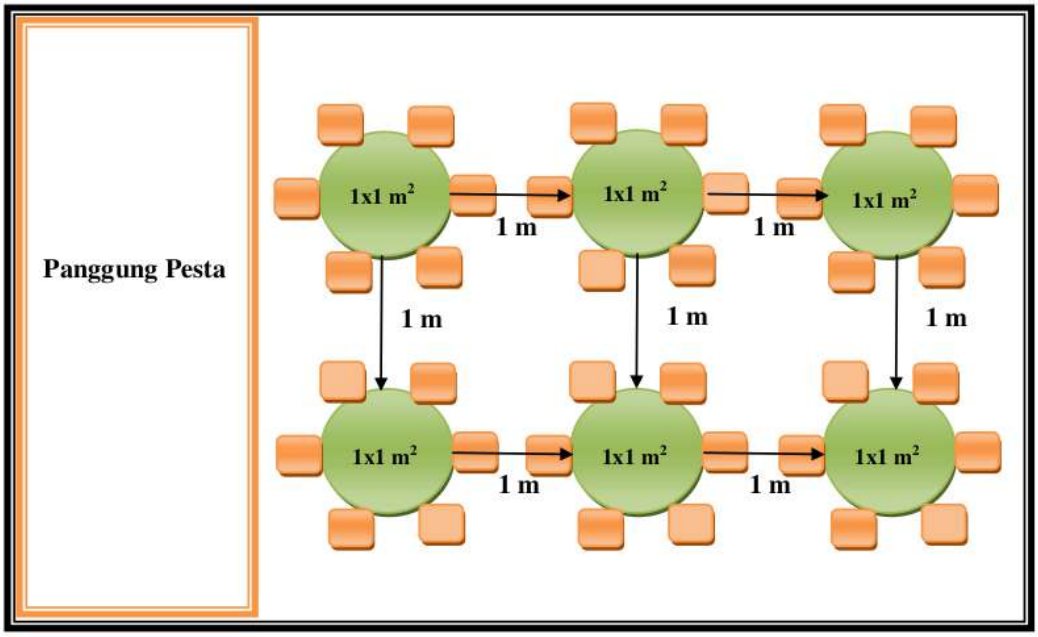
   Nilai Akhir = Nilai Tugas * 0.4 + Nilai Ujian * 0.6

   Untuk memudahkan Ibu Titi dalam melakukan proses perhitungan, buatlah flowchart dan pseudocode yang meminta inputan nilai tugas dan nilai ujian, kemudian cetaklah nilai akhir yang akan dilaporkan pada raport siswa.

2. Dalam rangka merayakan ulang tahun Titi yang ke-17, Titi ingin mengundang seluruh teman-temannya untuk menghadiri pesta ulang tahun yang diadakan di rumahnya. Pesta tersebut rencananya dilakukan di halaman depan yang berukuran 5x10 m2. Titi meminta bantuan kepada Mas Didi untuk memperkirakan berapa banyak teman yang dapat diundang dengan memperhatikan beberapa hal sebagai berikut :

a. Halaman depan untuk pesta berukuran 5x10 m^2.
b. Halaman sekitar 5x2 m2 akan digunakan untuk panggung pesta.
c. Meja bundar diletakkan berjajar pada halaman yang tersisa dengan ukuran meja 1x1 m^2. Beri jarak 1m antara meja yang satu dengan yang lainnya.
d. Setiap meja bundar akan dikelilingi oleh 6 kursi tamu.

Untuk memudahkan tugas Mas Didi, bantulah membuat flowchart dan pseudocode yang dapat menghitung berapa banyak meja dan kursi yang dibutuhkan untuk pesta ulang tahun Titi. Sertakan juga berapa banyak tamu yang akan diundang dengan memperhatikan jumlah kursi tamu.



Bab 5

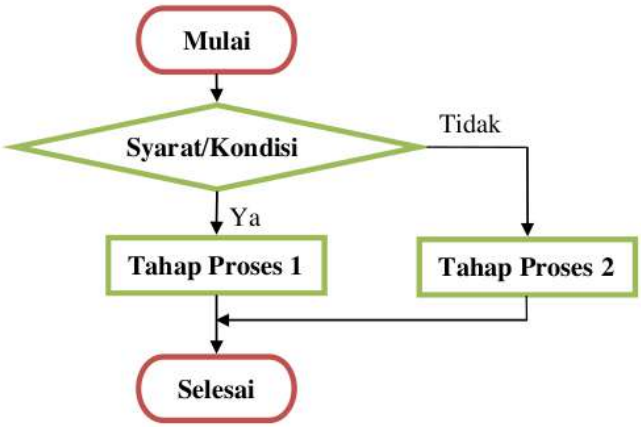
Percabangan (Selection/Decision)

Tujuan :

- ✓ Mahasiswa dapat menjelaskan alur proses percabangan.
- ✓ Mahasiswa dapat mendemonstrasikan pembuatan struktur proses percabangan.

5.1 Proses Percabangan

Dalam sebuah algoritma, tahapan penyelesaian sebuah permasalahan dapat berupa struktur berurutan *(sequence)*, struktur pemilihan atau pengambilan keputusan *(selection)*, dan struktur pengulangan *(iteration)*. Ketiga jenis langkah tersebut membentuk struktur konstruktif suatu algoritma. Proses percabangan adalah sebuah proses program dimana terdapat sebuah perintah/instruksi yang dapat dijalankan dengan syarat/kondisi tertentu harus dipenuhi terlebih dahulu. Apabila syarat/kondisi belum terpenuhi, maka perintah/instruksi tersebut tidak akan pernah dijalankan. Syarat atau kondisi yang digunakan boleh lebih dari satu kondisi (tidak terbatas). Proses percabangan juga dikenal sebagai proses pemilihan atau proses pengambilan keputusan.



Gambar 5.1. Gambar Flowchart Proses Percabangan.

Berdasarkan gambar 5.1, flowchart berjalan pada awal tahapan mulai, kemudian berjalan pada proses syarat/kondisi yang harus dipenuhi terlebih dahulu. Jika syarat/kondisi dapat dipenuhi, maka tahap proses 1 akan dijalankan. Namun sebaliknya, apabila syarat/kondisi tidak dapat dipenuhi, maka tahap proses 2 yang akan dijalankan, bukan tahap proses 1. Hal tersebut ditandai dengan kata "Ya" dan "Tidak" pada cabang panah.

Proses percabangan ini dapat terjadi pada 3 bagian flowchart, yaitu pada bagian awal, tengah, maupun akhir flowchart. Sesuaikan dengan kondisi permasalahan yang ingin diselesaikan. Apabila proses percabangan diletakkan pada bagian awal, pada tahap bagian tengah dan akhir dapat diisi dengan proses sekuensial lainnya yang merupakan proses kelanjutan dari proses percabangan. Sehingga tidak menutup kemungkinan, dalam satu flowchart terdapat proses sekuensial dan proses percabangan, hanya tergantung pada peletakan urutan prosesnya saja.

Contoh 1 : Flowchart menentukan kepemilikan SIM

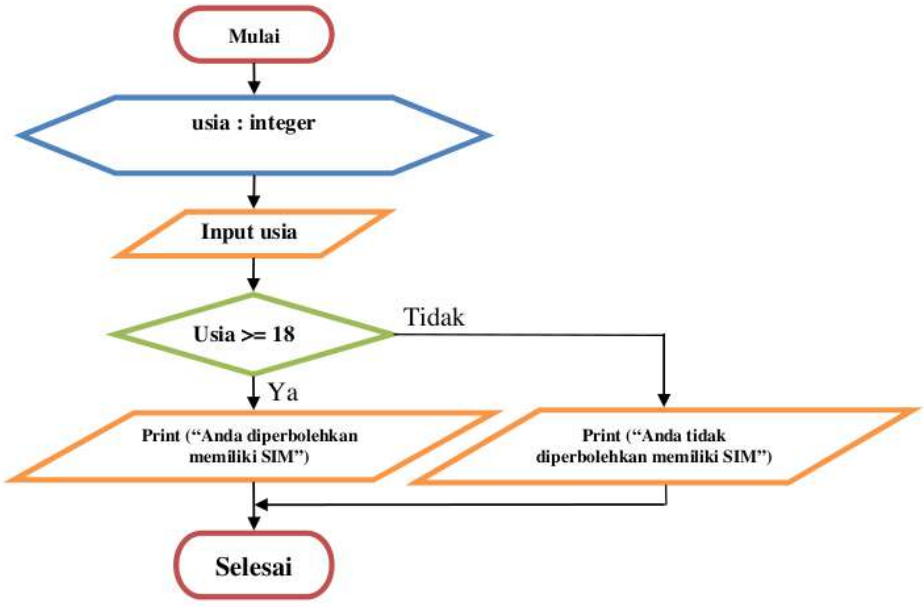
Input : usia

Syarat/Kondisi :

Jika usia diatas 18 tahun, maka dapat memiliki SIM.

Jika belum berusia 18 tahun, maka tidak dapat memiliki SIM.

Output : Kepemilikan SIM



Gambar 5.2. Flowchart Menentukan Kepemilikan SIM.

Contoh 2 : Flowchart Menghitung Total Belanja dengan Diskon

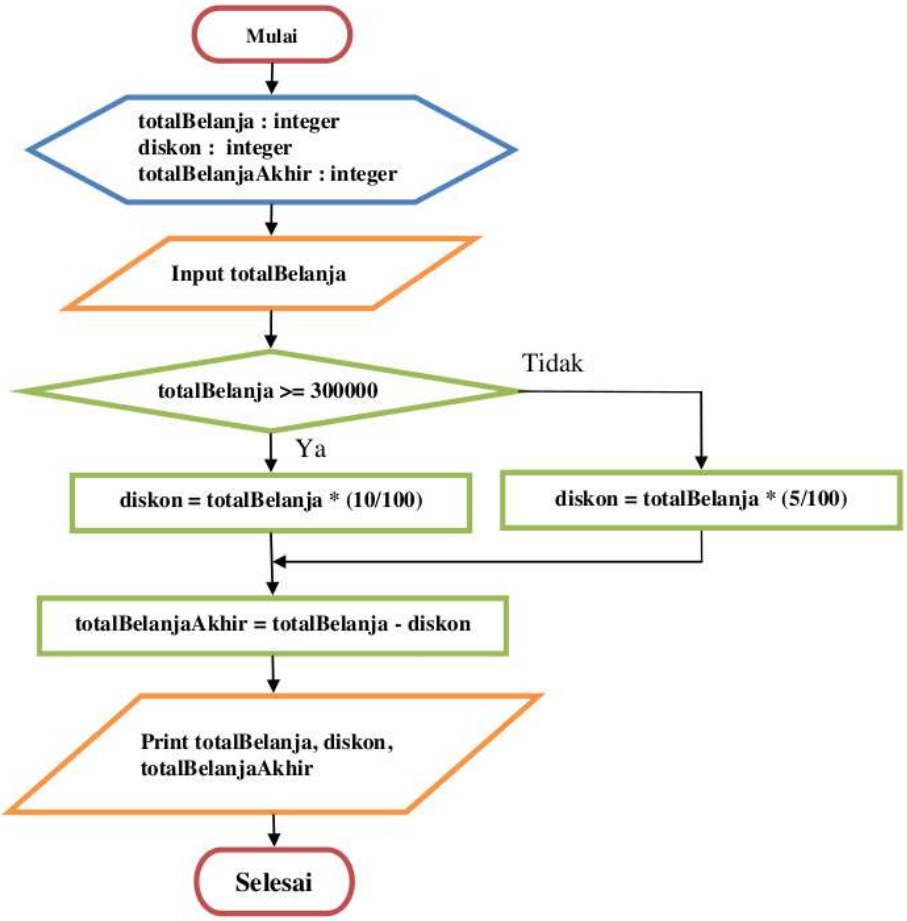
Input : totalBelanja

Syarat/Kondisi :

Jika totalBelanja diatas Rp 300.000, maka mendapatkan diskon 10%.

Jika totalBelanja dibawah Rp 300.000, maka mendapatkan diskon 5%.

Output : Total Belanja setelah diskon



Gambar 5.3. Flowchart Menghitung Total Belanja dengan Diskon.

5.2 Bentuk Proses Percabangan

Terdapat beberapa jenis bentuk proses percabangan yang penting untuk dipelajari sebagai konsep dasar percabangan. Bentuk Flowchart dan Pseuducode yang telah dipelajari pada sub bab sebelumnya merupakan bentuk umum dari proses percabangan. Adapun bentuk proses percabangan dibagi menjadi 4 bentuk, yaitu sebagai berikut :

1. **Bentuk If Sederhana (IF…THEN)**

   Bentuk If Sederhana ini merupakan bentuk dasar yang paling sederhana dibandingkan bentuk if yang lainnya. Bentuk ini dikatakan paling sederhana karena hanya memiliki satu syarat/kondisi dan juga hanya mempunyai satu tahap proses jika kondisi terpenuhi. Namun, apabila syarat/kondisi tidak terpenuhi, maka flowchart atau program tidak akan melakukan aksi apapun. Dalam Bahasa Indonesia, bentuk ini dikenal dengan sebutan JIKA…MAKA.

2. **Bentuk IF…THEN…ELSE**

   Bentuk If…Then…Else merupakan bentuk kedua dari if. Berbeda dengan bentuk pertama If Sederhana, bentuk kedua dari if ini memiliki satu syarat/kondisi dan dua tahap proses. Tahap proses 1 akan dijalankan apabila syarat/kondisi terpenuhi. Jika tidak terpenuhi, maka akan menjalankan tahap proses 2. Sehingga tetap ada aksi yang dijalankan meskipun syarat/kondisi tidak terpenuhi.

3. **Bentuk Nested If**

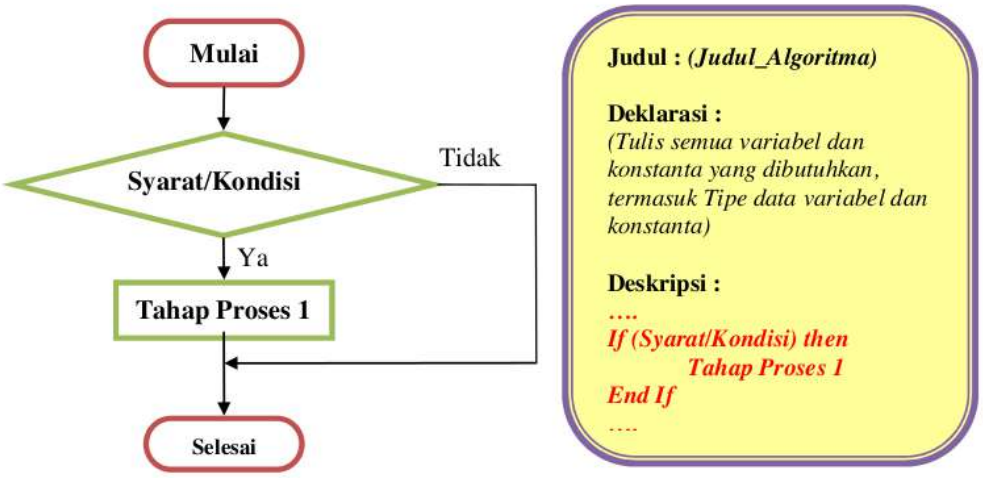
   Bentuk Nested if biasa disebut dengan If Bersarang atau If Bertingkat. Secara logika, bentuk if bersarang merupakan bentuk if di dalam if. Bentuk If bersarang mempunyai banyak syarat/kondisi yang dapat diberlakukan pada sebuah permasalahan. Dengan adanya banyak syarat/kondisi inilah menyebabkan banyaknya tahap proses yang harus dijalankan saat setiap syarat/kondisi terpenuhi.

4. **Bentuk Switch – Case**

   Bentuk Switch Case tidak jauh berbeda dengan bentuk if…then…else. Bentuk ini diciptakan sebagai pengganti if…then…else untuk syarat/kondisi yang lebih sederhana. Pada umumnya bentuk switch case ini banyak digunakan pada saat terdapat pemilihan menu. Sehingga syarat/kondisi yang digunakan hanya berdasar inputan berupa nomor atau abjad menu.

5.1.1 Bentuk IF-THEN

Bentuk If Sederhana ini merupakan bentuk dasar yang paling sederhana dibandingkan bentuk if yang lainnya. Bentuk ini dikatakan paling sederhana karena **hanya memiliki satu syarat/kondisi** dan juga **hanya mempunyai satu tahap proses** jika kondisi terpenuhi. Namun, apabila syarat/kondisi tidak terpenuhi, maka flowchart atau program tidak akan melakukan aksi apapun. Dalam Bahasa Indonesia, bentuk ini dikenal dengan sebutan JIKA…MAKA. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk If Sederhana ini dapat dilihat pada Gambar 5.4.



Gambar 5.4. Bentuk Flowchart dan Penulisan Pseudocode Bentuk If Sederhana.

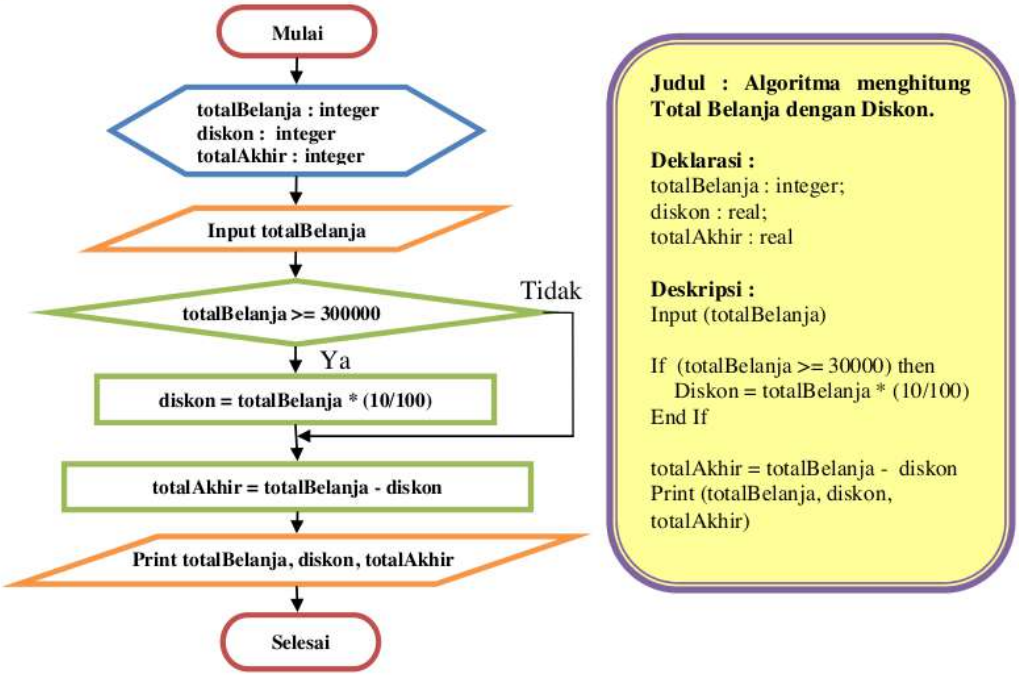
Contoh : Flowchart Menghitung Total Belanja dengan Diskon

Input : totalBelanja

Syarat/Kondisi :

Jika totalBelanja diatas Rp 300.000, maka mendapatkan diskon 10%.

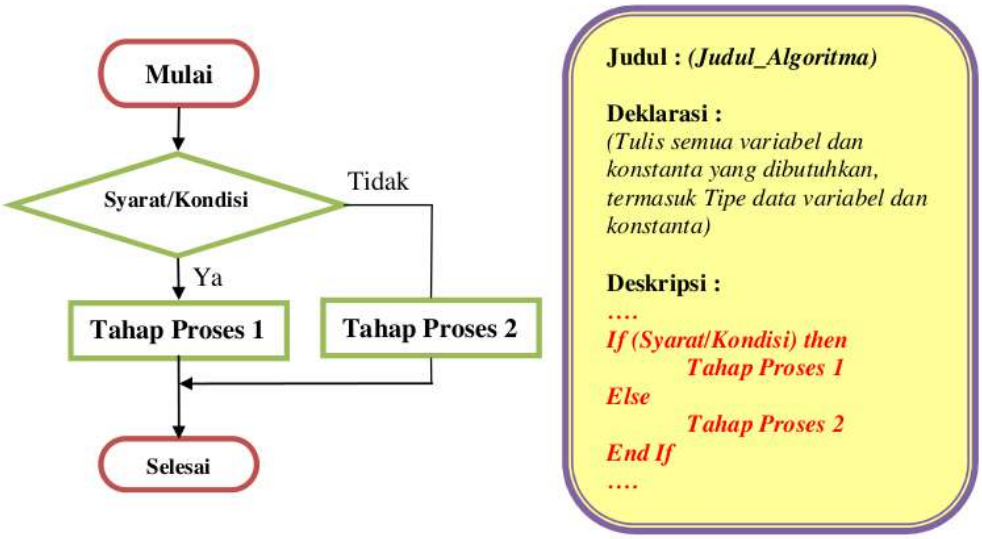
Output : Total Belanja setelah diskon



Gambar 5.5. Flowchart Menghitung Total Belanja dengan Diskon.

5.1.2 Bentuk IF-THEN-ELSE

Bentuk If…Then…Else merupakan bentuk kedua dari if. Berbeda dengan bentuk pertama If Sederhana, bentuk kedua dari if ini memiliki satu syarat/kondisi dan dua tahap proses. Tahap proses 1 akan dijalankan apabila syarat/kondisi terpenuhi. Jika tidak terpenuhi, maka akan menjalankan tahap proses 2. Sehingga tetap ada aksi yang dijalankan meskipun syarat/kondisi tidak terpenuhi. Dalam Bahasa Indonesia, bentuk ini dikenal dengan sebutan JIKA…MAKA…JIKA TIDAK. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk If Sederhana ini dapat dilihat pada Gambar 5.6.



Gambar 5.6. Bentuk Flowchart dan Penulisan Pseudocode Bentuk If-Then-Else.

Contoh : Flowchart menentukan Bilangan Ganjil dan Genap.

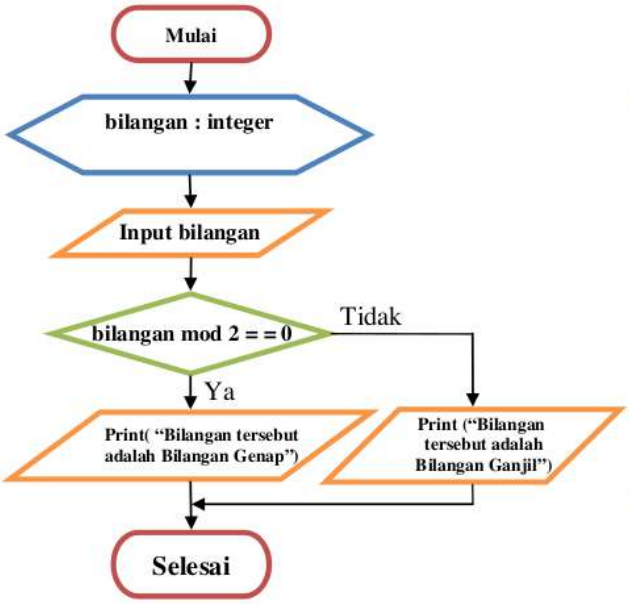
Input : bilangan

Syarat/Kondisi :

Jika bilangan tersebut habis dibagi 2, maka bilangan tersebut merupakan Bil.Genap.

Jika tidak, maka bilangan tersebut merupakan Bil.Ganjil.

Output : Status Kelompok Bilangan (Ganjil/Genap)



Judul : Algoritma Menentukan Bilangan Genap dan Ganjil.

Deklarasi :
bilangan : integer

Deskripsi :
Input (bilangan)
If (bilangan mod 2 = = 0) then
    Print(Bilangan tersebut adalah Bilangan Genap")
Else
    Print("Bilangan tersebut adalah Bilangan Genap")

End If

Gambar 5.7. Flowchart Menentukan Bilangan Ganjil dan Genap.

5.1.3 Bentuk Nested IF

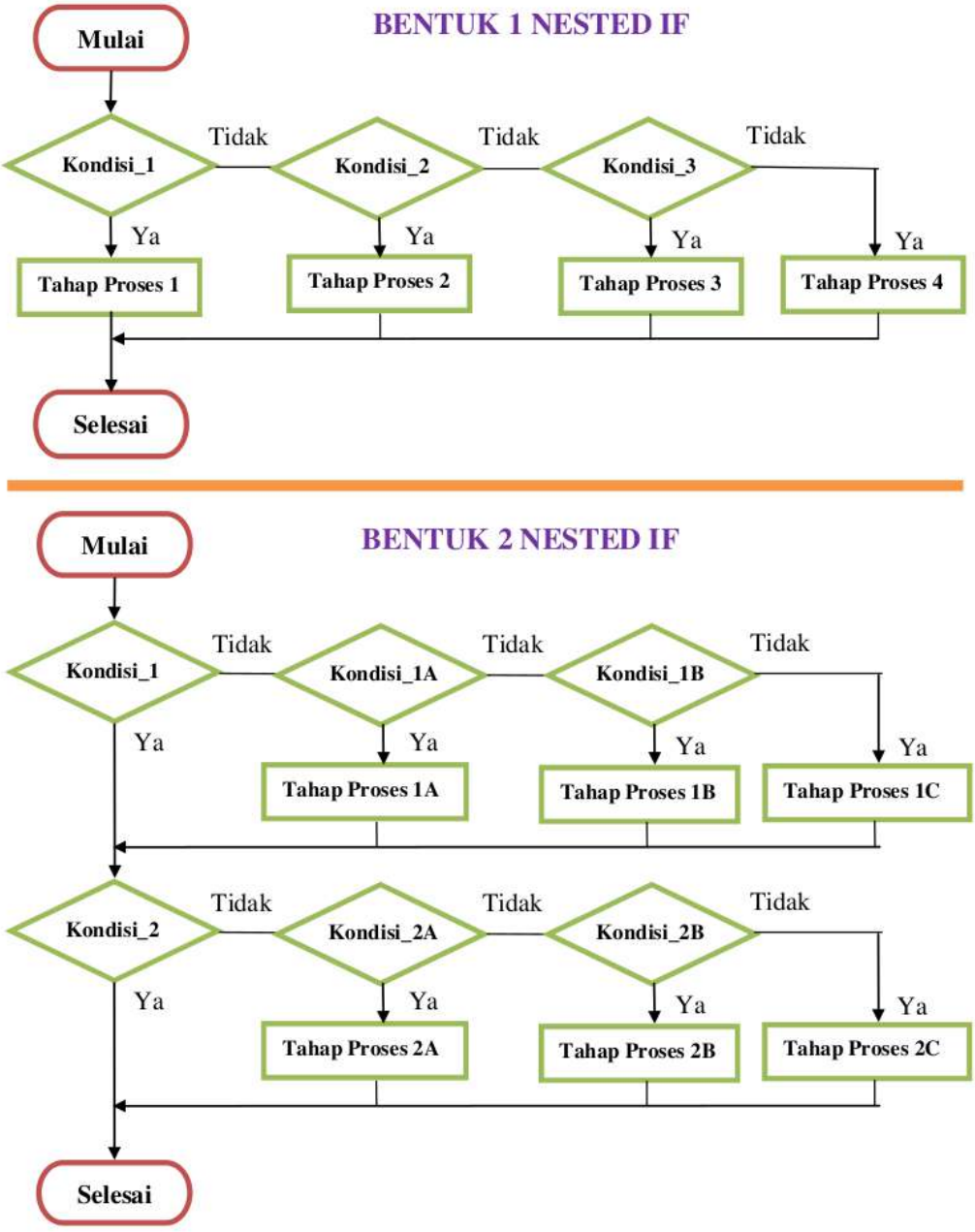
Bentuk Nested if disebut dengan If Bersarang. Secara logika, bentuk if bersarang merupakan bentuk if di dalam if. Bentuk If bersarang mempunyai banyak syarat/kondisi yang dapat diberlakukan pada sebuah permasalahan. Dengan adanya banyak syarat/kondisi inilah menyebabkan banyaknya tahap proses yang harus dijalankan saat setiap syarat/kondisi terpenuhi. Struktur Penulisan Nested If mempunyai 2 bentuk adalah sebagai berikut :

BENTUK 1 NESTED IF

```
If (Kondisi_1) then
        Tahap Proses 1
Else If (Kondisi_2) then
        Tahap Proses 2
Else If (Kondisi_3) then
        Tahap Proses 3
.....
Else If (Kondisi_N) then
        Tahap Proses N
Else
        Tahap Proses Selain
        Proses 1 sampai N
End If
```

BENTUK 2 NESTED IF

```
If (Kondisi_1) then
        If (Kondisi_1A) then
                Tahap Proses 1A
        Else If (Kondisi_1B) then
                Tahap Proses 1B
        Else
                Tahap Proses 1C
        End If
Else If (Kondisi_2) then
        If (Kondisi_2A) then
                Tahap Proses 2A
        Else If (Kondisi_2B) then
                Tahap Proses 2B
        Else
                Tahap Proses 2C
        End If
Else
        .....
End If
```



Gambar 5.8. Bentuk Flowchart Untuk Nested If.

Contoh 1 : Flowchart menentukan Nilai Akhir berupa Huruf.

Input : NilaiAkhir

Syarat/Kondisi :

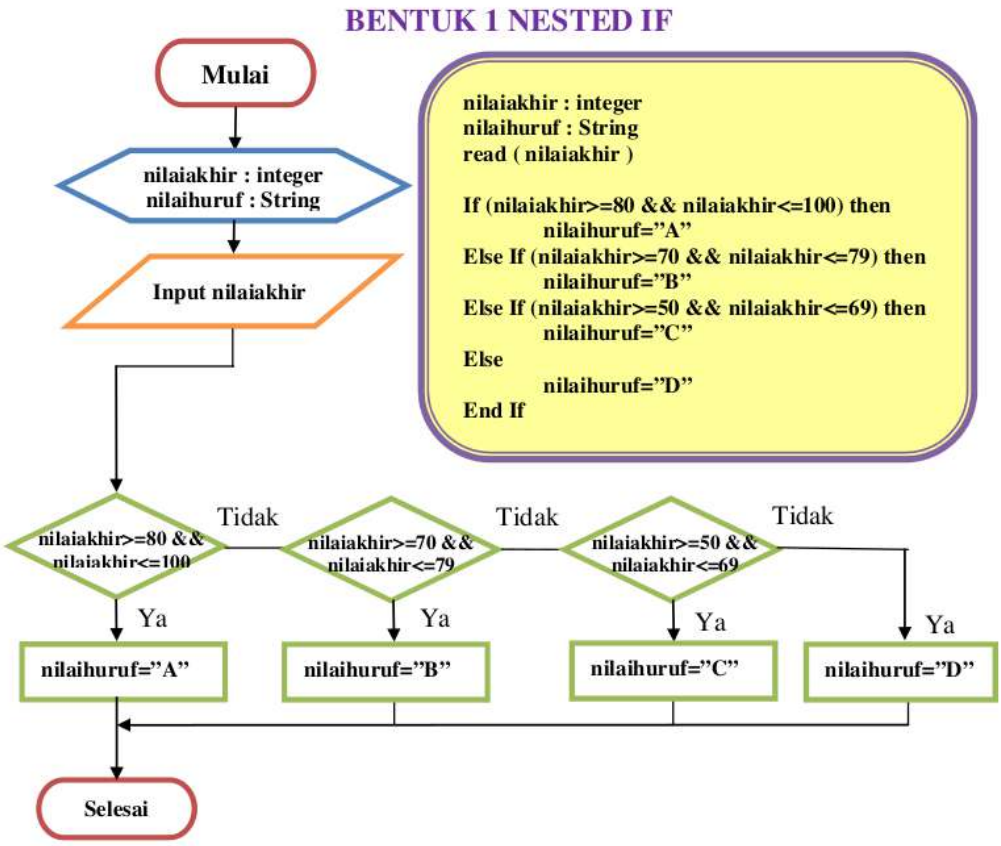
Jika Nilai Akhir antara 80-100, maka Nilai Huruf = "A"

Jika Nilai Akhir antara 70-79, maka Nilai Huruf = "B"

Jika Nilai Akhir antara 50-69, maka Nilai Huruf = "C"

Jika Nilai Akhir antara 0-49, maka Nilai Huruf = "D"

Output : Nilai Huruf



Gambar 5.9. Flowchart menentukan Nilai Akhir berupa Huruf.

Contoh 2 : Flowchart menentukan Harga Jual Buah pada setiap Kota Supplier.

Input : Kota Supplier, Jenis Buah

Syarat/Kondisi :

Jika kota supplier adalah Pacitan :

Jika jenis buah = Jeruk, maka Harga per kg = Rp 6.000

Jika jenis buah = Semangka, maka Harga per kg = Rp 7.000

Jika jenis buah lainnya, maka Harga per kg = Rp 8.000

Jika kota supplier adalah Yogyakarta :

Jika jenis buah = Jeruk, maka Harga per kg = Rp 10.000

Jika jenis buah = Semangka, maka Harga per kg = Rp 11.000

Jika jenis buah lainnya, maka Harga per kg = Rp 12.000

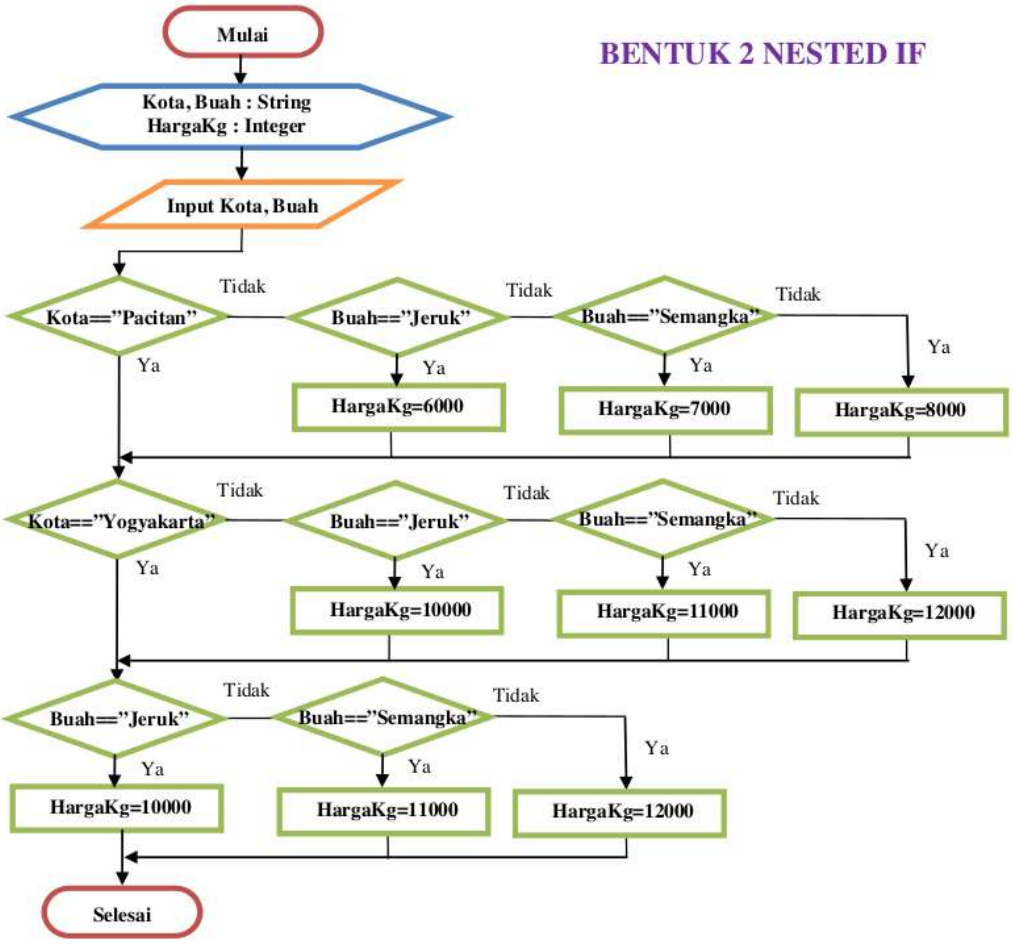
Jika kota supplier selain Pacitan dan Yogyakarta :

Jika jenis buah = Jeruk, maka Harga per kg = Rp 15.000

Jika jenis buah = Semangka, maka Harga per kg = Rp 16.000

Jika jenis buah lainnya, maka Harga per kg = Rp 17.000

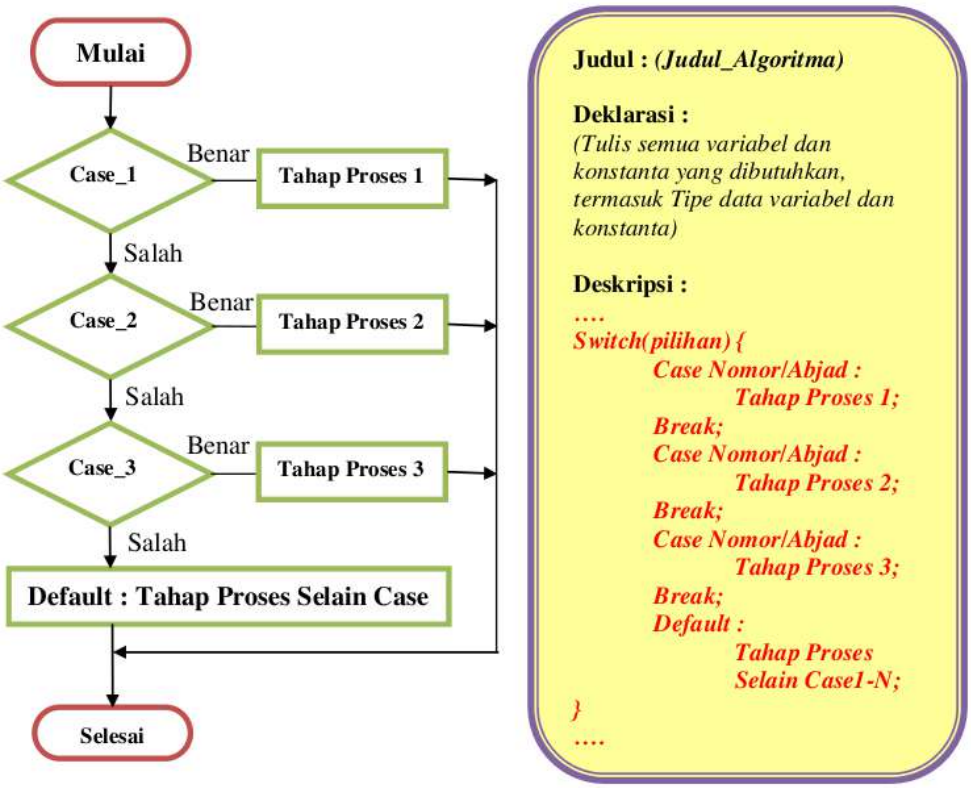
Output : Harga per Kg buah.



Gambar 5.10. Flowchart menentukan Harga Jual Buah pada setiap Kota Supplier.

5.1.4 Bentuk Switch Case

Bentuk Switch Case tidak jauh berbeda dengan bentuk if…then…else. Bentuk ini diciptakan sebagai pengganti if…then…else untuk syarat/kondisi yang lebih sederhana. Pada umumnya bentuk switch case ini banyak digunakan pada saat terdapat pemilihan menu. Sehingga syarat/kondisi yang digunakan hanya berdasar inputan berupa nomor atau abjad menu. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk Switch Case dapat dilihat pada Gambar 5.11.



Gambar 5.11. Bentuk Flowchart dan Penulisan Pseudocode Bentuk Switch Case.

Contoh : Flowchart menentukan Pilihan Menu Makanan.

Input :

Pilihan Menu : 1. Nasi Soto Ayam

2. Nasi Rawon

3. Batal/Keluar

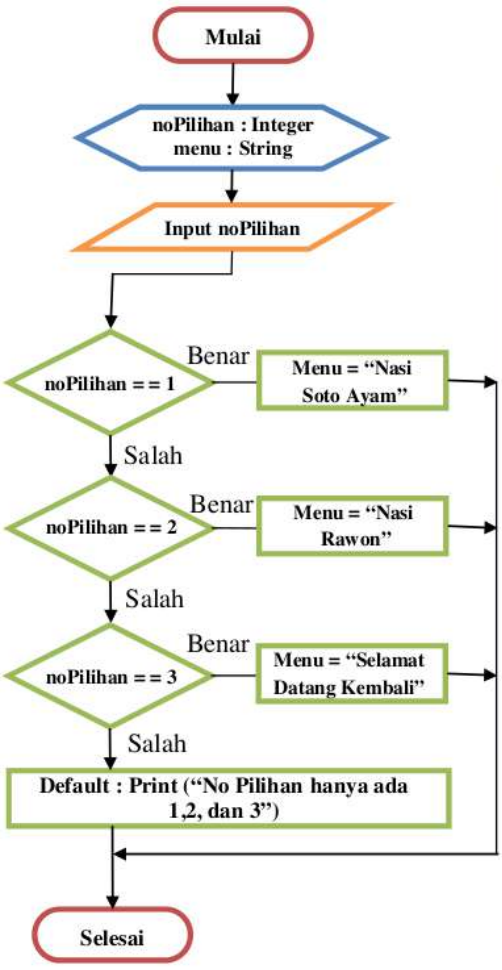
Syarat/Kondisi :

Jika memilih Menu no.1, maka akan memilih Nasi Soto Ayam

Jika memilih Menu no.2, maka akan memilih Nasi Rawon

Jika memilih Menu no.3, maka batal/keluar dari program

Output : Pesanan Makanan



Judul : Algoritma menentukan Pilihan Menu Makanan.

Deklarasi :
noPilihan : Integer
menuMakanan : String

Deskripsi :
Input (noPilihan)
10 tch(noPilihan) {
case 1 :
menu = "Nasi Soto Ayam"
break;
case 2 :
menu = "Nasi Rawon"
break;
case 3 :
menu = "Selamat Datang Kembali"
break;
default :
Print ("No Pilihan hanya ada no 1, 2, dan 3")
}

Gambar 5.12. Flowchart menentukan Pilihan Menu Makanan.

5.3 Rangkuman

- Proses percabangan adalah sebuah proses program dimana terdapat sebuah perintah/instruksi yang dapat dijalankan dengan syarat/kondisi tertentu harus dipenuhi terlebih dahulu. Apabila syarat/kondisi belum terpenuhi, maka perintah/instruksi tersebut tidak akan pernah dijalankan. Syarat atau kondisi yang digunakan boleh lebih dari satu kondisi (tidak terbatas).
- Proses percabangan juga dikenal sebagai proses pemilihan atau proses pengambilan keputusan.
- Adapun bentuk proses percabangan dibagi menjadi 4 bentuk, yaitu sebagai berikut :
  - Bentuk If Sederhana (If…Then)
  - Bentuk If…Then…Else
  - Bentuk Nested If
  - Bentuk Switch Case

5.4 Latihan Soal

Buatlah Flowchart dan Pseudocode untuk permasalahan Proses Percabangan berikut ini :

1. Masa Pendaftaran Mahasiswa baru dimulai minggu depan. Untuk menentukan jurusan yang tepat, Titi dan Didi meminta saran kepada petugas pendaftaran pada sebuah kampus ternama. Untuk keperluan tersebut, buatlah flowchart dan pseudocode untuk memudahkan petugas pendaftaran menentukan klasifikasi jurusan. Adapun klasifikasi tersebut adalah sebagai berikut :
   a. Jika calon mahasiswa pendaftar berjenis kelamin Laki-laki, maka disarankan untuk masuk Jurusan Teknik.
   b. Jika calon mahasiswa pendaftar berjenis kelamin Perempuan, maka disarankan untuk masuk Jurusan Ekonomi.

2. Mas Didi ingin membuka sebuah toko penjualan buah. Setiap jenis buah yang dijual berasal dari beberapa kota supplier yang berbeda. Untuk memudahkan proses transaksi penjualan buah kepada customer, Mas Didi meminta tolong kepada Titi untuk membuatkan flowchart dan pseudocode pada transaksi penjualan buah. Adapun ketentuan harga buah per kg ditentukan sebagai berikut :

Tabel Daftar Pasokan dan Harga per Kg Buah.

Yogyakarta			Jakarta			Bandung		
Jeruk	Semangka	Mangga	Jeruk	Semangka	Mangga	Jeruk	Semangka	Mangga
Rp 9.000	Rp 10.000	Rp 11.000	Rp 15.000	Rp 16.000	Rp 17.000	Rp 20.000	Rp 21.000	Rp 22.000

Setiap kali ada customer yang ingin melakukan traksasi pembelian, Titi diminta untuk memasukkan terlebih dahulu Asal Kota, Jenis Buah, dan Jumlah yang dibeli (Kg). Setelah diproses, maka output yang dihasilkan adalah Total Pembelian yang harus dibayar customer kepada Mas Didi.

3. Selain membantu Mas Didi pada usaha penjualan buahnya, Titi juga mulai merintis usahanya sendiri dengan membuka usaha Restoran Masakan Khas Jawa Timur. Adapun beberapa jenis masakan yang jual dan harga per porsinya adalah sebagai berikut :
   a. Rujak Cingur @ Rp 10.000
   b. Nasi Semanggi @ Rp 15.000
   c. Sate Klopo @ Rp 20.000

   Untuk memudahkan pelayanan dengan customer, Restoran Titi menggunakan pemesanan melalui aplikasi android. Untuk itu, Titi harus dapat membuat flowchart dan pseudocode untuk aplikasi restorannya yang dapat menampilkan Pilihan Menu Makanan, Jumlah Porsi yang dipesan, Harga per Porsi, dan Total Pembayaran Makanan.

Bab 6

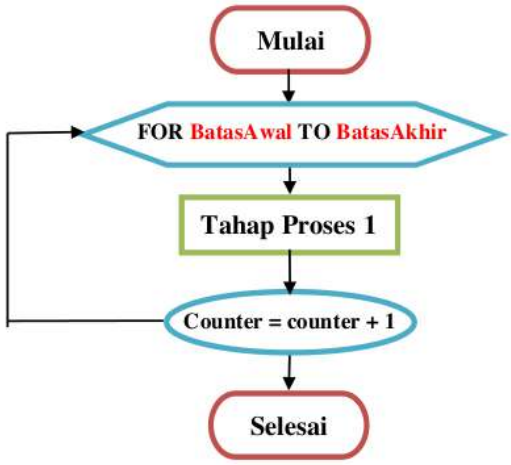
Perulangan (Looping/Iteration)

Tujuan :

- ✓ Mahasiswa dapat menjelaskan alur proses perulangan.
- ✓ Mahasiswa dapat mendemonstrasikan pembuatan struktur proses perulangan.

6.1 Proses Perulangan

Dalam sebuah algoritma, tahapan penyelesaian sebuah permasalahan dapat berupa struktur berurutan *(sequence)*, struktur pemilihan atau pengambilan keputusan *(selection)*, dan struktur pengulangan *(iteration)*. Ketiga jenis langkah tersebut membentuk struktur konstruktif suatu algoritma. Proses perulangan adalah sebuah proses program dimana setiap perintah/instruksi dapat dijalankan secara berulang dengan kondisi tertentu. Kondisi pada proses perulangan ini biasanya digunakan sebagai batas awal dan batas akhir perulangan. Apabila tidak ada kondisi sebagai batas perulangan, maka proses pada algoritma tersebut akan berjalan terus-menerus, sehingga mengakibatkan adanya *out of space memory*.



Gambar 6.1. Gambar Flowchart Proses Perulangan.

Berdasarkan Gambar 6.1, flowchart berjalan pada awal tahapan mulai, kemudian berjalan pada proses perulangan dengan batas awal dan batas akhir yang harus dipenuhi terlebih dahulu. Jika batas awal dan batas akhir telah diberi nilai awal, maka Tahap Proses 1 siap untuk dijalankan secara berulang. Proses perulangan tahap proses 1 akan terjadi melalui bantuan *counter*. *Counter* ini berfungsi sebagai penambah satu angka menuju tingkat perulangan berikutnya. Penambahan satu pada counter akan terjadi sampai batas akhir yang telah ditentukan sebelumnya. Jika penambahan counter telah lebih kecil atau sama dengan batas akhir, maka proses perulangan berhenti.

Proses perulangan ini dapat terjadi pada 3 bagian flowchart, yaitu pada bagian awal, tengah, maupun akhir flowchart. Sesuaikan dengan kondisi permasalahan yang ingin diselesaikan. Apabila proses percabangan diletakkan pada bagian awal, pada tahap bagian tengah dan akhir dapat diisi dengan proses sekuensial/percabangan lainnya yang merupakan proses kelanjutan dari proses perulangan. Sehingga tidak menutup kemungkinan, dalam satu flowchart terdapat proses sekuensial, percabangan, dan perulangan, hanya tergantung pada peletakan urutan prosesnya saja.

Contoh : Flowchart mencetak Kata "STIKOM Surabaya" sebanyak 5 kali.

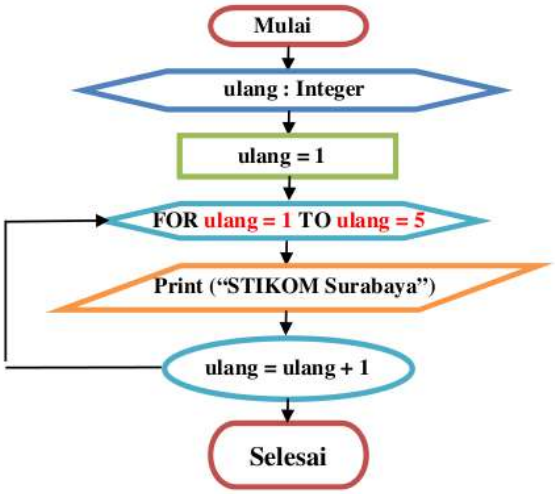
Input : -

Batas Awal = 1

Batas Akhir = 5

Proses yang diulang : Mencetak kata STIKOM Surabaya

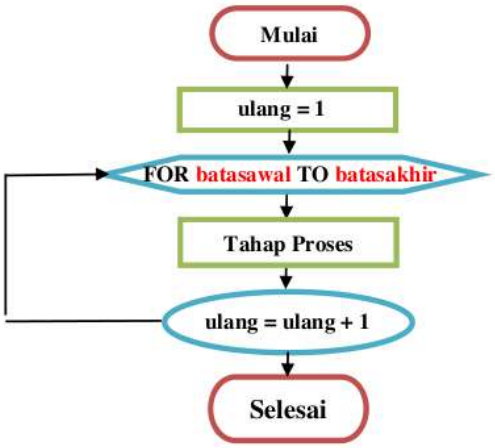
Output : Kata "STIKOM Surabaya" dicetak sebanyak 5 kali



Gambar 6.2. Flowchart mencetak Kata "STIKOM Surabaya" sebanyak 5 kali.

6.2 Counter

Proses perulangan dapat dilakukan melalui bantuan *counter*. *Counter* ini berfungsi sebagai pengontrol looping, yang dapat berupa variabel penambah satu angka menuju tingkat perulangan berikutnya. Penambahan satu pada counter terjadi sampai batas akhir yang telah ditentukan sebelumnya. Jika penambahan counter telah lebih kecil atau sama dengan batas akhir, maka proses perulangan berhenti. Cara kerja counter dengan nama variabel ulang dapat dilihat pada Gambar 6.3.



Gambar 6.3. Cara Kerja Counter pada Proses Perulangan.

6.3 Bentuk Proses Perulangan

Terdapat beberapa jenis bentuk proses perulangan yang penting untuk dipelajari sebagai konsep dasar perulangan. Bentuk Flowchart dan Pseuducode yang telah dipelajari pada sub bab sebelumnya merupakan bentuk umum dari proses perulangan. Adapun bentuk proses perulangan dibagi menjadi 4 bentuk, yaitu sebagai berikut :

1. **Bentuk For**

   Bentuk For ini merupakan bentuk dasar yang paling sederhana dibandingkan bentuk for yang lainnnya. Bentuk ini dikatakan paling sederhana karena paling mudah dalam prosesnya, masukkan batas awal dan akhir, dan gunakan counter sebagai variabel penambah satu. Dalam Bahasa Indonesia, bentuk ini dikenal dengan sebutan Perulangan untuk batas awal sama dengan sekian sampai batas akhir sama dengan sekian. Bentuk For juga dapat digunakan sebagai dasar pemrosesan untuk Array 1D yang akan dipelajari pada Bab 7.

2. Bentuk Do…While

Bentuk Do…While merupakan bentuk perulangan selain for yang menggunakan kondisi, seperti pada IF…THEN. Kondisi tersebut merupakan batas perulangan yang dapat ditentukan pada program. Proses perulangan dan counter dapat dijalankan apabila kondisi batasnya masih terpenuhi. Dalam Bahasa Indonesia, bentuk ini dikenal dengan kata LAKUKAN proses perulangan SELAMA masih memenuhi kondisi batas yang ditentukan. Hanya saja, pada penulisan pseudocode, kondisi berada di posisi paling bawah.

3. Bentuk While

Bentuk While merupakan bentuk perulangan kebalikan dari Do…While. Jika pada perulangan yang menggunakan Do…While, kondisi batas berada di posisi paling bawah, sedangkan pada perulangan yang menggunakan While, kondisi batas berada di posisi paling atas. Dalam Bahasa Indonesia, bentuk ini dikenal dengan kata SELAMA kondisi masih memenuhi batas yang ditentukan, LAKUKAN proses perulangan.

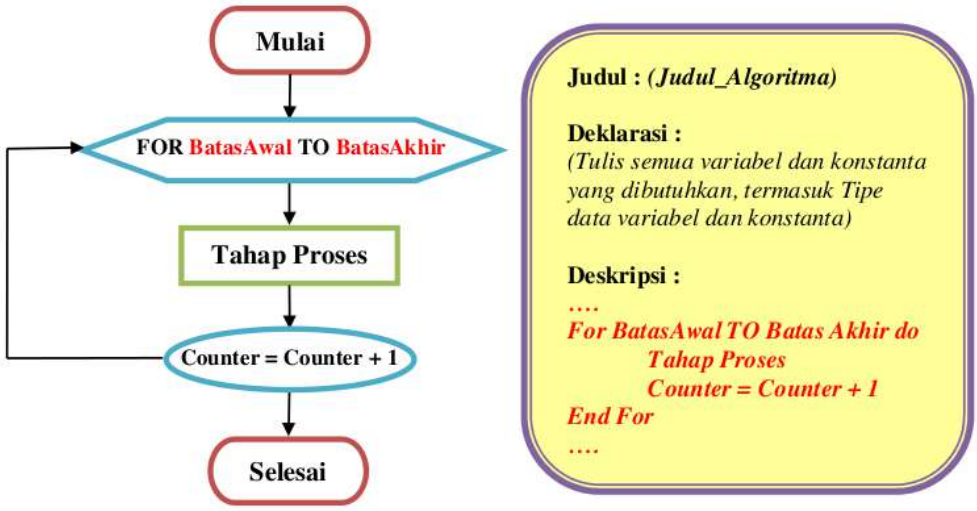
4. Bentuk Nested For

Bentuk Nested For memiliki konsep yang sama dengan bentuk Nested IF. Bentuk Nested For merupakan bentuk for di dalam for, yang biasa disebut For Bersarang atau For Bertingkat. Bentuk Nested For biasanya sering diaplikasikan untuk mencetak angka atau karakter yang mempunyai struktur baris dan kolom. Oleh karena itu, nested for merupakan dasar pemrosesan untuk Array 2D yang akan dipelajari pada Bab 8.

6.3.1 Bentuk FOR

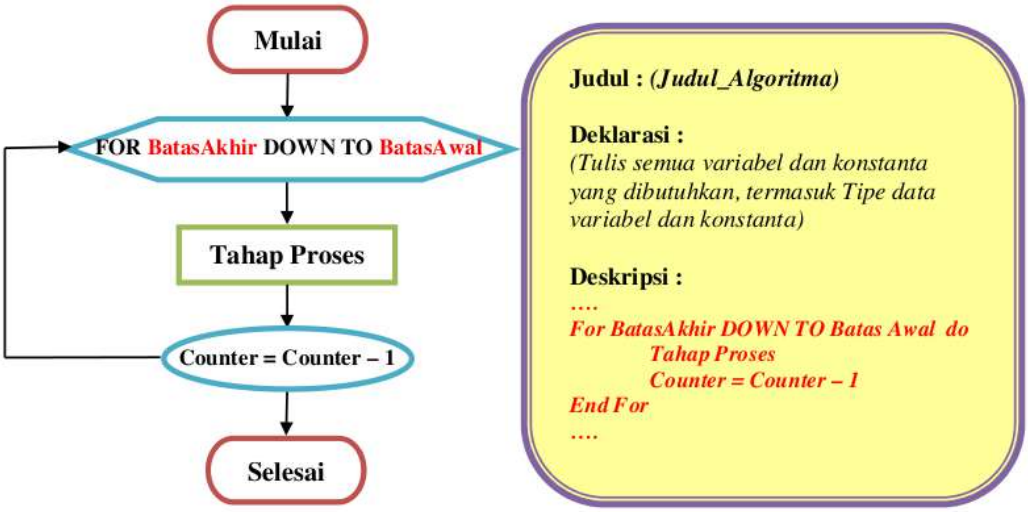
Bentuk For ini merupakan bentuk dasar yang paling sederhana dibandingkan bentuk for yang lainnnya. Bentuk ini dikatakan paling sederhana karena paling mudah dalam prosesnya, masukkan batas awal dan akhir, dan gunakan counter sebagai variabel penambah satu. Dalam Bahasa Indonesia, bentuk ini dikenal dengan sebutan Perulangan untuk batas awal sama dengan sekian samapai batas akhir sama dengan sekian. Bentuk For juga dapat digunakan sebagai dasar pemrosesan untuk Array 1D. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk For dapat dilihat pada Gambar 6.4 dan 6.5.

BENTUK FOR – ASCENDING



Gambar 6.4. Bentuk Flowchart dan Penulisan Pseudocode Bentuk For – Ascending.

BENTUK FOR – DESCENDING



Gambar 6.5. Bentuk Flowchart dan Penulisan Pseudocode Bentuk For – Descending.

Contoh 1 : Flowchart mencetak deret angka 1-10 secara Ascending (FOR).

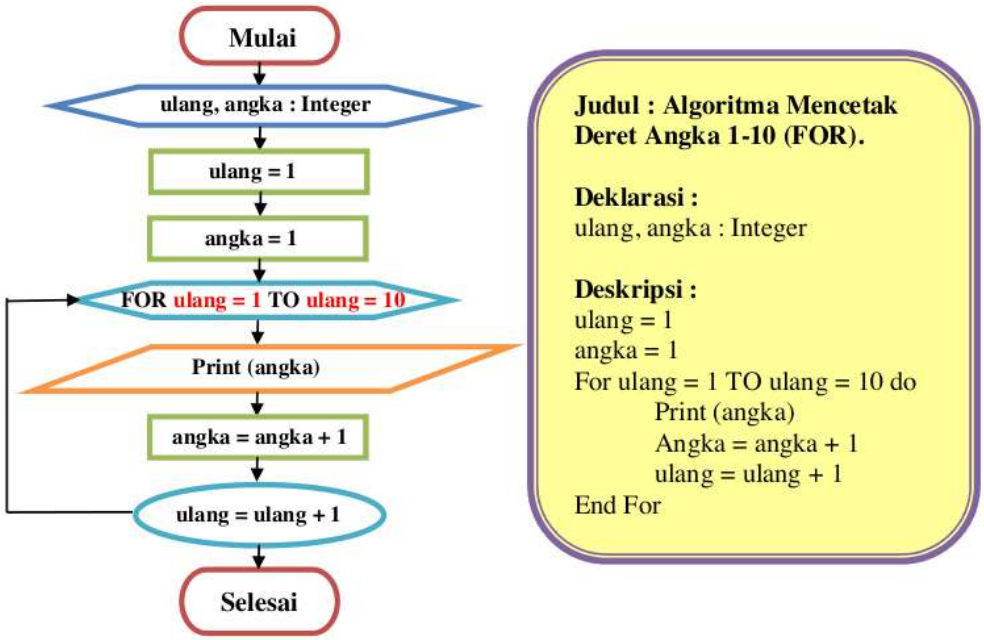
Input : -

Batas Awal = 1

Batas Akhir = 10

Proses yang diulang : Mencetak angka

Output : Mencetak deret angka 1, 2, 3, 4, 5, 6, 7, 8, 9, 10



Gambar 6.6. Flowchart mencetak deret angka 1-10 secara Ascending (FOR).

Contoh 2 : Flowchart mencetak deret angka 1-10 secara Descending.

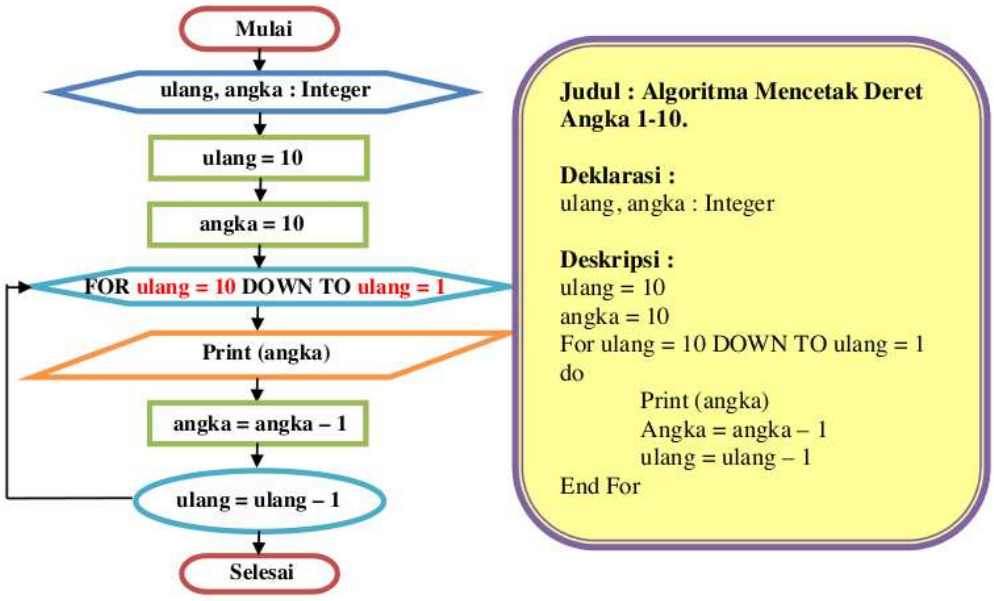
Input : -

Batas Awal = 10

Batas Akhir = 1

Proses yang diulang : Mencetak angka

Output : Mencetak deret angka 10, 9, 8, 7, 6, 5, 4, 3, 2, 1.



Gambar 6.7. Flowchart mencetak deret angka 1-10 secara Descending.

6.3.2 Bentuk DO-WHILE

Bentuk Do…While merupakan bentuk perulangan selain for yang mempunyai kondisi, seperti pada IF…THEN. Kondisi tersebut merupakan batas perulangan yang dapat ditentukan pada program. Dalam Bahasa Indonesia, bentuk ini dikenal dengan kata LAKUKAN proses perulangan SELAMA masih memenuhi kondisi batas yang ditentukan. Hanya saja, pada penulisan pseudocode, kondisi berada di posisi paling bawah. Adapun bentuk flowchart dan pseudocode untuk Bentuk Do…While dapat dilihat pada Gambar 6.8.

Gambar 6.8. Bentuk Flowchart dan Penulisan Pseudocode Bentuk Do While.

Contoh : Flowchart mencetak deret angka 1-10 secara Ascending (DO…WHILE).

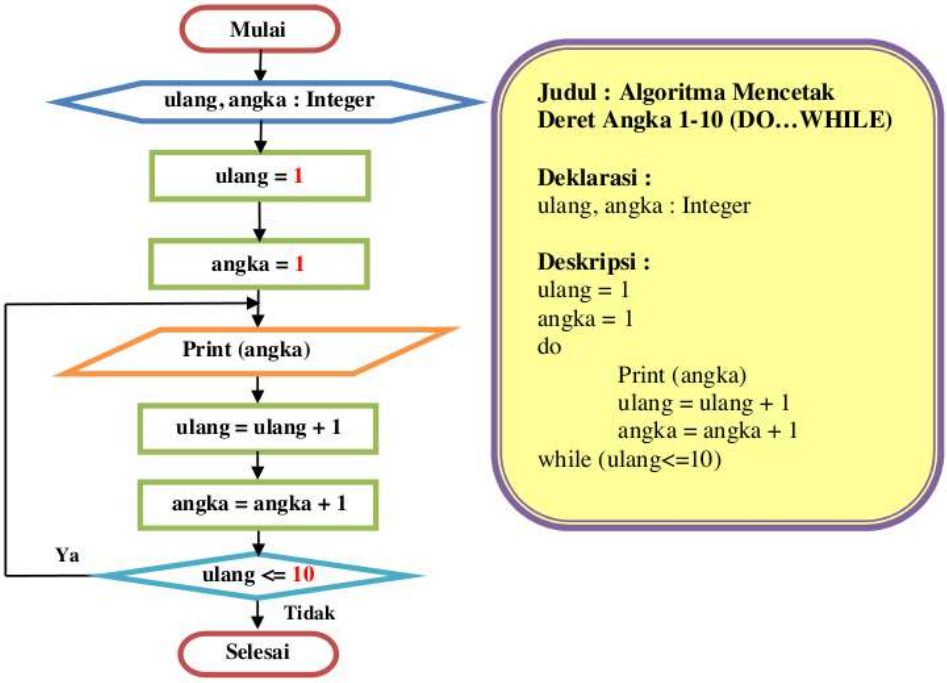
Input : -

Batas Awal = 1

Batas Akhir = 10

Proses yang diulang : Mencetak angka

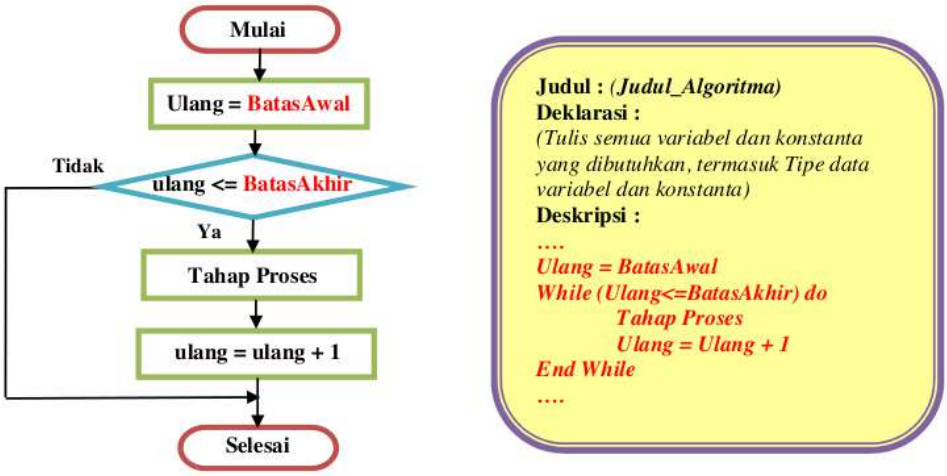
Output : Mencetak deret angka 1, 2, 3, 4, 5, 6, 7, 8, 9, 10



Gambar 6.9. Flowchart mencetak deret angka 1-10 secara Ascending (DO WHILE).

6.3.3 Bentuk WHILE

Bentuk While merupakan bentuk perulangan kebalikan dari Do…While. Jika pada perulangan yang menggunakan Do…While, kondisi batas berada di posisi paling bawah, sedangkan pada perulangan yang menggunakan While, kondisi batas berada di posisi paling atas. Dalam Bahasa Indonesia, bentuk ini dikenal dengan kata SELAMA kondisi masih memenuhi batas yang ditentukan, LAKUKAN proses perulangan. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk While dapat dilihat pada Gambar 6.10.



Gambar 6.10. Bentuk Flowchart dan Penulisan Pseudocode Bentuk While.

Contoh : Flowchart mencetak deret angka 1-10 secara Ascending (WHILE).

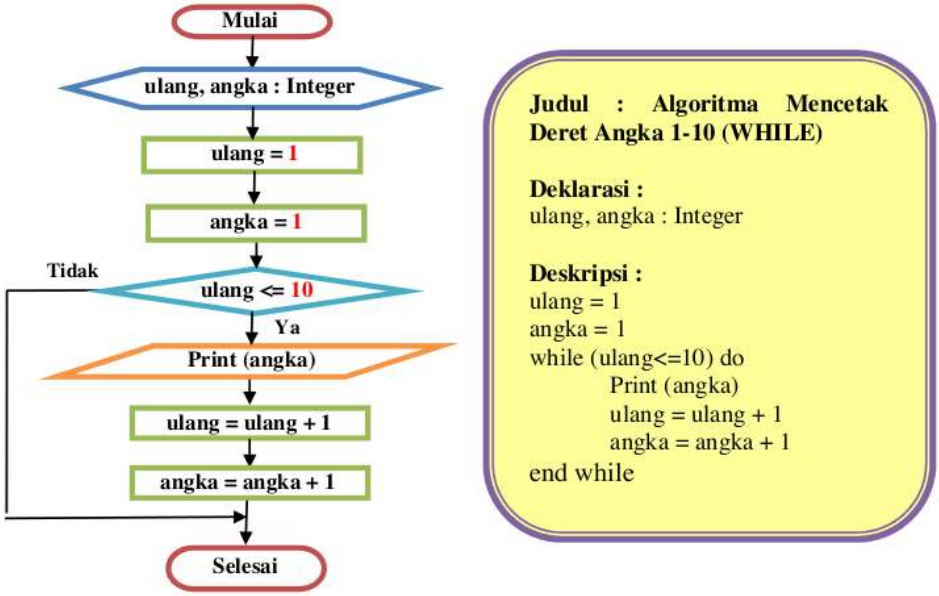
Input : -

Batas Awal = 1

Batas Akhir = 10

Proses yang diulang : Mencetak angka

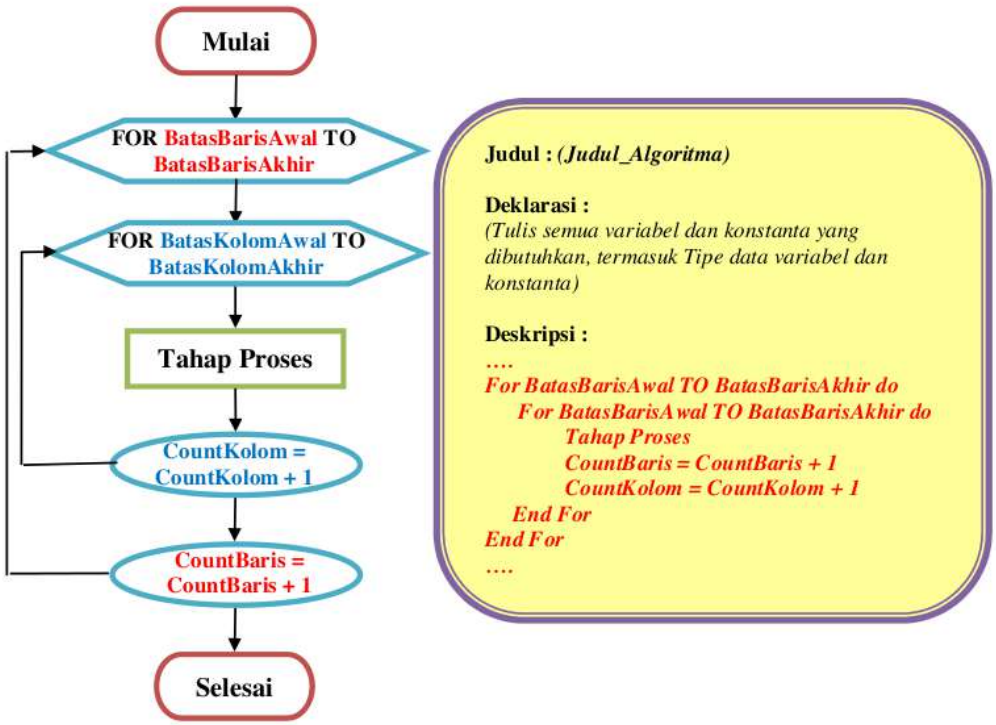
Output : Mencetak deret angka 1, 2, 3, 4, 5, 6, 7, 8, 9, 10



Gambar 6.11. Flowchart mencetak deret angka 1-10 secara Ascending (WHILE).

6.3.4 Bentuk Nested FOR

Bentuk Nested For memiliki konsep yang sama dengan bentuk Nested IF. Bentuk Nested For merupakan bentuk for di dalam for, yang biasa disebut For Bersarang atau For Bertingkat. Bentuk Nested For biasanya sering diaplikasikan untuk mencetak angka atau karakter yang mempunyai struktur baris dan kolom. Oleh karena itu, nested for merupakan dasar pemrosesan untuk Array 2D. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Bentuk Nested For dapat dilihat pada Gambar 6.12.



Gambar 6.12. Bentuk Flowchart dan Penulisan Pseudocode Bentuk NestedFor.

Contoh : Flowchart mencetak deret perkalian angka 1-5 secara urut.

Input : -

Batas Baris Awal = 1

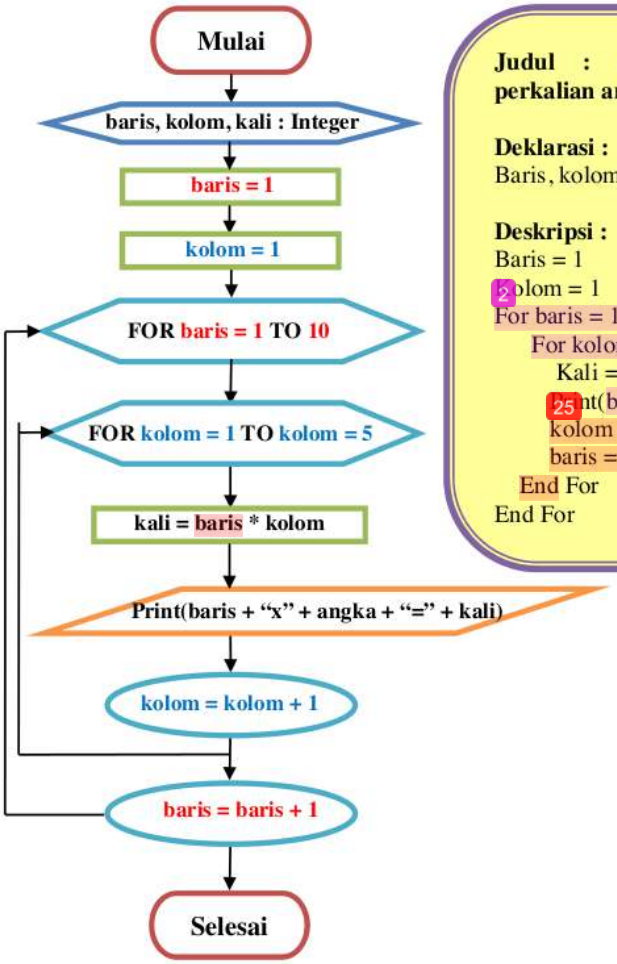
Batas Baris Akhir = 10

Batas Kolom Awal = 1

Batas Kolom Akhir = 5

Proses yang diulang : Mengalikan angka 1-5

Output : Mencetak deret perkalian angka 1-5.



Judul : Algoritma mencetak deret perkalian angka 1-5 secara urut.

Deklarasi :
Baris, kolom, kali : integer

Deskripsi :
Baris = 1
kolom = 1
For baris = 1 to 10 do
  For kolom = 1 to 5 do
    Kali = baris * kolom
    Print(baris + "x" + angka + "=" + kali)
    kolom = kolom+1
    baris = baris + 1
  End For
End For

Gambar 6.13. Flowchart mencetak deret perkalian angka 1-5 secara urut.

6.4 Rangkuman

- Proses percabangan adalah sebuah proses program dimana setiap perintah/instruksi dapat dijalankan secara berulang dengan kondisi tertentu. Kondisi pada proses perulangan ini biasanya digunakan sebagai batas awal dan batas akhir perulangan.
- Proses perulangan dapat dilakukan melalui bantuan *counter*. *Counter* ini berfungsi sebagai pengontrol looping, yang dapat berupa variabel penambah satu menuju tingkat perulangan berikutnya. Penambahan satu pada counter terjadi sampai batas akhir yang telah ditentukan sebelumnya.
- Adapun bentuk proses percabangan dibagi menjadi 4 bentuk, yaitu sebagai berikut :
  - Bentuk For
  - Bentuk Do…While
  - Bentuk While
  - Bentuk Nested For

6.5 Latihan Soal

Pada sebuah toko swalayan, Titi menemukan promo baru yang ditawarkan oleh pihak swalayan. Promo tersebut diberlakukan untuk pembelian shampoo dengan jumlah genap, minimal pembelian 2 botol shampoo akan diberikan gratis 1 botol shampoo yang sama. Promo ini berlaku untuk kelipatan. Buatlah daftar untuk mencatat berapa jumlah shampoo gratis yang harus diberikan kepada customer apabila dibatasi pembelian shampoo maksimal adalah 10 botol.

2 botol	4 botol	6 botol	8 botol	10 botol	12 botol	14 botol	16 botol	18 botol	20 botol
1 botol	2 botol	3 botol	4 botol	5 botol	6 botol	7 botol	8 botol	9 botol	10 botol

Buatlah Flowchart dan Pseudocode untuk permasalahan diatas menggunakan konsep Proses Perulangan berikut ini :

1. Bentuk For
2. Bentuk Do…While
3. Bentuk While
4. Bentuk Nested For

Bab 7

Array 1D (Satu Dimensi)

Tujuan :

- ✓ Mahasiswa dapat membedakan penggunaan Array 1D dan Array 2D.
- ✓ Mahasiswa dapat mendemonstrasikan pembuatan algoritma menggunakan konsep Array 1D.

7.1 Pengertian Array/Larik

Array atau Larik merupakan kumpulan dari beberapa nilai yang mempunyai tipe data yang sama. Array dapat disebut sebagai sebuah variabel yang mempunyai lebih dari satu nilai. Sedangkan variabel yang selama ini dikenal hanya mempunyai satu nilai saja. Array dapat diberi nama seperti nama variabel biasa. Namun, harus tetap memperhatikan kaidah atau aturan penulisan variabel, seperti yang telah dipelajari pada Bab 2. Array disebut juga sebagai Larik/Tabel/Vektor/Variabel Majemuk.

Ada beberapa hal penting yang menjadi ciri-ciri dari sebuah array, antara lain sebagai berikut :

a. Array dibagi menjadi 2 bentuk dimensi, yaitu **array satu dimensi dan array dua dimensi.**

b. Variabel array dapat **terdiri dari satu atau beberapa elemen.**

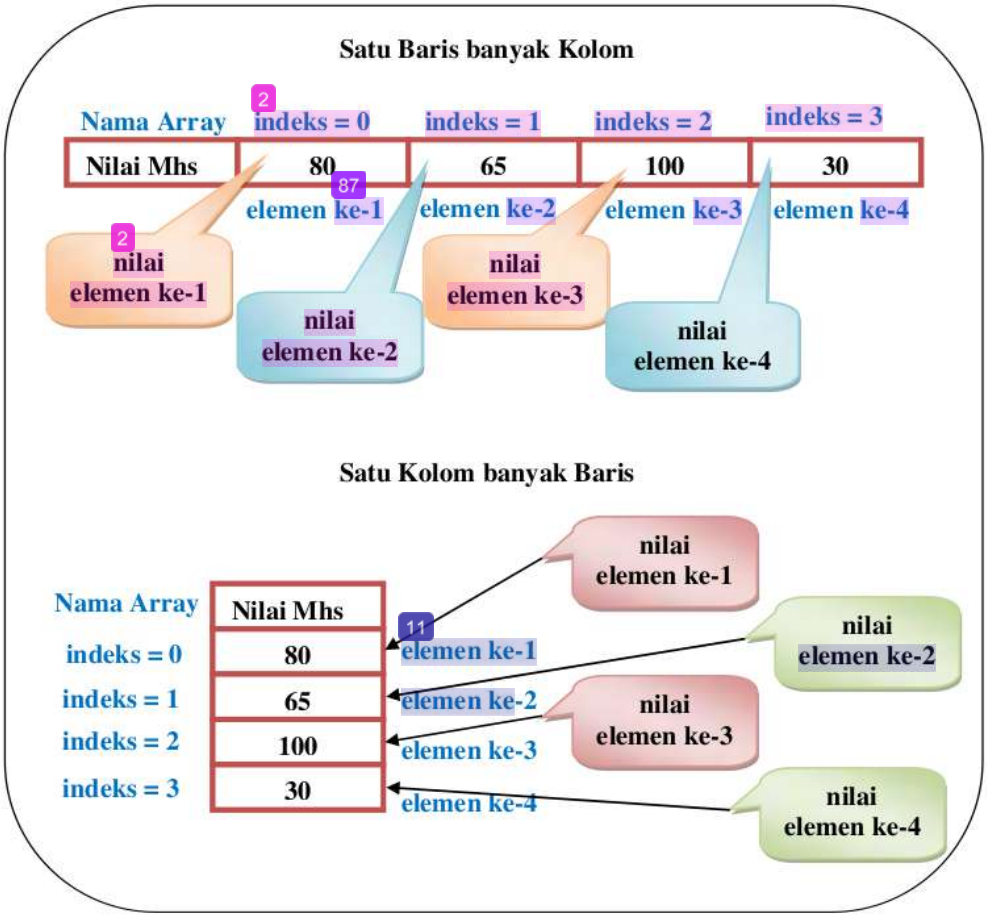
c. Nilai yang mengisi setiap elemen tersebut harus spesifik, yaitu **mempunyai tipe data yang sama.**

d. Setiap elemen pada sebuah array akan ditandai dengan sebuah **indeks.** Indeks tersebut berfungsi sebagai nomor urut elemen.

e. Mengingat algoritma yang digunakan pada buku ini ditulis menggunakan notasi C/Java, maka indeks pada array selalu dimulai dari **angka nol (0).**

f. Jumlah elemen array dapat dideklarasikan langsung pada awal program dan **harus lebih kecil atau sama dengan indeks terbesar** pada sebuah array.
g. Jumlah indeks yang ditentukan menunjukkan **banyaknya memori yang dialokasikan**.
h. Semua proses yang terjadi pada sebuah array, mulai dari proses pengisian nilai elemen array, proses perhitungan, maupun proses penampilan data array dapat menggunakan 2 cara, yaitu melalui proses inisialisasi langsung atau **menggunakan proses looping for.**

Array satu dimensi merupakan array yang terdiri dari satu baris banyak kolom atau satu kolom banyak baris. Sedangkan array dua dimensi merupakan array yang terdiri dari banyak baris dan banyak kolom. Pada gambar 7.1 di bawah ini akan dijelaskan bentuk dari array satu dimensi.



Gambar 7.1. Bentuk Array Satu Dimensi.

Notes : *Cara paling mudah memahami array adalah dengan menggunakan* ***Konsep Tabel****. Hal ini dikarenakan* ***tabel merupakan representasi nyata dari sebuah array*** *yang terdiri dari 2 komponen penting yaitu* ***baris dan kolom****.*

7.2 Deklarasi Array 1D

Sebuah array yang akan digunakan pada sebuah program atau algoritma harus dideklarasi terlebih dahulu. Hal ini dilakukan dengan tujuan agar variabel array dikenali di semua baris program atau di setiap tahapan algoritma. Adapun aturan pendeklarasian array adalah sebagai berikut :

a. Array dideklarasikan dengan menggunakan **nama variabel**.
b. Setelah diberi nama variabel, diikuti dengan penulisan kata "**Array**".
c. Tambahkan **tanda kurung siku** (**[…]**) yang **ditulis tanpa koma** (menandakan array satu dimensi). Jika kurung siku ditulis menggunakan tanda koma, maka array tersebut adalah array dua dimensi. Di dalam kurung siku diisi dengan **jangkauan** jumlah data atau jumlah elemen array yang dibutuhkan.
d. Setelah tanda kurung siku, tambahkan pula kata "**Of**" diikuti **tipe data** array.

Perhatikan Gambar 7.2 untuk bentuk deklarasi variabel array satu dimensi di bawah ini.

```
Var  Nama_variabel_array : Array [jangkauan] Of  Tipe_Data
```

Contoh :

```
Var nilaiSiswa : Array [0..9] Of  Integer
```

Gambar 7.2. Bentuk Deklarasi Variabel Array Satu Dimensi.

Berdasarkan gambar 7.2, dapat diketahui bahwa nama variabel array adalah nilaiSiswa dengan jangkauan data mulai dari indeks 0 sampai 9, sehingga pada array tersebut mempunyai 10 elemen array. Tipe data yang digunakan pada nilai elemen array adalah integer (bilangan bulat). Apabila pada salah satu elemen array diisi dengan nilai bukan bilangan bulat (Real/String), maka hal tersebut akan terbaca sebagai sebuah kesalahan.

Contoh lain deklarasi array satu dimensi adalah sebagai berikut :

- Var namaSiswa : Array[0..4] Of String
- Var jumlahBarang : Array[0..100] Of Integer
- Var nilaiUnas : Array[0..39] Of Real

7.3 Cara Inisialisasi Elemen Array 1D

Setelah melakukan proses deklarasi array, langkah selanjutnya yang harus dilakukan adalah mengisi nilai elemen array. Pengisian nilai elemen array dapat dilakukan dengan 2 cara, diantaranya sebagai berikut :

a. Cara Inisialisasi Nilai Elemen Array.
b. Cara Membaca Inputan User.

Cara inisialisasi nilai elemen array dapat dilakukan dengan lebih mudah dibandingkan cara membaca inputan user, meskipun terkesan lebih panjang (sesuai jumlah elemen array). Cara inisialisasi hampir sama dengan cara deklarasi. Hanya saja pada saat deklarasi langsung disebutkan nilai setiap elemen array atau dengan menyebut indeks setiap elemen. Namun, jika menggunakan cara pembacaan inputan user, harus menggunakan konsep perulangan (looping). Adapun aturan yang harus diperhatikan dalam proses inisialisasi nilai elemen array, diantaranya sebagai berikut :

1. **Langsung menyebutkan nilai elemen array pada saat deklarasi.**
   (Lihat Gambar 7.3)

```
Var Nama_variabel_array : Array [jangkauan] Of Tipe_Data =
(nilai_elemen_array)
```

Contoh :

```
Var nilaiSiswa : Array [0..4] Of Integer = (30, 95, 75, 63, 100)
```

Gambar 7.3. Bentuk Pertama Inisialisasi Nilai Elemen Array Satu Dimensi.

2. **Pemberian nilai elemen array dengan menyebutkan indeks elemennya.**
   (Lihat Gambar 7.4)

```
Nama_variabel_array [indeks] = nilai_elemen_indeks
```

Contoh :

```
nilaiSiswa[0] = 30      nilaiSiswa[2] = 75      nilaiSiswa[4] = 100
nilaiSiswa[1] = 95      nilaiSiswa[3] = 63
```

Gambar 7.4. Bentuk Kedua Inisialisasi Nilai Elemen Array Satu Dimensi.

7.4 Cara Menginputkan Nilai Elemen Array 1D

Cara menginputkan nilai elemen array sama artinya dengan membaca inputan user. Tidak seperti cara inisialisasi yang langsung menyebutkan nilai elemennya, cara membaca inputan user merupakan sebuah proses dimana nilai pada elemen array besarnya ditentukan oleh user. Bukan berdasarkan inisialisasi pada awal algoritma. Cara membaca inputan user dilakukan menggunakan konsep perulangan (looping).

Konsep perulangan (looping) yang paling mudah digunakan adalah menggunakan konsep perulangan bentuk pertama, yaitu FOR. Jika pengisian nilai elemen array dilakukan menggunakan looping FOR, maka hanya membutuhkan 3 baris notasi algoritma. Perhatikan penulisan notasi algoritma untuk pembacaan inputan user array satu dimensi pada Gambar 7.5.

```
FOR nilai_awal TO nilai_akhir DO
        INPUT (Nama_Var_Array[indeks_ke]
END FOR
```

Contoh :

```
21
FOR i = 0 TO 9 DO
        INPUT (nilaiSiswa[i])
END FOR
```

Gambar 7.5. Bentuk Notasi Algoritma untuk Input Nilai Elemen Array Satu Dimensi.

Berdasarkan gambar 7.5 dapat dilihat bahwa nilai awal i = 0 digunakan sebagai indeks array yang selalu dimulai dari 0, sampai dengan batas akhir jangkauan = 9. Sehingga apabila diimplementasikan, maka user akan diminta melakukan proses pengisian nilai elemen array sebanyak 10 kali.

Di dalam looping for ditambahkan notasi INPUT untuk membaca inputan user. Pembacaan nilai elemen array dilakukan dengan menyebutkan nama variabel array diikuti dengan indeks. Karena penulisan notasi menggunakan looping FOR, maka untuk penulisan indeks digantikan dengan variabel i sebagai variabel *counter* (penambah satu). Dengan counter tersebut, maka indeks array secara otomatis akan berganti (tambah satu) selama proses perulangan berlangsung atau sampai batas jangkauan akhir sama dengan 9.

7.5 Cara Pemrosesan Nilai Elemen Array 1D

Cara memproses nilai elemen array merupakan tahap lanjutan setelah proses penginputan nilai elemen array. Pada dasarnya, tahapan paling mudah pada array adalah pengisian nilai elemen array dilanjutkan dengan proses menampilkan nilai elemen array, tanpa adanya pemrosesan lebih lanjut terhadap nilai elemen array. Namun, pemrosesan nilai elemen array mempunyai banyak manfaat yang dapat diambil, terutama dapat diaplikasikan terhadap semua kebutuhan user, mulai dari pembelian, penjualan, maupun proses bisnis lainnya. **Satu hal yang penting dan harus diingat pada pemrosesan array satu dimensi adalah, selalu gunakan konsep perulangan (looping) bentuk FOR.**

Adapun 3 pemrosesan pada array yang sering dibutuhkan pada pembuatan algoritma, yaitu sebagai berikut :

1. **Proses Aritmatika.**

   Pemrosesan elemen array paling mudah adalah digunakan untuk proses aritmatika. Mulai dari proses penambahan, pengurangan, perkalian, pembagian, dan lain sebagainya. Pada pemrosesan array yang menggunakan proses aritmatika, nilai elemen array yang akan diproses harus berupa angka/bilangan, baik bilangan bulat (Integer) maupun bilangan pecahan (Real). Selain itu, proses aritmatika dapat diterapkan pada elemen array dengan syarat minimal terdapat dua buah array untuk bertugas sebagai operand dan hasil. Contoh Notasi algoritma untuk pemrosesan array menggunakan proses aritmatika penjumlahan dapat dilihat pada Gambar 7.6.

```
FOR nilai_awal TO nilai_akhir DO
        nama_array_hasil[indeks] = nama_arrayOperand1[indeks] + Operand2
END FOR
```

Contoh :

```
21
FOR i = 0 TO 9 DO
        nilaiTugasAkhir[i] = nilaiTugas[i] + 20
END FOR
```

```
FOR i = 0 TO 9 DO
        nilaiAkhir[i] = nilaiTugas[i] + nilaiUjian[i]
END FOR
```

Gambar 7.6. Notasi Algoritma untuk Proses Aritmatika Penjumlahan pada Array 1D.

Pada gambar 7.7 di bawah ini dapat dilihat bentuk notasi algoritma untuk proses aritmatika pengurangan.

```
FOR nilai_awal TO nilai_akhir DO
        nama_array_hasil[indeks] = nama_arrayOperand1[indeks] – Operand2
END FOR
```

Contoh :

```
FOR i = 0 TO 9 DO
        hargaBaju[i] = hargaAwal[i] – diskon
END FOR
```

```
FOR i = 0 TO 9 DO
        hargaBarang[i] = hargaAwal[i] – hargaDiskon[i]
END FOR
```

Gambar 7.7. Notasi Algoritma untuk Proses Aritmatika Pengurangan.

Pada gambar 7.8 di bawah ini dapat dilihat bentuk notasi algoritma untuk proses aritmatika perkalian.

```
FOR nilai_awal TO nilai_akhir DO
        nama_array_hasil[indeks] = nama_arrayOperand1[indeks] * Operand2
END FOR
```

Contoh :

```
FOR i = 0 TO 9 DO
        nilaiTugasAkhir[i] = nilaiTugas[i] * 0.7
END FOR
```

```
FOR i = 0 TO 9 DO
        nilaiAkhir[i] = (nilaiTugas[i] * 0.7) + (nilaiUjian[i] *0.3)
END FOR
```

Gambar 7.8. Notasi Algoritma untuk Proses Aritmatika Perkalian.

Pada gambar 7.9 di bawah ini dapat dilihat bentuk notasi algoritma untuk proses aritmatika pembagian.

```
FOR nilai_awal TO nilai_akhir DO
        nama_array_hasil[indeks] = nama_arrayOperand1[indeks] / Operand2
END FOR
```

Contoh :

```
FOR i = 0 TO 9 DO
        hargaDiskon[i] = hargaBrg[i] * 20 / 100
END FOR
```

```
FOR i = 0 TO 9 DO
        hargaperBrg[i] = (TotalPerBrg[i] / jumlahBrg
END FOR
```

Gambar 7.9. Notasi Algoritma untuk Proses Aritmatika Pembagian.

Notasi-notasi algoritma yang digambarkan pada Gambar 7.7 sampai 7.9 merupakan notasi sederhana dari setiap proses aritmatika yang sering digunakan, Untuk proses aritmatika yang lainnya, seperti perpangkatan atau modulus dapat dikembangkan sendiri sesuai dengan kebutuhan dan permasalahan pada algoritma. Namun, untuk notasi operator aritmatika yang digunakan, tetap mengikuti kaidah atau atura penulisan operator pada Bab 2.

2. Pencarian Nilai Maksimal dan Minimal.

Setelah memahami proses dasar aritmatika yang telah dibahas pada sub bab sebelumnya, terdapat proses yang juga paling sering digunakan dalam pemrosesan sebuah elemen array, yaitu proses pencarian nilai maksimal dan minimal. Proses pencarian nilai maksimal dan minimal ini biasanya digunakan dalam permasalahan yang sering dijumpai sehari-hari, contohnya mencari nilai terbesar/terkecil dari sekian mahasiswa dalam 1 kelas. Selain itu, juga dapat digunakan untuk mencari jumlah penjualan terbesar/terkecil dari pemasaran sebuah produk di beberapa kota.

Konsep pencarian nilai maksimal dan minimal harus menggunakan konsep percabangan (selection). Hal ini dikarenakan perlunya proses perbandingan antara

nilai pada indeks pertama dengan nilai pada indeks selanjutnya, sampai nilai pada indeks terakhir. Proses perbandingan ini diletakkan di dalam proses perulangan. Hal ini akan mempermudah dalam proses algoritma, biarkan proses perbandingan berjalan secara otomatis mengikuti counter pada proses perulangan, tanpa harus menulis notasi perbandingan satu per satu setiap nilai elemen array.

Secara logika, sebelum melakukan proses perbandingan setiap nilai elemen array, **deklarasikan terlebih dahulu sebuah variabel yang akan menampung nilai terbesar (maks).** Setelah dideklarasi, langkah selanjutnya adalah **memberikan nilai awal bahwa variabel maks tersebut diisi dengan nilai elemen indeks pertama**. Jadi, anggap nilai elemen indeks pertama merupakan nilai terbesar. **Jika nilai pada indeks selanjutnya lebih besar dari variabel maks, maka variabel maks akan diisi nilai baru *(initialization)* dari nilai indeks tersebut. Jika tidak, nilai pada indeks pertama akan tetap mengisi variabel maks.** Lakukan hal yang sama pada saat mencari nilai terkecil. Pada gambar 7.10 dapat dilihat sedikit perbedaan notasi perbandingan saat mencari nilai terbesar dan terkecil.

Mencari Nilai Maksimal/Terbesar

```
maks = nama_array[0]

FOR nilai_awal TO nilai_akhir DO
        if (nama_array[indeks_lanjut] > maks) then
                maks = nama_array[indeks_lanjut]
        end if
END FOR
```

Mencari Nilai Minimal/Terkecil

```
min = nama_array[0]

FOR nilai_awal TO nilai_akhir DO
        if (nama_array[indeks_lanjut] < min) then
                min = nama_array[indeks_lanjut]
        end if
END FOR
```

Gambar 7.10. Notasi Algoritma Mencari Nilai Maksimal dan Minimal pada Array 1D.

Pada gambar 7.11 dapat dilihat contoh mencari nilai terbesar dan terkecil.

Contoh Mencari Nilai Maksimal/Terbesar

```
maks = nilaiMhs[0]
FOR i=0 TO 9 DO
        if (nilaiMhs [i+1] > maks) then
                maks = nilaiMhs [i+1]
        end if
END FOR
```

Contoh Mencari Nilai Minimal/Terkecil

```
min = nilaiMhs[0]
FOR i=0 TO 9 DO
        if (nilaiMhs [i+1] < min) then
                min = nilaiMhs [i+1]
        end if
END FOR
```

Gambar 7.11. Contoh Mencari Nilai Maksimal dan Minimal pada Array 1D.

3. Perhitungan Nilai Rata-rata.

Selain proses aritmatika dan pencarian nilai maksimal/minimal, terdapat satu proses lagi yang sering digunakan sebagai pemrosesan nilai elemen array, yaitu perhitungan nilai rata-rata. Proses perhitungan nilai rata-rata ini biasanya digunakan untuk mencari nilai rata-rata mahasiswa dalam satu kelas. Secara logika, proses perhitungan nilai rata-rata bisa didapatkan dengan cara menjumlahkan seluruh nilai mahasiswa, kemudian dibagi dengan jumlah mahasiswa yang ada dalam kelas tersebut. Notasi algoritma untuk proses perhitungan nilai rata-rata dapat dilihat pada Gambar 7.12.

Proses untuk menjumlahkan seluruh nilai mahasiswa adalah dengan menambahakan deklarasi satu variabel yang akan menampung jumlah keseluruhan (total). Setelah dideklarasikan, langkah selanjutnya adalah memberikan nilai awal *(initialization)* variabel total dengan nilai 0. Nilai total tersebut akan terus bertambah, jika perhitungan total dilakukan menggunakan proses perulangan untuk indeks selanjutnya.

```
total = 0
FOR nilai_awal TO nilai_akhir DO
        total = total + nama_array[indeks]
END FOR
rata = total/N
```

Contoh :

```
total = 0
FOR I=0 TO 9 DO
        total = total + nilaiAkhirMhs[I]
END FOR
rata = total/jumlahMhs
```

Gambar 7.12. Notasi Algoritma Proses Perhitungan Nilai Rata-rata pada Array 1D.

7.6. Cara Menampilkan Nilai Elemen Array 1D

Cara menampilkan nilai elemen array satu dimensi tidak jauh berbeda dengan cara pembacaan inputan user. Proses menampilkan nilai elemen array juga dapat dilakukan dengan menggunakan konsep perulangan (looping). Setelah user mengisi sejumlah nilai pada elemen array, memproses elemen tersebut, tahap terakhir merupakan proses menampilkan hasil akhir dari nilai elemen array. Masih sama dengan nilai awal atau sudah berubah karena adanya sejumlah proses perhitungan yang dilakukan pada nilai elemen array. Konsep perulangan (looping) yang paling mudah digunakan adalah menggunakan konsep perulangan bentuk pertama, yaitu FOR. Jika proses menampilkan nilai elemen array dilakukan menggunakan looping FOR, maka hanya membutuhkan 3 baris notasi algoritma. Perhatikan penulisan notasi algoritma untuk menampilkan nilai elemen array satu dimensi pada Gambar 7.12.

```
FOR nilai_awal TO nilai_akhir DO
        PRINT (Nama_Var_Array[indeks_ke])
END FOR
```

Contoh :

```
FOR i = 0 TO 9 DO
        PRINT (nilaiSiswa[i])
END FOR
```

Gambar 7.12. Bentuk Notasi Algoritma untuk Menampilkan Nilai Elemen Array 1D.

Berdasarkan gambar 7.6 dapat dilihat bahwa nilai awal i = 0 digunakan sebagai indeks array yang selalu dimulai dari 0, sampai dengan batas akhir jangkauan = 9. Sehingga apabila diimplementasikan, maka algoritma tersebut akan melakukan proses menampilkan nilai elemen array sebanyak 10 kali.

Di dalam looping for ditambahkan notasi PRINT untuk menampilkan nilai elemen. Menampilkan nilai elemen array dilakukan dengan menyebutkan nama variabel array diikuti dengan indeks. Karena penulisan notasi menggunakan looping FOR, maka untuk penulisan indeks digantikan dengan variabel i sebagai variabel *counter* (penambah satu). Dengan counter tersebut, maka indeks array secara otomatis akan berganti (tambah satu) selama proses perulangan berlangsung atau sampai batas jangkauan akhir sama dengan 9.

7.7 Rangkuman

- Sebuah Array atay Larik merupakan kumpulan dari beberapa nilai yang mempunyai tipe data yang sama. Array disebut juga sebagai Larik/Tabel/Vektor/Variabel Majemuk
- Semua proses yang berkaitan dengan array 1D, paling mudah dilakukan dengan menggunakan konsep looping FOR.
- Array satu dimensi merupakan array yang terdiri dari satu baris banyak kolom atau satu kolom banyak baris
- Bentuk deklarasi array satu dimensi adalah sebagai berikut :
  Var nama_var_array : Array[jangkauan] Of Tipe_Data
- Bentuk pertama proses inisialisasi Nilai Elemen Array adalah sebagai berikut :
  Var nama_var_array : Array[jangkauan] Of Tipe_Data = (nilai_elemen_array)
- Bentuk kedua proses inisialisasi Nilai Elemen Array adalah sebagai berikut :
  Array[indeks] = nilai_elemen_array
- Contoh Pemrosesan Array, yaitu :
  - Proses Aritmatika
  - Pencarian Nilai Maksimal dan Minimal
  - Perhitungan Rata-rata

7.8 Latihan Soal

Selesaikan soal di bawah ini menggunakan **konsep Array Satu Dimensi.**

1. Ibu menyuruh Titi untuk berbelanja sejumlah barang di supermarket. Setibanya di supermarket, ternyata sedang diadakan diskon untuk semua jenis barang. Diskon yang diberikan setiap barang berbeda dengan barang lainnya. Berdasarkan catatan yang dititipkan ibu kepada Titi, Titi membeli sejumlah barang dengan daftar diskon barang sebagai berikut :
   - Sabun → diskon 5%
   - Beras → diskon 7%
   - Gula dan Minyak Goreng → 10%

   Anda diminta untuk membuat flowchart dan pseudocode dengan hasil akhir sebagai berikut :
   a. Daftar barang yang dibeli Titi.
   b. Harga per barang
   c. Diskon per barang
   d. Jumlah beli barang (Qty)
   e. Total per jenis barang
   f. Total Belanja Titi keseluruhan
2. Di kelas Titi terdapat 40 siswa yang akan mengikuti Ujian Nasional. Ujian Nasional akan dilaksanakan dalam 3 hari. Hari pertama siswa akan menghadapi ujian Mata Pelajaran Bahasa Indonesia. Hari kedua untuk Mata Pelajaran Matematika. Sedangkaan hari ketiga untuk Mata Pelajaran Bahasa Inggris. Nilai Akhir Ujian Nasional akan diumumkan satu bulan berikutnya. Untuk mengatasi permasalahan tersebut, Anda diminta untuk membuat flowchart dan pseudocode dengan hasil akhir sebagai berikut :
   a. Nama Siswa Peserta Ujian Nasional.
   b. Nilai Siswa untuk setiap Mata Pelajaran, yaitu Bahasa Indonesia, Matematika, dan Bahasa Inggris.
   c. Nilai Akhir Ujian Nasional setiap Siswa
   d. Rata-rata Nilai Akhir Ujian Nasional dalam 1 kelas tersebut.

Bab 8

Array 2D (Dua Dimensi)

Tujuan :

- ✓ Mahasiswa dapat membedakan penggunaan Array 1D dan Array 2D.
- ✓ Mahasiswa dapat mendemonstrasikan pembuatan algoritma menggunakan konsep Array 2D.

8.1 Pengertian Array 2D

Array atau Larik merupakan kumpulan dari beberapa nilai yang mempunyai tipe data yang sama. Array dapat diberi nama seperti nama variabel biasa. Array disebut juga sebagai Larik/Tabel/Vektor/Variabel Majemuk. Array dibagi menjadi 2 bentuk dimensi, yaitu array satu dimensi dan array dua dimensi. Sebagaimana yang telah dibahas mengenai array satu dimensi pada bab 8, array satu dimensi merupakan array yang terdiri dari satu baris banyak kolom atau satu kolom banyak baris. Sedangkan array dua dimensi merupakan array yang terdiri dari banyak baris dan banyak kolom.

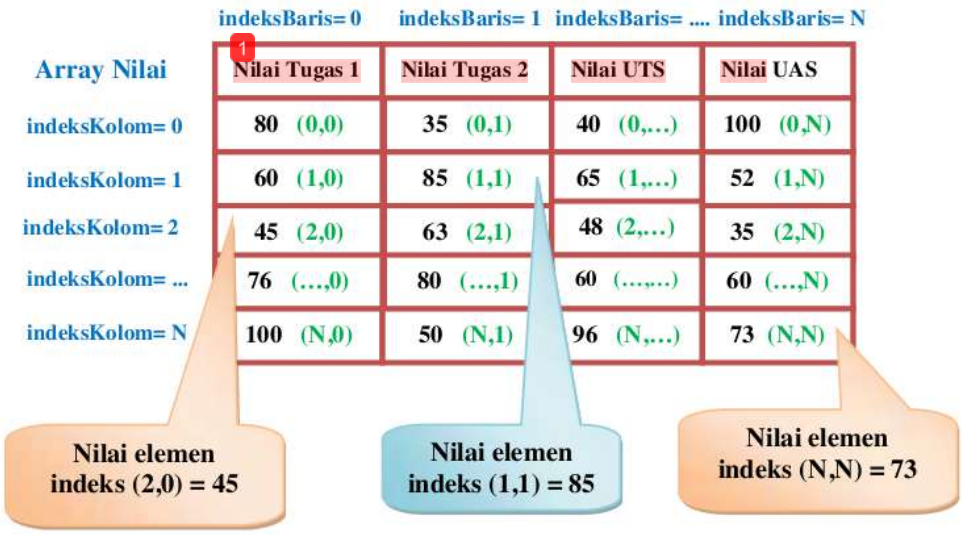
Ada beberapa hal penting yang menjadi ciri-ciri dari sebuah array dua dimensi, antara lain sebagai berikut :

a. Array dua dimensi pasti mempunyai **banyak baris dan banyak kolom.**
b. **Bentuk realita dari aray dua dimensi adalah tabel**, karena sebuah tabel dapat dipastikan mempunyai banyak baris dan banyak kolom untuk menyimpan data.
c. Seperti halnya pada array satu dimensi, nilai yang mengisi setiap elemen array harus spesifik, yaitu mempunyai **tipe data yang sama.**
d. Setiap elemen pada sebuah array dua dimensi juga ditandai dengan sebuah indeks. Indeks tersebut pada array dua dimensi terdiri dari 2 jenis, yaitu **indeks baris dan indeks kolom.** Indeks baris dan indeks kolom pada array dua dimensi juga dimulai dari **angka nol (0).**

e. Jumlah elemen pada array dua dimensi dapat dideklarasikan langsung pada awal program dan **harus lebih kecil atau sama dengan indeks terbesar** baris/kolom.
f. Semua proses yang terjadi pada array dua dimensi, mulai dari proses pengisian nilai elemen array, proses perhitungan, maupun proses penampilan data array dua dimensi dapat menggunakan 2 cara, yaitu melalui proses inisialisasi langsung atau **menggunakan proses looping, yaitu nested for.**

Satu hal penting yang harus diingat dalam pemrosesan array dua dimensi adalah gunakan selalu konsep Nested For pada setiap prosesnya. Mengapa harus menggunakan konsep Nested For ? Karena konsep Nested For merupakan konsep paling sederhana dan mudah untuk melakukan proses apada array dua dimensi yang terdiri dari baris dan kolom. Pada gambar 8.1 di bawah ini akan dijelaskan bentuk dari array dua dimensi.

Banyak Baris banyak Kolom



Gambar 8.1. Bentuk Array Dua Dimensi.

Notes :

Cara paling mudah memahami array dua dimensi adalah dengan menggunakan **Konsep Tabel**. Hal ini dikarenakan **tabel merupakan representasi nyata dari sebuah array dua dimensi** yang terdiri dari 2 komponen penting yaitu mempunyai **banyak baris dan banyak kolom**.

8.2 Deklarasi Array 2D

Sebuah array dua dimensi yang akan digunakan pada sebuah program atau algoritma harus dideklarasikan terlebih dahulu. Hal ini dilakukan dengan tujuan agar variabel array dua dimensi dikenali di semua baris program atau di setiap tahapan algoritma. Adapun aturan pendeklarasian array dua dimensi adalah sebagai berikut :

a. Array dua dimensi dideklarasikan dengan menggunakan **nama variabel**.
b. Setelah diberi nama variabel, diikuti dengan penulisan kata "**Array**".
c. Tambahkan **tanda kurung siku** ([... , ...]) yang **ditulis menggunakan koma** (menandakan array dua dimensi). Jika kurung siku ditulis tanpa koma, maka array tersebut adalah array satu dimensi. Di dalam kurung siku yang pertama (sebelum koma) diisi dengan **jangkauan/jumlah baris**. Sedangkan di dalam kurung siku yang kedua (setelah koma) diisi dengan **jangkauan/jumlah kolom.**
d. Setelah tanda kurung siku, tambahkan pula kata "**Of**" diikuti **tipe data** array.

Perhatikan Gambar 8.2 untuk bentuk deklarasi variabel array satu dimensi di bawah ini.

```
Var  Nama_variabel_array : Array [baris, kolom] Of  Tipe_Data
```

Contoh :

```
Var nilaiSiswa : Array [5, 4] Of  Integer
```

Atau :

```
Var nilaiSiswa : Array [0..4, 0..3] Of  Integer
```

Gambar 8.2. Bentuk Deklarasi Variabel Array Dua Dimensi.

Berdasarkan gambar 8.2, dapat diketahui bahwa nama variabel array dua dimensi adalah nilaiSiswa dengan 5 baris dan 4 kolom, sehingga pada array tersebut mempunyai 20 elemen array. Deklarasi array dua dimensi untuk jumlah baris dan kolom juga dapat ditulis dengan menyebutkan nomor indeks awal baris sampai indeks akhir baris. Begitu juga dengan kolom dapat ditulis dengan menyebutkan nomor indeks awal kolom sampai indeks akhir kolom. Karena indeks array, baik satu dimensi maupun dua dimensi selalu dimulai dari angka nol (0), maka indeks awal baris maupun kolom juga dimulai dari angka nol (0).

Tipe data yang digunakan pada nilai elemen array adalah integer (bilangan bulat). Apabila pada salah satu elemen array diisi dengan nilai bukan bilangan bulat (Real/String), maka hal tersebut akan terbaca sebagai sebuah kesalahan. Contoh lain deklarasi array dua dimensi adalah sebagai berikut :

- Var namaSiswa : Array[3, 2] Of String
- Var karakter : Array[0..2, 0..4] Of Character
- Var nilaiUnas : Array[0..39, 0..2] Of Real
- Var jumlahBarang : Array[100, 5] Of Integer

8.3 Cara Inisialisasi Elemen Array 2D

Setelah melakukan proses deklarasi array dua dimensi, langkah selanjutnya yang harus dilakukan adalah mengisi nilai elemen array dua dimensi. Pengisian nilai elemen array dua dimensi dapat dilakukan dengan 2 cara, diantaranya sebagai berikut :

c. Cara Inisialisasi Nilai Elemen Array Dua Dimensi.
d. Cara Membaca Inputan User.

Cara inisialisasi nilai elemen array dua dimensi dapat dilakukan dengan lebih mudah dibandingkan cara membaca inputan user, meskipun terkesan lebih panjang (sesuai jumlah elemen array). Cara inisialisasi array dua dimensi hampir sama seperti cara inisialisasi array satu dimensi. Hanya saja pada saat deklarasi langsung disebutkan nilai setiap elemen array atau dengan menyebut indeks baris dan kolom setiap elemen. Namun, jika menggunakan cara pembacaan inputan user, harus menggunakan konsep perulangan (looping – Nested For). Adapun aturan yang harus diperhatikan dalam proses inisialisasi nilai elemen array dua dimensi, diantaranya sebagai berikut :

1. **Langsung menyebutkan nilai elemen array pada saat deklarasi.**
   (Lihat Gambar 8.3)

Var Nama_variabel_array : Array [baris, kolom] Of Tipe_Data =
{(nilai_elemen_baris_pertama), (nilai_elemen_baris_kedua), (nilai_elemen_baris_keN)}

Contoh :

Var nilaiSiswa : Array [2,3] Of Integer = {(30, 95, 75), (40, 100, 80)}

Gambar 8.3. Bentuk Pertama Inisialisai Nilai Elemen Array Dua Dimensi.

2. **Pemberian nilai elemen array dengan menyebutkan indeks elemennya.**

(Lihat Gambar 8.4)

```
Nama_variabel_array [indeksBaris, indeksKolom] = nilai_elemen_array
```

Contoh :

```
nilaiSiswa[0,0] = 30        nilaiSiswa[1,0] = 40
nilaiSiswa[0,1] = 95        nilaiSiswa[1,1] = 100
nilaiSiswa[0,2] = 75        nilaiSiswa[1,2] = 80
```

Gambar 8.4. Bentuk Kedua Inisialisasi Nilai Elemen Array Dua Dimensi.

8.4 Cara Pemrosesan Array 2D

Pemrosesan nilai elemen array dua dimensi tidak jauh berbeda dengan pemrosesan pada array satu dimensi. **Satu hal yang penting dan harus diingat pada pemrosesan array dua dimensi adalah, selalu gunakan konsep perulangan (looping) bentuk NESTED FOR.**

a. Cara Menginputkan Nilai Elemen Array 2D.

Konsep perulangan (looping) yang paling mudah digunakan adalah menggunakan konsep perulangan bentuk keempat, yaitu NESTED FOR. Jika pengisian nilai elemen array dilakukan menggunakan looping NESTED FOR, maka hanya membutuhkan 5 baris notasi algoritma. Perhatikan penulisan notasi algoritma untuk pembacaan inputan user array dua dimensi pada Gambar 8.5.

```
FOR baris_awal TO baris_akhir DO
        FOR kolom_awal TO kolom_akhir DO
                INPUT (Nama_Var_Array[barisKe, kolomKe]
        END FOR
END FOR
```

Contoh :

```
FOR baris = 0 TO baris = 9 DO
        FOR kolom = 0 TO kolom = 3 DO
                INPUT(nilaiSiswa[baris, kolom]
        END FOR
END FOR
```

Gambar 8.5. Bentuk Notasi Algoritma untuk Input Nilai Elemen Array Dua Dimensi.

b. Proses Aritmatika pada Array 2D.

```
FOR baris_awal TO baris_akhir DO
  FOR kolom_awal TO kolom_akhir DO
     array_hasil[baris, kolom] = arrayOperand1[baris, kolom] * Operand2
  END FOR
END FOR
```

Contoh :

```
FOR baris = 0 TO baris = 9 DO
   FOR kolom = 0 TO kolom = 2 DO
                nilaiTugasAkhir[baris, 3] = nilaiTugas[baris, 0] * 0.4
    END FOR
END FOR
```

```
FOR baris = 0 TO baris = 9 DO
   FOR kolom = 0 TO kolom = 2 DO
     nilaiTugasAkhir[baris, 2] = (nilaiTugas[baris, 0] * 0.4) + (nilaiTugas[baris, 1] * 0.6)
   END FOR
END FOR
```

Gambar 8.6. Notasi Algoritma untuk Proses Aritmatika Perkalian Pada Array 2D.

c. Proses Nilai Maksimal dan Minimal.

Mencari Nilai Maksimal/Terbesar

```
maks = nama_array[baris, kolom]
FOR baris_awal TO baris_akhir DO
  FOR kolom_awal TO kolom_akhir DO
       if (nama_array[baris, kolom] > maks) then
            maks = nama_array[baris, kolom]
       end if
  END FOR
END FOR
```

Mencari Nilai Minimal/Terkecil

```
min = nama_array[baris, kolom]
FOR baris_awal TO baris_akhir DO
  FOR kolom_awal TO kolom_akhir DO
       if (nama_array[baris, kolom] < min) then
            min = nama_array[baris, kolom]
       end if
  END FOR
END FOR
```

Gambar 8.7. Notasi Algoritma Mencari Nilai Maksimal dan Minimal pada Array 2D.

Pada gambar 8.8 dapat dilihat contoh mencari nilai terbesar dan terkecil.

Contoh Mencari Nilai Maksimal/Terbesar

```
maks = nilaiAkhirMhs[0, 0]

FOR baris = 0 TO baris = 9 DO
   FOR kolom = 0 kolom = 3 DO
         if (nilaiAkhirMhs[baris+1, kolom+1] > maks) then
               maks = nilaiAkhirMhs[baris+1, kolom+1]
         end if
    END FOR
END FOR
```

Contoh Mencari Nilai Minimal/Terkecil

```
min = nilaiAkhirMhs[0, 0]

FOR baris = 0 TO baris = 9 DO
   FOR kolom = 0 kolom = 3 DO
         if (nilaiAkhirMhs[baris+1, kolom+1] < min) then
               min = nilaiAkhirMhs[baris+1, kolom+1]
         end if
    END FOR
END FOR
```

Gambar 8.8. Contoh Mencari Nilai Maksimal dan Minimal pada Array 2D.

d. Proses Perhitungan Nilai Rata-rata.

```
total = 0
FOR baris_awal TO baris_akhir DO
   FOR kolom_awal TO kolom_akhir DO
         total = total + nama_array[baris][kolom]
END FOR
rata = total/N
```

Contoh :

```
total = 0
FOR baris=0 TO baris=9 DO
   FOR kolom=0 TO kolom=2 DO
         total = total + nilaiAkhirMhs[baris][kolom]
END FOR
rata = total/(jumlahMhs*3)
```

Gambar 8.9. Notasi Algoritma Proses Perhitungan Nilai Rata-rata pada Array 2D.

8.5 Cara Menampilkan Nilai Elemen Array 2D.

```
FOR baris_awal TO baris_akhir DO
  FOR kolom_awal TO kolom_akhir DO
      PRINT (Nama_Var_Array[baris][kolom])
  END FOR
END FOR
```

Contoh :

```
FOR baris=0 TO baris=9 DO
  FOR kolom=0 TO kolom-3 DO
      PRINT( nilaiAkhirMhs[baris][kolom])
  END FOR
END FOR
```

Gambar 8.10. Bentuk Notasi Algoritma untuk Menampilkan Nilai Elemen Array 2D.

8.6 Rangkuman

- Semua proses yang berkaitan dengan array 2D, paling mudah dilakukan dengan menggunakan konsep looping NESTED FOR.
- Array dua dimensi merupakan array yang terdiri dari banyak baris dan banyak kolom.
- Bentuk deklarasi array dua dimensi adalah sebagai berikut :
  Var nama_var_array : Array[baris, kolom] Of Tipe_Data
- Bentuk pertama proses inisialisasi Nilai Elemen Array adalah sebagai berikut :**Var nama_var_array : Array[baris, kolom] Of Tipe_Data = {(nilai_elemen_baris_pertama), (nilai_elemen_baris_kedua), (nilai_elemen_ baris_keN)}**
- Bentuk kedua proses inisialisasi Nilai Elemen Array adalah sebagai berikut :
  Array[indeksBaris, indeksKolom] = nilai_elemen_array.

8.7 Latihan Soal

Selesaikan soal di bawah ini menggunakan **gabungan konsep Array 1D dan 2D.**

1. Ibu menyuruh Titi untuk berbelanja sejumlah barang di supermarket. Setibanya di supermarket, ternyata sedang diadakan diskon untuk semua jenis barang. Diskon yang diberikan setiap barang berbeda dengan barang lainnya. Berdasarkan catatan yang dititipkan ibu kepada Titi, Titi membeli sejumlah barang dengan daftar diskon barang sebagai berikut :

- Sabun → diskon 5%
- Beras → diskon 7%
- Gula dan Minyak Goreng → 10%

Anda diminta untuk membuat flowchart dan pseudocode dengan hasil akhir sebagai berikut :

a. Daftar barang yang dibeli Titi.
b. Harga per barang
c. Diskon per barang
d. Jumlah beli barang (Qty)
e. Total per jenis barang
f. Total Belanja Titi keseluruhan

2. Di kelas Titi terdapat 40 siswa yang akan mengikuti Ujian Nasional. Ujian Nasional akan dilaksanakan dalam 3 hari. Hari pertama siswa akan menghadapi ujian Mata Pelajaran Bahasa Indonesia. Hari kedua untuk Mata Pelajaran Matematika. Sedangkaan hari ketiga untuk Mata Pelajaran Bahasa Inggris. Nilai Akhir Ujian Nasional akan diumumkan satu bulan berikutnya. Untuk mengatasi permasalahan tersebut, Anda diminta untuk membuat flowchart dan pseudocode dengan hasil akhir sebagai berikut :

a. Nama Siswa Peserta Ujian Nasional.
b. Nilai Siswa untuk setiap Mata Pelajaran, yaitu Bahasa Indonesia, Matematika, dan Bahasa Inggris.
c. Nilai Akhir Ujian Nasional setiap Siswa.
d. Rata-rata Nilai Akhir Ujian Nasional dalam 1 kelas tersebut.

Bab 9

Sub Program (Metode/Modular)

Tujuan :

- ✓ Mahasiswa dapat membedakan penggunaan Fungsi dan Prosedur.
- ✓ Mahasiswa dapat menunjukkan pembuatan algoritma menggunakan konsep Fungsi dan Prosedur.

9.1 Pengantar Sub Program

Pada umumnya sebuah program yang dibuat oleh programmer hanya terdiri dari program utama saja *(main program)*. Main program ini merupakan syarat utama apabila program yang dibuat dapat dijalankan dengan baik dan benar. Dengan rumitnya jenis permasalahan yang muncul, maka solusi program yang dibuat juga semakin banyak. Hal ini menyebabkan semakin banyak baris yang ditulis pada bagian main program.

Untuk mengatasi hal tersebut, maka dipecahlah kode program yang ditulis pada main program menjadi program-program kecil, tanpa mengubah tugas utamanya. Program-program kecil inilah yang selanjutnya **disebut sebagai Sub Program.** Sub Program merupakan bagian dari program utama yang dapat berdiri sendiri. Kode program yang ditulis pada sub program dapat dijalankan, apabila sub program tersebut dipanggil pada program utama. Namun sebaliknya, apabila telah membuat sebuah sub program, namun tidak dipanggil pada program utama, maka sub program tersebut tidak akan pernah dijalankan.

Beberapa manfaat yang dapat diambil apabila menggunakan sub program adalah sebagai berikut:

1. Simplifikasi

   Untuk memudahkan programmer, jika ada bagian pada program utama yang ditulis berulang. Cukup membuat sebuah sub program dan lakukan pemanggilan sub program tersebut pada baris program utama yang diinginkan.

2. Modularisasi

   Lebih mudah dalam mengolah dan menganalisis program (dalam mengoreksi, memodifikasi dan menulis ulang program).

Beberapa pemrograman sering menyebut sub program sebagai Teknik Pemrograman Modular/Routine. Pada dasarnya sub program dibagi menjadi 2 jenis, yaitu sebagai berikut :

1. **Prosedur** ***(Procedure)***

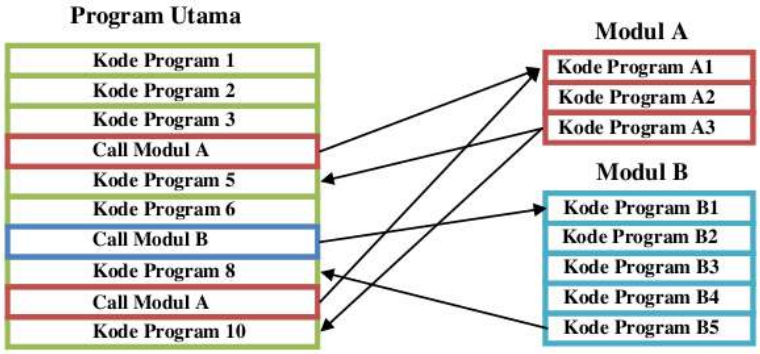
   Prosedur adalah bagian dari suatu program yang disusun secara terpisah untuk melakukan suatu tugas khusus/fungsi tertentu, tanpa mengembalikan nilai dari hasil proses tersebut. Pada prosedur diijinkan melakukan penginputan dan pencetakan hasil output dari sebuah proses, misalnya mencetak hasil penjumlahan dari proses aritmatika.

2. **Fungsi** ***(Function)***

   Fungsi adalah bagian dari program yang dibuat terpisah untuk melaksanakan fungsi tertentu yang menghasilkan suatu nilai untuk dikembalikan ke program utama *(return)*. Pada fungsi tidak diijinkan melakukan penginputan dan pencetakan hasil output dari sebuah proses, misalnya mencetak hasil penjumlahan dari proses aritmatika. Fungsi hanya bertugas sebagai tempat penampungan nilai sementara. Jika nilai tersebut hendak dicetak, maka cetaknya pada program utama, bukan pada fungsi itu sendiri.

9.2 Cara Kerja Sub Program

Sub program dapat dijalankan apabila dipanggil melalui program utama. Program Utama yang dibuat oleh seorang programmer berfungsi untuk menggabungkan satu sub program atau lebih agar dapat dapat dijalankan dengan baik dan benar. Perhatikan bagan cara kerja sub program pada Gambar 9.1 di bawah ini.



Gambar 9.1. Bagan Cara Kerja Sub Program.

Berdasarkan Gambar 9.1 Bagan Cara Kerja Sub Program, terdapat sebuah program utama yang terdiri dari 10 baris kode program dan 2 sub program (modul) yaitu Modul A dan Modul B. Modul A hanya memiliki 3 baris kode program, yaitu Kode Program A1, A2, dan A3. Sedangkan Modul B memiliki 5 baris kode program, yaitu Kode Program B1, B2, B3, B4, dan B5. Perhatikan kembali kode program pada Program Utama, tidak tertulis kode program 4, 7, dan 9. Pada ketiga baris tersebut terjadi Pemanggilan Sub Program atau Modul. **Cara memanggil Sub Program/Modul pada Program Utama dilakukan dengan notasi CALL[spasi]NamaSubProgram.** Kode Program 4 diganti dengan memanggil Modul A, kode program 7 diganti dengan memanggil Modul B, dan kode program 9 diganti dengan mengulangi pemanggilan modul A kembali.

Program utama dijalankan mulai dari Kode Program 1, Kode Program 2, dan Kode 3, dan diilanjutkan dengan CALL Modul A. Pada saat CALL ModulA, program akan menjalankan kode program A1, A2, dan A3 terlebih dahulu secara berurutan. Setelah kode program A3 selesai dijalankan, maka kembali meneruskan alur program dengan menjalankan kode program 5, kode program 6, dan dilanjutkan dengan CALL Modul B. Pada saat CALL Modul B, program akan menjalankan kode program B1, B2, B3, B4 dan B5 terlebih dahulu secara berurutan. Setelah kode program B5 selesai dijalankan, maka kembali meneruskan alur program dengan menjalankan kode program 8, dan dilanjutkan dengan memanggil kembali CALL Modul A. Pada saat CALL ModulA, program akan menjalankan kode program A1, A2, dan A3 terlebih dahulu secara berurutan. Setelah kode program A3 selesai dijalankan, maka kembali meneruskan menjalankan kode program 10, dan selesai sudah program utama dijalankan. ☺

9.3 Prosedur

Prosedur adalah bagian dari suatu program yang disusun secara terpisah untuk melakukan suatu tugas khusus/fungsi tertentu, tanpa mengembalikan nilai dari hasil proses tersebut. Pada prosedur diijinkan melakukan penginputan dan pencetakan hasil output dari sebuah proses, misalnya mencetak hasil penjumlahan dari proses aritmatika.

Ketika suatu prosedur dipanggil, maka pada hakikatnya dapat melakukan proses pertukaran data antara program utama dan prosedur. Proses pertukaran ini dilakukan melalui parameter. Pada prosedur, terdapat 2 jenis parameter yaitu sebagai berikut :

a. Parameter Aktual

Parameter aktual merupakan parameter yang disertakan pada saat prosedur dipangggil untuk dijalankan pada program utama, sering disebut sebagai ***argumen***.

b. Parameter Formal

Parameter formal merupakan parameter yang dituliskan pada definisi suatu prosedur/fungsi. Ada 3 jenis dari parameter formal yang dapat digunakan pada prosedur, yaitu :

1. **Parameter Masukan (Input)**

   Parameter yang menerima nilai dari parameter aktual.

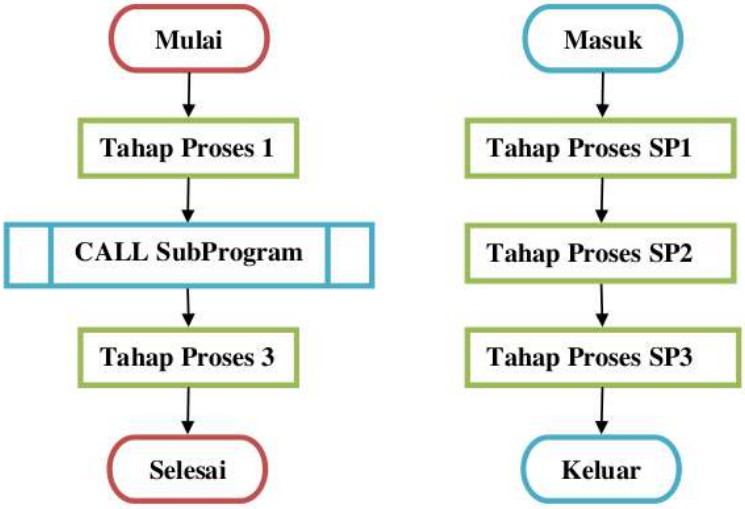
2. **Parameter Keluaran (Output)**

   Parameter yang menyerahkan nilai ke parameter aktual.

3. **Parameter masukan dan keluaran (Input-Output)**

   Parameter yang menerima nilai dari parameter aktual untuk diproses dalam prosedur kemudian diserahkan kembali ke parameter aktual setelah selesai.

Adapun bentuk flowchart dan cara penulisan pseudocode untuk Prosedur/Sub Program dapat dilihat pada Gambar 9.2 dan 9.3.



Gambar 9.2. Bentuk Flowchart Prosedur/Sub Program.

Perhatikan Gambar 9.2 bentuk flowchart untuk Sub Program, pada saat pemanggilan Sub Program, simbol flowchart yang digunakan berbeda, tidak seperti simbol flowchart untuk proses pada umunya. Simbol flowchart untuk Sub Program mempunyai bentuk persegi panjang yang mempunyai tambahan kotak pada samping kiri dan kanan. Ingat Bab 3. ☺

ALGORITMA UTAMA
{Keterangan Judul_Algoritma_Utama}

Deklarasi :
(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :
(Tulis setiap tahapan/proses yang dilakukan di dalam algoritma, mulai dari awal sampai akhir. Sesuaikan penulisan deskripsi dengan apa yang ditulis di dalam symbol flowchart atau sesuaikan dengan penulisan notasi algoritmik. ***Ingat, panggil NamaProsedur pada algoritma utama ini.****)*

PROCEDURE NamaProsedur()
{Keterangan Judul Prosedur}

Deklarasi :
(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :
(Tulis setiap tahapan/proses yang dilakukan di dalam algoritma, mulai dari awal sampai akhir. Sesuaikan penulisan deskripsi dengan apa yang ditulis di dalam simbol flowchart atau sesuaikan dengan penulisan notasi algoritmik.)

Gambar 9.3. Struktur Penulisan Pseudocode untuk Sub Program.

Contoh : Algoritma Menghitung Luas Persegi Panjang Menggunakan Prosedur Tanpa Parameter.

Algoritma Utama :

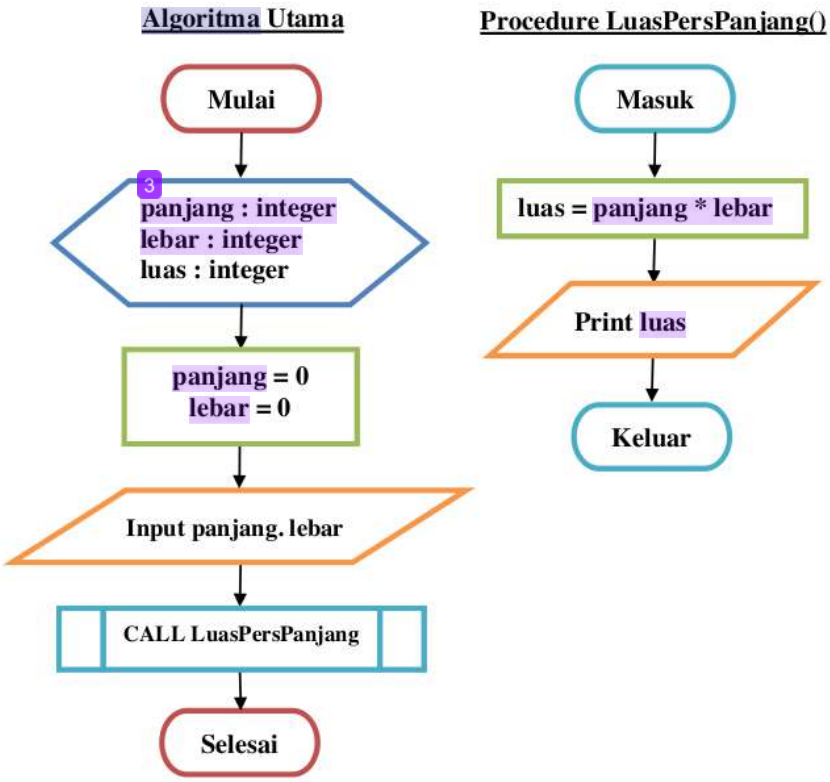
Input : panjang, lebar

Panggil Procedure LuasPersPanjang()

Procedure LuasPersPanjang :

Rumus Luas Persegi Panjang = panjang x lebar

Output : Hasil Perhitungan Luas Persegi Panjang




```
ALGORITMA UTAMA
{Algoritma Menghitung Luas Persegi Panjang, panjang dan lebar diinputkan
terlebih dahulu}

Deklarasi :
        panjang, lebar, luas : Integer
Deskripsi :
        Panjang = 0
        Lebar = 0
        Print (panjang, lebar)
        Procedure LuasPersPanjang()

PROCEDURE LuasPersPanjang()
{Menghitung Luas Persegi Panjang}

Deklarasi :
        -   (tidak ada)
Deskripsi :
        luas = panjang * lebar
        Print (luas)
```

Gambar 9.4. Flowchart dan Pseudocode Menghitung Luas Persegi Panjang Menggunakan Prosedur Tanpa Parameter.

9.3.1 Prosedur dengan Parameter Masukan

Prosedur dengan menggunakan parameter masukan merupakan prosedur yang menggunakan parameter dengan isi variabel berasal dari inputan user. Inputan isi variabel dilakukan pada algoritma utama. Setelah inputan user tersebut disimpan dalam sebuah variabel, maka saat memanggil prosedur yang dibutuhkan, gunakan variabel inputan tersebut sebagai parameternya. Sedangkan proses inti dan hasil output dilakukan di dalam prosedur.

Contoh : Algoritma Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Masukan (Inputan).

Algoritma Utama :

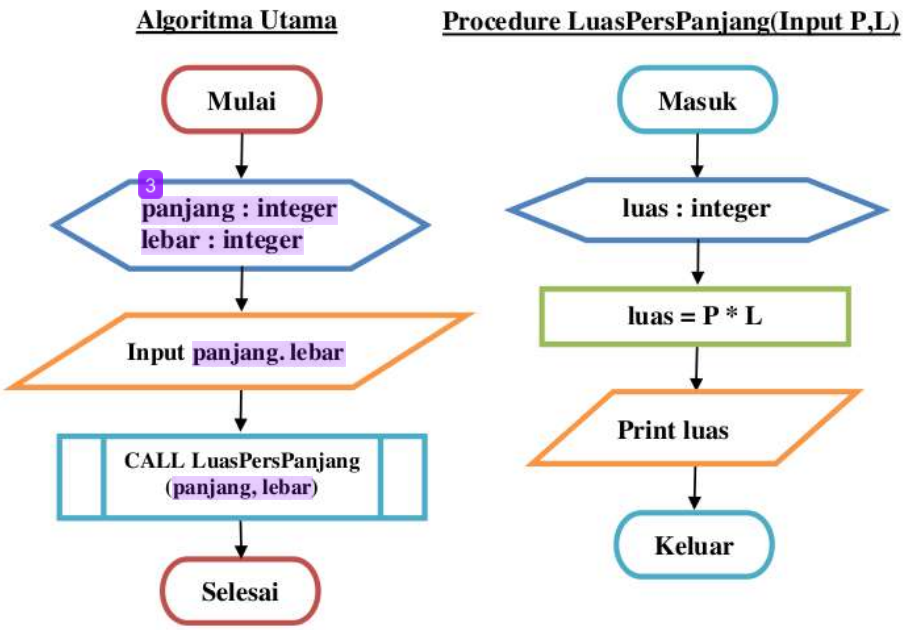
Input : panjang, lebar

Panggil Procedure LuasPersPanjang(Input P, L)

Procedure LuasPersPanjang (Input P, L) :

Rumus Luas Persegi Panjang = P x L

Output : Hasil Perhitungan Luas Persegi Panjang



Gambar 9.5. Flowchart Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Masukan.

```
ALGORITMA UTAMA
{Algoritma Menghitung Luas Persegi Panjang, panjang dan lebar
diinputkan terlebih dahulu}

Deklarasi :
        panjang, lebar: Integer
        Procedure  LuasPersPanjang (Input P, L : integer)
Deskripsi :
        Input (panjang, lebar)
        LuasPersPanjang(panjang, lebar)

PROCEDURE LuasPersPanjang (Input P, L:integer)
{Menghitung Luas Persegi Panjang}

Deklarasi :
        luas : integer
Deskripsi :
        luas = P * L
        Print (luas)
```

Gambar 9.6. Pseudocode Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Masukan.

9.3.2 Prosedur dengan Parameter Keluaran

Prosedur dengan menggunakan parameter keluaran merupakan prosedur yang menggunakan parameter dengan hasil output dari sebuah proses perhitungan. Proses perhitungan dilakukan pada prosedur. Setelah dilakukan proses perhitungan tersebut disimpan dalam sebuah variabel, maka saat memanggil prosedur yang dibutuhkan, gunakan variabel output tersebut sebagai parameternya. Sedangkan proses inti dilakukan di dalam prosedur.

Contoh : Algoritma Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Keluaran (Output).

Algoritma Utama :

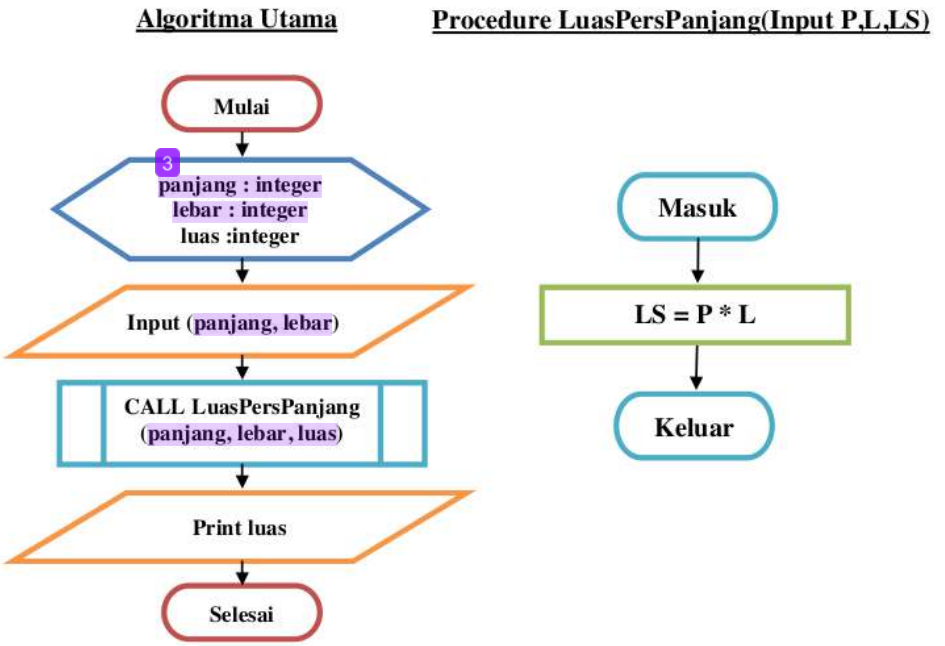
Input : panjang, lebar

Panggil Procedure LuasPersPanjang(Input P, L : integer; Output LS:integer)

Procedure LuasPersPanjang (Input P, L :integer; Output LS :integer) :

Rumus Luas Persegi Panjang = P x L

Output : Hasil Perhitungan Luas Persegi Panjang



Gambar 9.7. Flowchart Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Keluaran.

```
ALGORITMA UTAMA
{Algoritma Menghitung Luas Persegi Panjang, panjang dan lebar diinputkan
terlebih dahulu}

Deklarasi :
        panjang, lebar, luas: Integer
        Procedure  LuasPersPanjang(Input P,L : integer, Output LS : integer)
Deskripsi :
        Input (panjang, lebar)
        LuasPersPanjang(panjang, lebar, luas)
        Print (luas)

PROCEDURE LuasPersPanjang(Input P, L:integer; Output LS : integer)
{Menghitung Luas Persegi Panjang}
Deklarasi :
        -   (tidak ada)
Deskripsi :
        LS = P * L
```

Gambar 9.8. Pseudocode Menghitung Luas Persegi Panjang Menggunakan Prosedur Dengan Parameter Keluaran.

9.3.3 Prosedur dengan Parameter Masukan&Keluaran

Prosedur dengan menggunakan parameter masukan & keluaran merupakan prosedur yang menggunakan parameter dengan menggabungkan inputan user dan hasil output dari sebuah proses pada variabel yang sama. Artinya, variabel yang digunakan sebagai parameter inputan, juga akan digunakan sebagai penampung hasil output algoritma. Sehingga, apabila pada prosedur menggunakan 2 buah variabel sebagai parameter inputan, maka prosedur tersebut juga harus mempunyai minimal 1 hasil output.

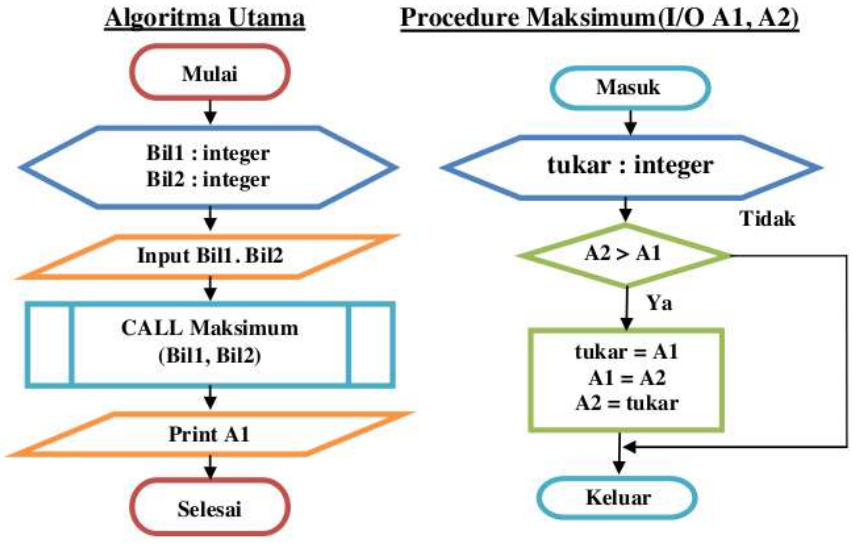
Contoh : Algoritma Menentukan Nilai Terbesar dari 2 buah bilangan menggunakan Prosedur Dengan Parameter Masukan/Keluaran (Input/Output).

Algoritma Utama :

```
Input : Bil1, Bil2
Panggil Procedure Maksimum(I/O A1, A2 : integer)
```

Procedure Maksimum (I/O A1, A2 : integer) :

```
Mencari Angka Terbesar : if (A2>A1) then
                             tukar = A1
                             A1 = A2
                             A2 = tukar
                         end if
```



Gambar 9.9. Flowchart Menentukan Bilangan Terbesar Menggunakan Prosedur Dengan Parameter Masukan/Keluaran.

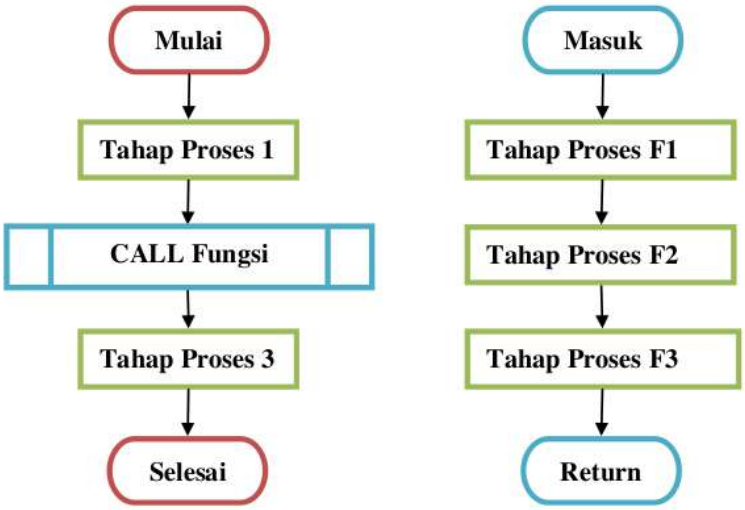
```
ALGORITMA UTAMA
{Algoritma Menentukan Nilai Terbesar, 2 buah bilangan diinputkan terlebih
dahulu}

Deklarasi :
        Bil1, Bil2: Integer
        Procedure  Maksimum(I/O A1, A2 : Integer)
Deskripsi :
        Input (Bil1, Bil2)
        Maksimum(Bil1, Bil2)
        Print (A1)

PROCEDURE Maksimum(I/O A1, A2 : Integer)
{Menentukan Bilangan Terbesar}
Deklarasi :
        tukar : integer
Deskripsi :
        if (A2>A1) then
                tukar = A1
                A1 = A2
                A2 = tukar
        end if
```

Gambar 9.10. Pseudocode Menentukan Bilangan Terbesar Menggunakan Prosedur Dengan Parameter Masukan/Keluaran.

9.4 Fungsi

Fungsi sebenarnya sama dengan prosedur, yaitu merupakan bagian dari program (sub program) yang ditulis terpisah dari program utama. Fungsi adalah sub program yang melakukan suatu tugas untuk dapat mengembalikan nilai dari sebuah hasil proses. Berbeda dengan prosedur, pada fungsi tidak diijinkan melakukan penginputan dan pencetakan hasil output dari sebuah proses, misalnya mencetak hasil penjumlahan dari proses aritmatika.

Ketika sebuah fungsi dipanggil, maka pada hakikatnya fungsi tersebut melakukan proses pengembalian nilai yang didapatkan dari proses perhitungan aritmatika. Pada saat pemanggilan fungsi, harus disertai dengan penggunaan Print(). Hal ini dilakukan karena fungsi sebenarnya hanya bertugas sebagai tempat penyimpanan nilai sementara. Sehingga apabila nilai tersebut tidak dipanggil menggunakan print(), maka nilai tidak akan pernah ditampilkan. Sama seperti halnya prosedur, pada fungsi juga dapat ditambahkan parameter, baik parameter aktual maupun parameter formal. Adapun bentuk flowchart dan cara penulisan pseudocode untuk Fungsi dapat dilihat pada Gambar 9.11 dan 9.12.



Gambar 9.11. Bentuk Flowchart Fungsi.

ALGORITMA UTAMA
{Keterangan Judul_Algoritma_Utama}

Deklarasi :
(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :
(Tulis setiap tahapan/proses yang dilakukan di dalam algoritma, mulai dari awal sampai akhir. Sesuaikan penulisan deskripsi dengan apa yang ditulis di dalam simbol flowchart atau sesuaikan dengan penulisan notasi algoritmik. ***Ingat, panggil NamaFungsi pada algoritma utama ini.*** *)*

FUNGSI NamaFungsi()
{Keterangan Judul Fungsi}

Deklarasi :
(Tulis semua variabel dan konstanta yang dibutuhkan, termasuk Tipe data variabel dan konstanta)

Deskripsi :
(Tulis setiap tahapan/proses yang dilakukan di dalam algoritma, mulai dari awal sampai akhir. Sesuaikan penulisan deskripsi dengan apa yang ditulis di dalam simbol flowchart atau sesuaikan dengan penulisan notasi algoritmik.)

Gambar 9.12. Struktur Penulisan Pseudocode untuk Sub Program.

Contoh : Algoritma Menghitung Luas Persegi Panjang Menggunakan Fungsi Dengan Parameter Masukan.

Algoritma Utama :

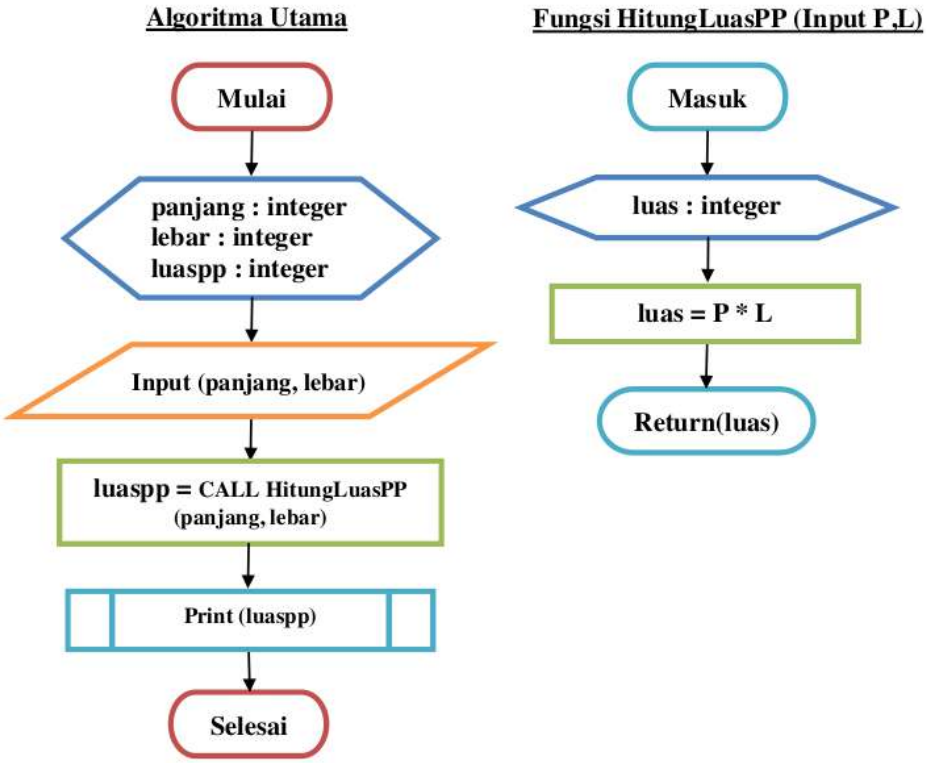
Input : panjang, lebar

Panggil Fungsi HitungLuasPP()

Fungsi HitungLuasPP :

Rumus Luas Persegi Panjang = panjang x lebar

Output : Return (Luas)



Gambar 9.13. Flowchart Menghitung Luas Persegi Panjang Menggunakan Fungsi Dengan Parameter Masukan.

```
ALGORITMA UTAMA
{Algoritma Menghitung Luas Persegi Panjang, panjang dan lebar
diinputkan terlebih dahulu}

Deklarasi :
        panjang, lebar, luaspp: Integer
        Fungsi  HitungLuasPP(Input P,L : integer)
Deskripsi :
        Input (panjang, lebar)
        luaspp = HitungLuasPP(panjang, lebar)
        Print (luaspp)

FUNGSI HitungLuasPP(Input P, L:integer)
{Menghitung Luas Persegi Panjang}

Deklarasi :
        luas : integer
Deskripsi :
        luas = P * L
        Return (luas)
```

Gambar 9.14. Pseudocode Menghitung Luas Persegi Panjang Menggunakan Fungsi Dengan Parameter Masukan.

9.5 Rangkuman

- **Sub Program merupakan** bagian dari program yang ditulis terpisah dari program utama. Sub program dapat dijalankan apabila dipanggil melalui program utama. Program Utama yang dibuat oleh seorang programmer berfungsi untuk menggabungkan satu sub program atau lebih agar dapat dapat dijalankan dengan baik dan benar.
- Sub Program dibagi menjadi 2 yaitu **Prosedur dan Fungsi.**
- **Prosedur adalah** bagian dari suatu program yang disusun secara terpisah untuk melakukan suatu tugas khusus/fungsi tertentu, tanpa mengembalikan nilai dari hasil proses tersebut.
- **Fungsi adalah** bagian dari program yang dibuat terpisah untuk melaksanakan fungsi tertentu yang menghasilkan suatu nilai untuk dikembalikan ke program utama (return).
- Ada 3 jenis parameter formal yang dapat digunakan pada sub program, yaitu **Parameter Masukan, Parameter Keluaran, serta Parameter Masukan dan Keluaran (I/O).**

9.6 Latihan Soal

Selesaikan soal di bawah ini menggunakan **konsep Prosedur dan Fungsi.**

1. Menghitung Keliling dan Luas Lingkaran dengan inputan jari-jari dan konstanta phi = 3.14. Buat Flowchart dan Pseudocode menggunakan **konsep Prosedur dengan 2 macam parameter yaitu parameter masukan dan keluaran.**
2. Menentukan urutan tinggi badan dari 3 orang dengan inputan tinggi badan dari masing-masing orang tersebut. Buat Flowchart dan Pseudocode menggunakan **konsep Prosedur dengan parameter masukan dan keluaran (I/O).**
3. Mengitung nilai rata2 nilai dari sejumlah siswa yang diinputan sesuai dengan permintaan user. Simpan nilai tersebut pada array satu dimensi, kemudian hitung rata-rata nilainya. Buat Flowchart dan Pseudocodenya menggunakan **konsep Fungsi dengan parameter masukan.**

Bab 10

Stack dan Queue

Tujuan :

- ✓ Mahasiswa dapat membedakan penggunaan Stack dan Queue.

10.1 Pengertian Stack

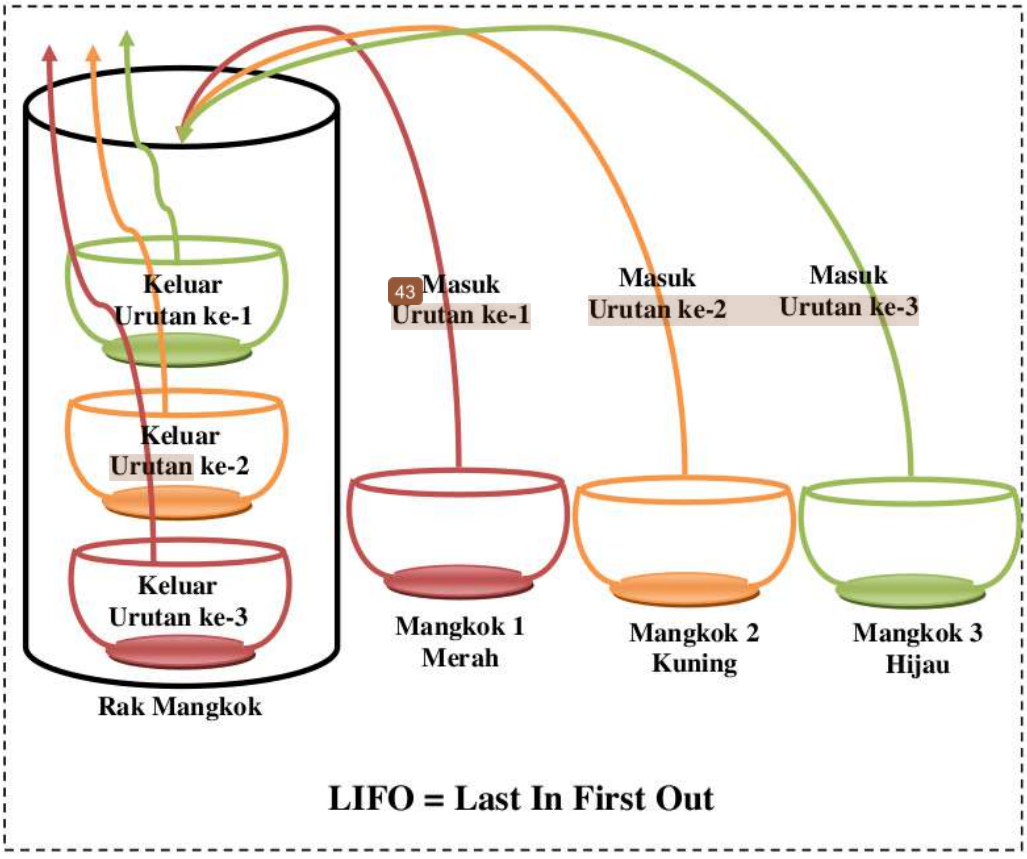
Stack dalam terjemahan Bahasa Indonesia dapat diartikan sebagai tumpukan. Tumpukan berarti sejumlah barang yang disusun dari bawah ke atas, secara umum susunan tersebut dapat diartikan sebagai garis yang sifatnya vertikal. Contoh nyata dari tumpukan dapat dilihat dari benda-benda yang ada disekitar kita dan sering dijumpai sehari-hari, diantaranya :

a. Pakaian dalam lemari yang dilipat kemudian ditumpuk.
b. Buku yang ditumpuk di perpustakaan pada saat pengembalian buku.
c. Piring yang ditumpuk saat penyajian secara prasmanan pada pesta pernikahan.
d. Mangkok sup yang juga ditumpuk saat penyajian secara prasmanan pada pesta pernikahan.

Masih banyak lagi contoh nyata dari tumpukan berbagai benda yang dapat dibayangkan sebagai sebuah Stack. Pada dasarnya cara kerja stack juga mengacu pada tumpukan benda-benda tersebut. Misalnya tumpukan mangkok sup. Tumpukan mangkok sup dapat terjadi apabila pada saat pertama kali diawali dengan 1 mangkok sup yang diletakkan di bagian paling bawah. Disusul mangkok ke-2 diletakkan diatas mangkok pertama, mangkok ke-3 diletakkan diatas mangkok ke-2, mangkok ke-4 diletakkan diatas mangkok ke-3, dan seterusnya. Menyusun mangkok secara demikian dapat menyebabkan terjadinya tumpukan. Setelah membentuk tumpukan, mau tidak mau pada saat pengambilan mangkok, seseorang harus mengambil mangkok yang posisnya paling atas. Tidak mungkin mangkok yang berada diposisi tengah atau yang berada di posisi paling bawah diambil terlebih dahulu. Jika hal

tersebut terjadi, maka kemungkinan akan ada mangkok yang pecah, karena diambil tidak sesuai dengan urutannya yang paling atas.

Berdasarkan contoh cerita diatas, dapat disimpulkan bahwa tumpukan dapat terjadi apabila terdapat sebuah benda yang diletakkan pada bagian bawah, kemudian disusul benda yang lainnya yang diletakkan diatasnya. Jika ingin mengambil salah satu benda tersebut, ambilan benda pada posisi bagian atas terlebih dahulu. Konsep seperti ini disebut dengan konsep LIFO. **LIFO adalah kepanjangan dari *Last In First Out.*** Benda yang diletakkan/dimasukkan terakhir kali, maka benda itulah yang akan diambil/dikeluarkan terlebih dahulu. Adapun bagan cara kerja Stack dapat dilihat pada Gambar 10.1 dibawah ini.



Gambar 10.1. Bagan Cara Kerja Stack.

Perhatikan gambar 10.1. Pada Bagan Cara Kerja Stack disediakan 1 rak mangkok besar berbentuk tabung dan di sebelahnya terdapat 3 buah mangkok masing-masing berwarna merah, kuning, dan hijau. Pada saat melakukan proses pengisian rak kosong, mangkok

pertama yang berwarna merah merupakan mangkok pertama yang dimasukkan terlebih dahulu pada rak tabung. Sehingga posisi mangkok warna merah berada pada urutan paling bawah. Dilanjutkan dengan memasukkan mangkok kedua berwarna kuning ke dalam rak tabung, diletakkan diatas mangkok merah. Sehingga mangkok kuning berada diurutan ke-2. Kemudian mangkok yang terakhir berwarna hijau dimasukkan ke dalam rak tabung, sehingga diletakkan diatas rak kuning. Posisinya terdapat pada urutan ke-3 atau yang paling atas.

Setelah selesai memasukkan setiap mangkok pada rak tabung, mangkok tersebut dapat diambil kembali atau dikeluarkan untuk dipakai. Mangkok berwarna hijau adalah mangkok yang terakhir masuk dan berada di posisi paling atas rak tabung. Hal ini menyebabkan mangkok hijau yang diambil terlebih dahulu atau dikeluarkan. Setelah mangkok hijau keluar, dilanjutkan dengan mengeluarkan mangkok kuning. Kemudian yang terakhir mangkok merah yang dikeluarkan. Padahal mangkok merah adalah mangkok yang pertama dimasukkan ke dalam rak tabung.

Bagaimana, sudah mengertikah Anda dengan cara kerja Stack ? ☺

Jika telah mengerti dan paham terhadap cara kerja Stack, berikut ini terdapat beberapa hal penting yang berkaitan dengan stack, diantaranya :

a. Bentuk deklarasi Stack :

```
Var Nama_variabel_stack : Stack Of Tipe_Data = {(nilai_elemen_ pertama),
(nilai_elemen_kedua), (nilai_elemen_keN)}
```

Contoh :

```
Var mangkok : Stack Of String = {"Merah", "Kuning", "Hijau"}
```

b. Stack berisi sejumlah elemen yang dapat diisikan seperti pada array. Tidak ada penyebutan jumlah elemen stack pada saat deklarasi, sehingga jumlah elemen stack tidak terbatas.
c. Batasan jumlah elemen stack dapat dilakukan pada saat pengisian elemen stack menggunakan looping for dengan batas akhir sesuai inputan user.
d. Stack juga mempunyai tipe data yang sama dan berlaku untuk semua elemennya dalam satu stack yang sama.
e. Stack mempunyai sejumlah metode yang sering digunakan dalam pengolahan stack, diantaranya sebagai berikut :

1. **Metode push(elemen E)** : Metode yang digunakan untuk memasukkan elemen E ke dalam stack.
2. **Metode pop()** : Metode yang digunakan untuk mengeluarkan satu elemen teratas pada stack.
3. **Metode peek()** : Metode yang digunakan untuk mengetahui elemen yang siap diambil atau elemen yang letaknya pada posisi paling atas.
4. **Metode size()** : Metode yang digunakan untuk mengetahui jumlah elemen yang tersisa di dalam rak stack.
5. **Metode empty()** : Metode yang digunakan untuk mengecek apakah stack tersebut kosong atau tidak.

Contoh : Algoritma Memasukkan Mangkok ke dalam Rak Tabung Menggunakan Konsep Stack.

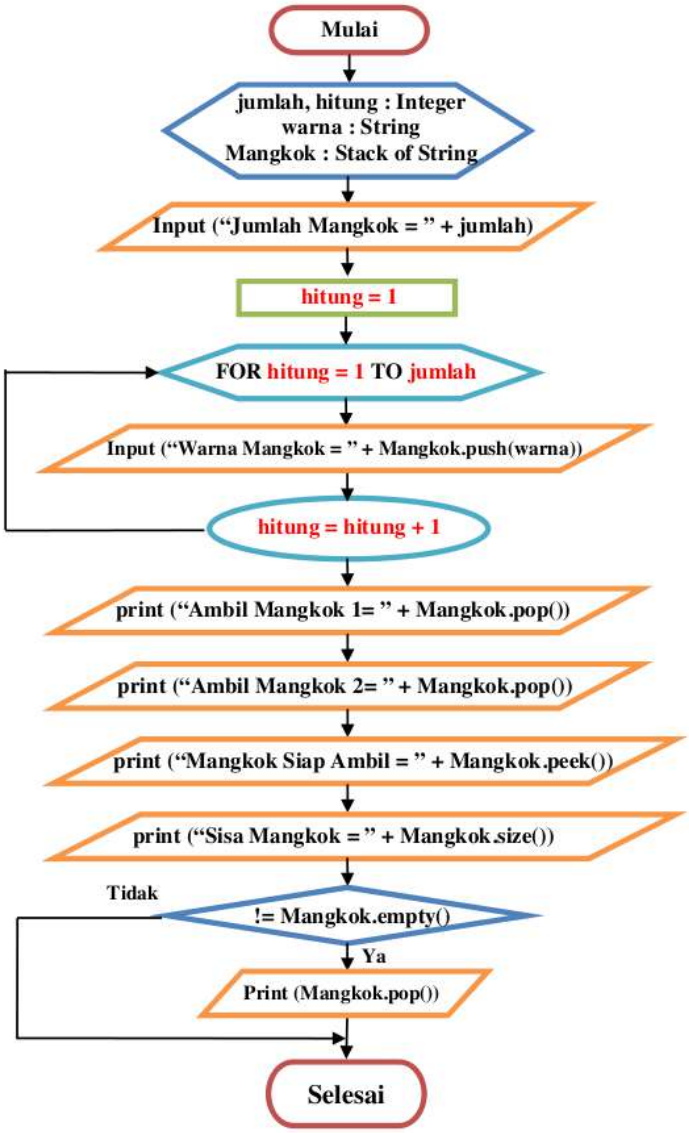
Input : jumlah, warnaMangkok, ambil 2 mangkok teratas

Output : Mangkok yang siap diambil, jumlah mangkok yang tersisa di rak tabung beserta warnanya.

```
Judul : Algoritma Memasukkan Mangkok ke dalam Rak Tabung
Menggunakan Konsep Stack.

Deklarasi :
jumlah, hitung : Integer
warna : String
Mangkok : Stack of String

Deskripsi :
Input ("Jumlah Mangkok = " + jumlah)
hitung = 1
for hitung = 1 to jumlah do
        read ("Warna Mangkok = " + Mangkok.push(warna))
        hitung = hitung + 1
end for
Print ("Ambil Mangkok 1= " + Mangkok.pop())
Print ("Ambil Mangkok 2= " + Mangkok.pop())
Print ("Mangkok Siap Ambil = " + Mangkok.peek())
Print ("Sisa Mangkok = " + Mangkok.size())
while (!= Mangkok.empty()) do
        Print (Mangkok.pop())
end while
```

Gambar 10.2. Pseudocode Memasukkan Mangkok ke dalam Rak Tabung.

Menggunakan Konsep Stack.



Gambar 10.3. Flowchart Memasukkan Mangkok ke dalam Rak Tabung Menggunakan Stack.

10.2 Pengertian Queue

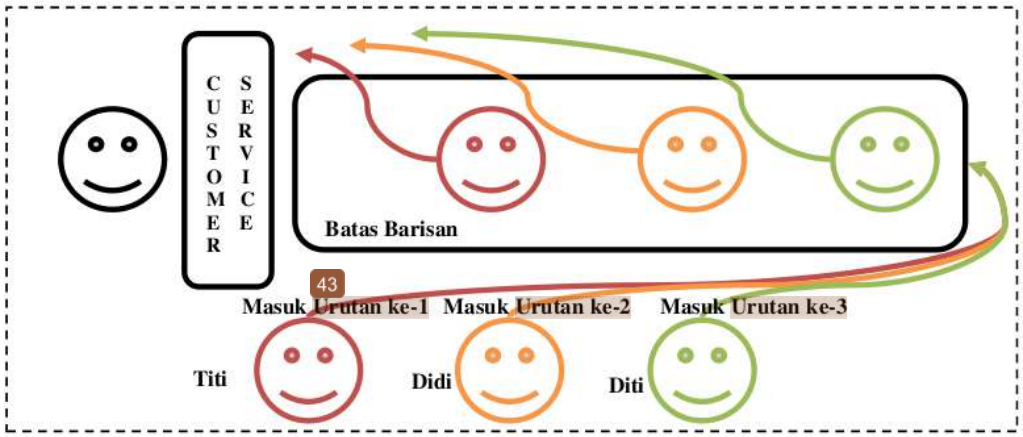
Queue dalam terjemahan bahasa indonesia dapat diartikan sebagai antrian. Antrian berarti sejumlah orang yang berjajar dari samping kiri ke samping kanan atau sebaliknya yang membentuk sebuah barisan. Secara umum barisan tersebut dapat diartikan sebagai garis

yang sifatnya horizontal. Contoh nyata dari antrian dapat dilihat dari orang-orang yang ada di sekitar kita dan sering dijumpai sehari-hari, diantaranya :

a. Antrian orang untuk membeli tiket bioskop.
b. Antrian orang untuk membeli tiket kereta api di loket umum.
c. Antrian orang dalam menunggu pelayanan dari customer service.

Masih banyak lagi contoh nyata dari antrian sejumlah orang yang dapat dibayangkan sebagai sebuah queue. Pada dasarnya cara kerja queue juga mengacu pada antrian sejumlah orang tersebut. Misalnya antrian orang untuk membeli tiket bioskop. Antrian orang dapat terjadi apabila pada saat pertama kali diawali dengan 1 orang yang mengantri di posisi paling depan. Disusul dengan orang ke-2 yang berada di belakang orang ke-1, orang ke-3 yang antri dibelakang orang ke-2, orang ke-4 yang antri di belakang orang ke-3, dan seterusnya. Berjajarnya orang yang demikian dapat menyebabkan terjadinya antrian. Setelah membentuk antrian, mau tidak mau pada saat pelayanan oleh petugas, petugas harus melayani orang di posisi paling depan. Tidak mungkin orang yang berada di posisi bagian tengah atau bagian belakang dilayani terlebih dahulu. Jika hal tersebut terjadi, maka kemungkinan akan ada orang yang marah-marah, karena tidak dilayani sesuai dengan urutan yang paling depan.

Berdasarkan contoh cerita diatas, dapat disimpulkan bahwa antrian dapat terjadi apabila terdapat seseorang yang mengantri pada posisi bagian depan, kemudian disusul orang yang lainnya yang mengantri dibelakannya. Jika ingin melayani seseorang, layani orang pada posisi paling depan terlebih dahulu. Konsep seperti ini disebut dengan konsep FIFO. **FIFO adalah kepanjangan dari *First In First Out***. Orang yang mengantri pertama, maka orang tersebut yang akan dilayani terlebih dahulu. Adapun bagan cara kerja Queue dapat dilihat pada Gambar 10.4.



Gambar 10.4. Bagan Cara Kerja Queue.

Perhatikan Gambar 10.4. Pada Bagan Cara Kerja Queue disediakan 1 batas barisan berbentuk persegi panjang dan terdapat 3 orang yang masing-masing bernama Titi, Didi, dan Diti. Pada saat melakukan proses pengantrian, orang pertama yang bernama Titi merupakan orang yang memasuki barisan antrian pertama kali, sehingga posisi Titi ada di barisan paling depan. Dilanjutkan dengan Didi yang mengantri dibelakang Titi, sehingga posisi Didi berada diurutan ke-2. Kemudian yang terakhir Diti juga ikut mengantri dibelakang Didi, sehingga posisi Diti ada di bagian belakang.

Setelah selesai melakukan proses mengantri, orang yang pertama kali dilayani oleh customer service adalah Titi, karena Titi merupakan orang yang mengantri pertama kali. Sedangkan Diti adalah orang terakhir yang akan dilayani oleh customer service. Hal ini dapat disimpulkan bahwa customer service akan melakukan tugas pelayanan secara urut kepada orang yang mengantri dari posisi paling depan sampai posisi paling belakang.

Bagaimana, sudah mengertikah Anda dengan cara kerja Queue ? ☺

Jika telah mengerti dan paham terhadap cara kerja Queue, berikut ini terdapat beberapa hal penting yang berkaitan dengan stack, diantaranya :

a. **Bentuk deklarasi Queue :**

```
Var  Nama_variabel_queue : Queue of TipeData = {(nilai_elemen_ pertama),
(nilai_elemen_kedua), (nilai_elemen_keN)}
```

Contoh :

```
Var Antrian : Queue of String = {"Titi", "Didi", "Diti"}
```

b. Queue berisi sejumlah elemen yang dapat diisikan seperti pada array. Tidak ada penyebutan jumlah elemen queue pada saat deklarasi, sehingga jumlah elemen queue tidak terbatas.
c. Batasan jumlah elemen queue dapat dilakukan pada saat pengisian elemen queue menggunakan looping for dengan batas akhir sesuai inputan user.
d. Queue juga mempunyai tipe data yang sama dan berlaku untuk semua elemennya dalam satu queue yang sama.
e. Queue mempunyai sejumlah metode yang sering digunakan dalam pengolahan queue, diantaranya sebagai berikut :
    1. **Metode add(elemen E)** : Metode yang digunakan untuk memasukkan elemen E ke dalam queue.

2. **Metode remove()** : Metode yang digunakan untuk mengeluarkan satu elemen terdepan pada queue.
3. **Metode peek()** : Metode yang digunakan untuk mengetahui elemen yang siap dilayani atau elemen yang letaknya pada posisi paling depan.
4. **Metode size()** : Metode yang digunakan untuk mengetahui jumlah elemen yang tersisa di dalam barisan queue.
5. **Metode isEmpty()** : Metode yang digunakan untuk mengecek apakah queue tersebut kosong atau tidak.

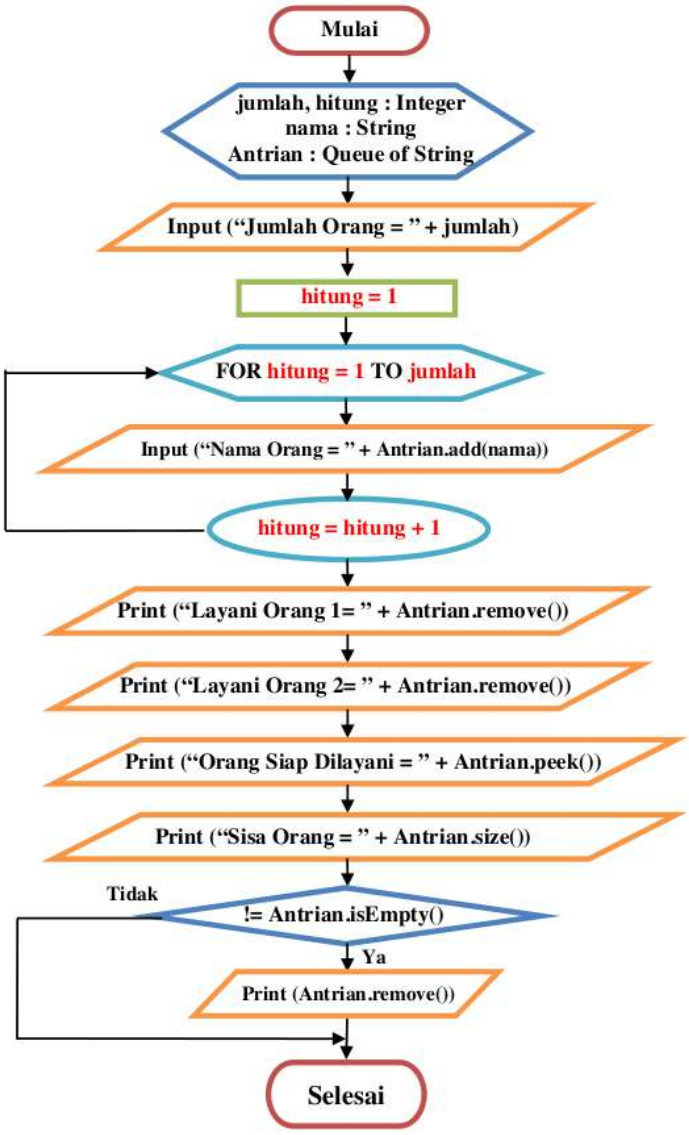
Contoh : Algoritma Mengantri Tiket Bioskop Menggunakan Konsep Queue.

Input : jumlah, namaOrang, layani 2 orang terdepan

Output : Orang yang siap dilayani, jumlah orang yang tersisa di barisan beserta namanya.

```
Judul : Algoritma Mengantri Tiket Bioskop Menggunakan Konsep Queue.

Deklarasi :
jumlah, hitung : Integer
nama : String
Antrian : Queue of String

Deskripsi :
Input ("Jumlah Orang = " + jumlah)
hitung = 1
for hitung = 1 to jumlah do
        read ("Nama Orang = " + Antrian.add(nama))
        hitung = hitung + 1
end for
Print ("Layani Orang 1= " + Antrian.remove())
Print ("Layani Orang 2= " + Antrian.remove())
Print ("Orang Siap Dilayani = " + Antrian.peek())
Print ("Sisa Orang = " + Antrian.size())
while (!= Antrian.isEmpty()) do
        Print (Antrian.remove())
end while
```

Gambar 10.5. Pseudocode Mengantri Tiket Bioskop Menggunakan Konsep Queue.



Gambar 10.6. Flowchart Mengantri Tiket Bioskop Menggunakan Konsep Queue.

10.3 Rangkuman

- **Stack** merupakan sekumpulan elemen yang disusun secara vertikal dengan konsep kerja menggunakan **Metode LIFO (Last In First Out).**
- **Queue** merupakan sekumpulan elemen yang disusun secara horizontal dengan konsep kerja menggunakan **Metode FIFO (First In First Out).**
- Contoh nyata dari aplikasi stack dan queue sering dijumpai dalam kehidupan sehari-hari.
- Bentuk Deklarasi Stack :
  Var Nama_variabel_stack : Stack Of Tipe_Data = {(nilai_elemen_ pertama), (nilai_elemen_kedua), (nilai_elemen_keN)}
- Bentuk Deklarasi Queue :
  Var Nama_variabel_queue : Queue Of Tipe_Data = {(nilai_elemen_ pertama), (nilai_elemen_kedua), (nilai_elemen_keN)}
- Stack dan Queue memiliki sejumlah metode yang dapat dimanfaatkan sebagai pengolahan data pada proses stack dan queue.

10.4 Latihan Soal

Buat Flowchart dan Pseudocode untuk permasalahan di bawah ini menggunakan konsep Stack dan Queue.

1. Di Pesta Ulang Tahun Titi yang ke-21, Titi mengundang seluruh teman kuliahnya. Untuk menyiapkan pesta ulang tahunnya yang direncanakan diadakan secara prasmanan, Titi harus menyusun sejumlah mangkok dan piring untuk keperluan prasmanan. Adapun daftar sejumlah barang harus disusun Titi di meja prasmanan, yaitu sebagai berikut :
   a. Meja I : Mangkok sebanyak 50 buah akan disusun bertumpuk berdasarkan warna, dengan maksimal tumpukan adalah 10 mangkok dengan warna yang sama. Sehingga terdapat 5 susunan mangkok pada Meja I secara urut, yaitu susunan mangkok berwarna merah, kuning, hijau, biru dan coklat.
   b. Meja II : Piring sebanyak 30 buah juga akan disusun bertumpuk berdasarkan warna, dengan maksimal tumpukan adalah 10 piring dengan warna yang sama. Sehingga terdapat 3 susunan piring pada Meja II secara urut, yaitu susunan piring berwarna putih, abu-abu dan hitam.

   Apabila Titi mengundang 15 orang untuk hadir di Pesta Ulang Tahunnya, maka hitunglah sisa jumlah mangkok dan piring di meja I dan meja II. Asumsikan para undangan mengambil mangkok dan piring berdasarkan urutan warna.

2. Di salah satu bioskop ternama di Kota Surabaya sedang mengadakan promo harga tiket nonton, yaitu Rp 25.000 yang berlaku hanya untuk 1 hari saja. Tanpa berpikir panjang, Titi segera mengajak 20 orang temannya untuk nonton bareng dan datang lebih awal. Setibanya di bioskop ada 2 orang customer service yang siap melayani para pembeli tiket. Langsung saja Titi dan teman-temannya segera mengantri. Buatlah algoritma untuk memudahkan pelayanan pembelian tiket dengan pembagian antrian pada 2 orang customer service tersebut.

Bab 11

Rekursif

Tujuan :

- ✓ Mahasiswa dapat membedakan penggunaan proses Looping dan Rekursif untuk permasalahan aritmatika.

11.1 Pengertian Rekursif

Rekursif berasal dari kata *recursive* yang berarti mengulang. Pengulangan ini dapat terjadi dengan cara mengulang kembali dalam memanggil dirinya sendiri. Dalam kamus algoritma dan pemrograman, rekursif dapat diartikan sebagai sebuah bagian dari program (sub program) yang memanggil dirinya sendiri. Mayoritas konsep rekursif lebih banyak digunakan untuk proses aritmatika. Mengapa ? Hal ini dikarenakan dengan proses aritmatika, pada akhir pemanggilan dirinya sendiri terdapat sebuah batas nilai yang dapat dikembalikan lagi untuk menghitung hasil akhirnya. Konsep Rekursif dapat diterapkan dengan menggunakan konsep Fungsi dan Prosedur.

Adapun beberapa hal yang terkait dengan konsep Rekursif, diantara sebagai berikut :

a. Rekursif merupakan bagian dari program (sub program) yang memanggil dirinya sendiri.
b. Syarat sebuah sub program dikatakan melakukan rekursif apabila sub program tersebut memanggil dirinya sendiri dan terdapat batasan berupa nilai untuk menghentikan perulangan yang terjadi di dalam rekursif. Batasan tersebut berfungsi sebagai kondisi bahwa proses rekursif dapat selesai atau berhenti.
c. Sebenarnya penggunaan proses rekursif sangat tidak dianjurkan untuk diterapkan pada algoritma dan pemrograman. Mengapa ? Hal ini dikarenakan apabila sub program yang melakukan rekursif tidak mempunyai kondisi atau batasan yang tepat, maka proses perulangan pemanggilan dirinya sendiri akan terjadi secara terus – menerus. Hal ini mengakibatkan lebih banyak memakan penggunaan sumber daya memory.

d. Beberapa proses aritmatika yang dapat dilakukan menggunakan proses rekursif yaitu :
   1. Menghitung nilai Faktorial.
   2. Menghitung hasil perkalian 2 buah bilangan.
   3. Menghitung hasil perpangkatan 2 buah bilangan.
   4. Mencetak deret fibonacci, dan lain sebagainya.

11.2 Cara Kerja Rekursif

Cara kerja rekursif sebenarnya menggunakan konsep tumpukan (Stack). Rekursif bekerja dengan memanggil sub programnya kembali sampai pada saat dihasilkan sebuah batas nilai. Jika telah selesai, maka nilai tersebut dapat digunakan untuk menghitung nilai sebelumnya. Cara kerja rekursif dapat digambarkan pada Bagan Pemanggilan Sub Program secara Rekursif pada Gambar 11.1 di bawah ini.

Gambar 11.1. Bagan Pemanggilan Sub Program secara Rekursif.

Contoh Proses Menghitung 5 Faktorial menggunakan konsep Rekursif dapat dilihat pada Tabel 11.1 di bawah ini.

Tabel 11.1. Proses Menghitung 5 Faktorial menggunakan Rekursif.

N	Fungsi Faktorial(N)
N = 5	5 * Faktorial (5)
N = 4	4 * Faktorial (4)
N = 3	3 * Faktorial (3)
N = 2	2 * Faktorial (2)
N = 1	1

Berdasarkan Tabel 11.1, dapat dilihat bahwa pada saat 1 faktorial dapat menghasil sebuah nilai yaitu 1. Nilai 1 inilah yang menjadi batasan dari fungsi rekursif. Saat N=1, maka nilai yang dapat direturnkan adalah 1. Selain N=1, untuk N=2,3,4 dan 5 mempunyai pola yang sama yaitu N * (N-1)! atau dapat ditulis secara pseudocode dengan N* Faktorial (N-1). Dalam algoritma, penulisan kondisi seperti ini dapat dinyatakan sebagai berikut :

- n! = n*(n-1)! , jika n>1
- n! = 1 , jika n = 0 atau n=1

```
ALGORITMA UTAMA
{Algoritma Menghitung Nilai Faktorial Menggunakan Rekursif}

Deklarasi :
        nilai, hasilakhir: Integer
        Fungsi  Faktorial (Input N : integer)
Deskripsi :
        Input (nilai)
        hasilakhir = Faktorial(nilai)
        Print ("Nilai Faktorial " + nilai + "=" +hasilakhir)

FUNGSI Faktorial (Input N:integer)
{Menghitung Faktorial}

Deklarasi :
        -   (tidak ada)
Deskripsi :
        if (N == 0) || (N == 1) then
                return (1)
        else
                return (N * Faktorial (N-1))
        end if
```

Gambar 11.2. Pseudocode Menghitung Nilai Faktorial Menggunakan Rekursif Fungsi Dengan Parameter Masukan.

Jika n diinputkan oleh dengan angka 5, bagaimana hasilnya ?

N = 5 : nilai kembalian → **5*faktorial(4) = 5 x 24 = 120**

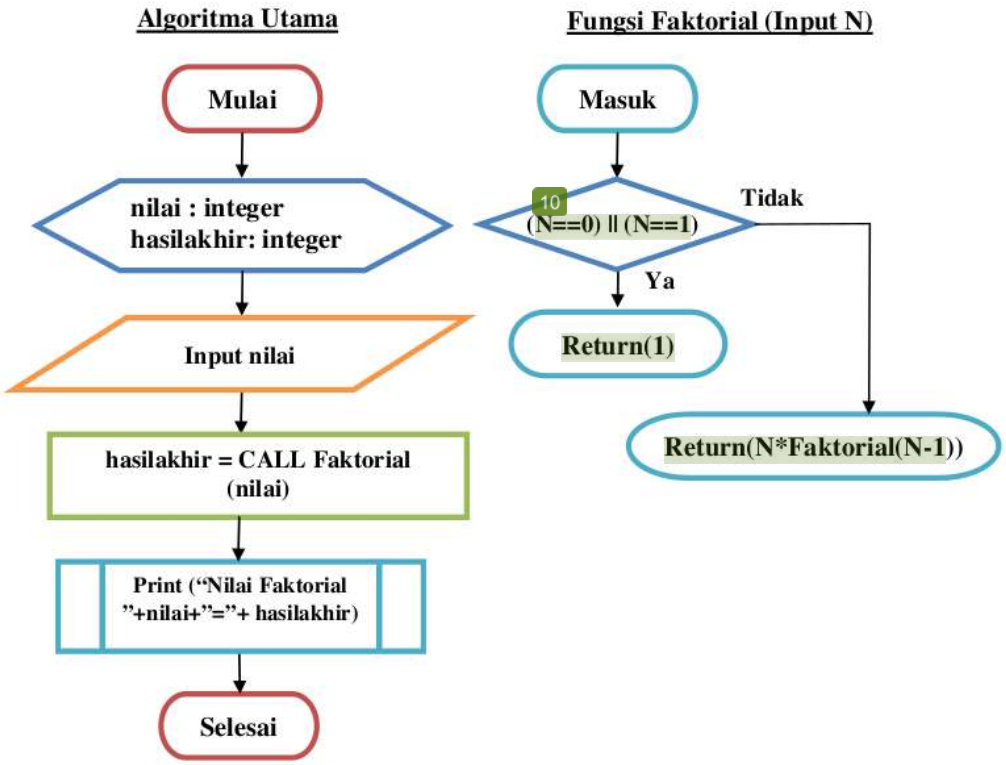
N = 4 : nilai kembalian → **4*faktorial(3) = 4 x 6 = 24**

N = 3 : nilai kembalian → **3*faktorial(2) = 3 x 2 = 6**

N = 2 : nilai kembalian → **2*faktorial(1) = 2 x 1 = 2**

N = 1 : nilai kembalian → **1**

Hasil : 120



Gambar 11.3. Flowchart Menghitung Nilai Faktorial Menggunakan Rekursif Fungsi Dengan Parameter Masukan.

11.3 Rangkuman

- **Rekursif** merupakan bagian dari program (sub program) yang memanggil dirinya sendiri.
- **Syarat sebuah sub program dikatakan melakukan rekursif** apabila sub program tersebut memanggil dirinya sendiri dan terdapat batasan berupa nilai untuk menghentikan perulangan yang terjadi di dalam rekursif. Batasan tersebut berfungsi sebagai kondisi bahwa proses rekursif dapat selesai atau berhenti.
- **Beberapa proses aritmatika** yang dapat dilakukan menggunakan proses rekursif yaitu :
  - Menghitung nilai Faktorial.
  - Menghitung hasil perkalian 2 buah bilangan.
  - Menghitung hasil perpangkatan 2 buah bilangan.
  - Mencetak deret fibonacci, dan lain sebagainya.

11.4 Latihan Soal

Buat Flowchart dan Pseudocode untuk permasalahan di bawah ini menggunakan konsep Rekursif.

1. Menghitung hasil perkalian dari inputan 2 buah bilangan.

   Contoh :

   Masukkan bilangan pertama : 3

   Masukkan bilangan kedua : 5

   Hasil Perkalian 3 x 5 = 15

   Secara Logika → 3 + 3 + 3 + 3 + 3 =15

   Perhatikan : bahwa sebenarnya secara logika perhitungan perkalian berasal dari perhitungan hasil penjumlahan bilangan pertama sebanyak bilangan kedua.

2. Menghitung hasil perpangkatan dari inputan 2 buah bilangan, untuk bilangan yang akan dipangkatkan dan angka pangkatnya.

   Contoh :

   Masukkan bilangan pertama : 6

   Masukkan bilangan kedua : 3

   Hasil Perkalian $6^3 = 216$

   Secara Logika → 6 x 6 x 6 =216

   Perhatikan : bahwa sebenarnya secara logika perhitungan perpangkatan berasal dari perhitungan hasil perkalian bilangan pertama sebanyak bilangan kedua.

3. Mencetak deret fibonacci sebanyak N bilangan.

   Deret Fibonacci : Deret angka dengan pola angka selanjutnya merupakan hasil penjumlahan dari dua angka sebelumnya.

   Contoh :

   Masukkan banyak deret angka yang akan dicetak : 8

   Hasil : 1, 1, 2, 3, 5, 8, 13, 21

   Secara logika → angka awal yang harus ada adalah 0 dan 1

   Cara : **0, 1, 0 + 1 = 1, 1 + 1 = 2, 1 + 2 = 3, 2 + 3 = 5, 3 + 5 = 8, 5 + 8 = 13, 8 + 13 = 21.**

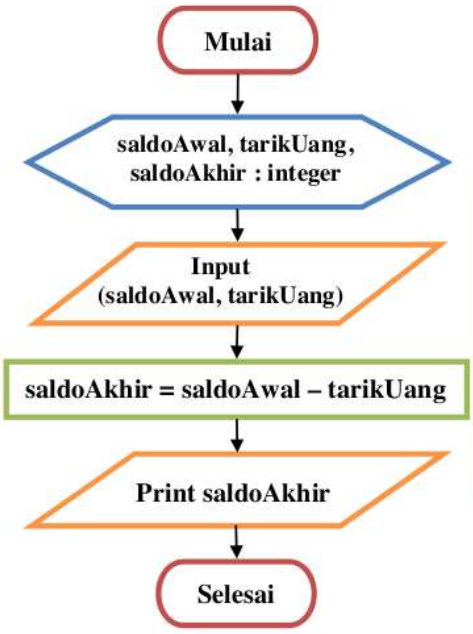
Bab 12

Kumpulan Soal Kombinasi dan Pembahasan

12.1. Kumpulan Soal dan Pembahasan

1. Buat algoritma untuk melakukan transaksi di ATM dengan melakukan penarikan tunai. Nilai uang yang diambil berasal dari inputan user. Tampilkan nilai saldo akhir setelah uang diambil. (Gunakan 3 elemen utama, yaitu input, proses, dan output).

   Pembahasan :



Judul : Algoritma Menghitung Saldo Akhir ATM.

Deklarasi :
saldoAwal, tarikTunai, saldoAkhir : integer;

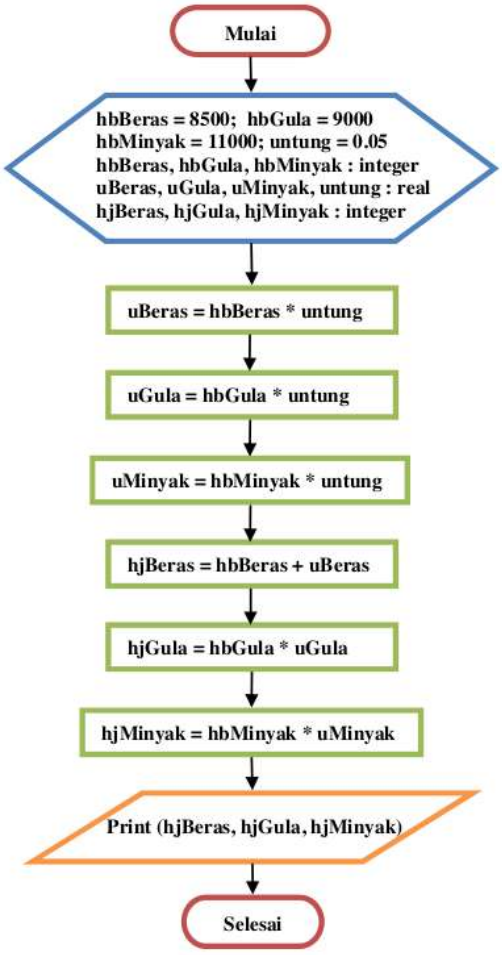
Deskripsi :
Input (saldoAwal, tarikUang)
saldoAkhir ← saldoAwal - tarikUang
Print (saldoAkhir)

2. Sebuah Toko Serba Ada milik Titi menjual sejumlah barang-barang sembako dengan harga murah. Untuk menarik pembeli, Titi harus berhati-hati dalam mengambil untung. Dengan saran dari Mas Didi, perhitungan untung dikenakan sebesar 5% dari harga beli supplier. Adapun jenis barang yang dijual di Toko Titi adalah sebagai berikut :

Nama Barang	Harga Beli	Besar Untung 5%	Harga Jual
Beras	Rp 8.500	?	?
Gula	Rp 9.000	?	?
Minyak	Rp 11.000	?	?

Hitung berapa besar untung yang diambil Titi untuk ketiga jenis barang tersebut. Kemudian cetaklah berapa harga jual ketiga barang tersebut saat dijual ke customer.

Pembahasan :



Judul : Algoritma Menghitung Harga Jual Barang Sembako.

Deklarasi :
hbBeras = integer;
hbGula = integer
hbMinyak = integer;
uBeras = integer;
uGula = integer;
uMinyak = integer;
untung = real;
hjBeras = integer;
hjGula = integer;
hjMinyak = integer;

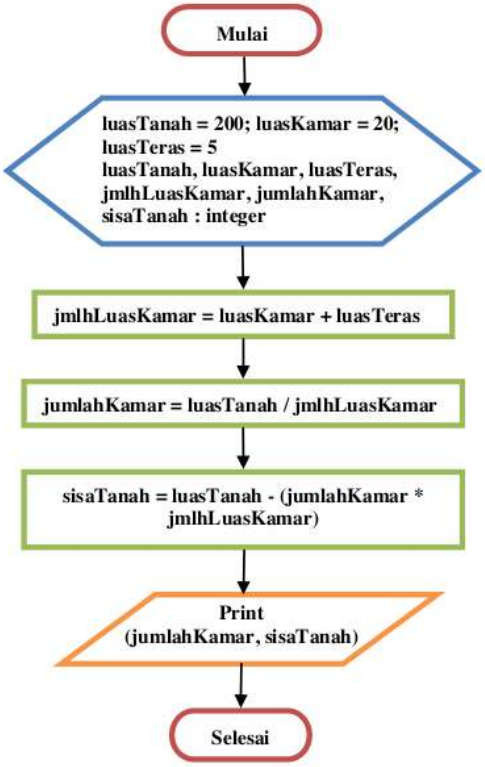
Deskripsi :
hbBeras ← 8500
hbGula ← 9000
hbMinyak ← 11000
untung ← 0.05
uBeras ← hbBeras * untung
uGula ← hbGula * untung
uMinyak ← hbMinyak * untung
hjBeras ← hbBeras + uBeras
hjGula ← hbGula * uGula
hjMinyak ← hbMinyak * uMinyak
print (hjBeras, hjGula, hjMinyak)

3. Ayah Titi sedang melakukan investasi pembangunan kos-kosan bagi para mahasiswa. Kos-kosan tersebut dibangun pada tanah kosong dengan ukuran $10x20m^2$. Adapun pembagian tanah tersebut adalah sebagai berikut :

a. Satu kamar kos akan dibangun dengan luas kamar 5x4m^2.
b. Setiap kamar kos akan ditambahkan 5x1m^2 untuk teras depan.

Hitunglah berapa banyak kamar kos yang dapat dibangun pada tanah tersebut. Apabila terdapat sisa tanah, cetaklah juga luas tanah yang tersisa.

Pembahasan :



Judul : Algoritma Menghitung Pembangunan Jumlah Kamar Kos.

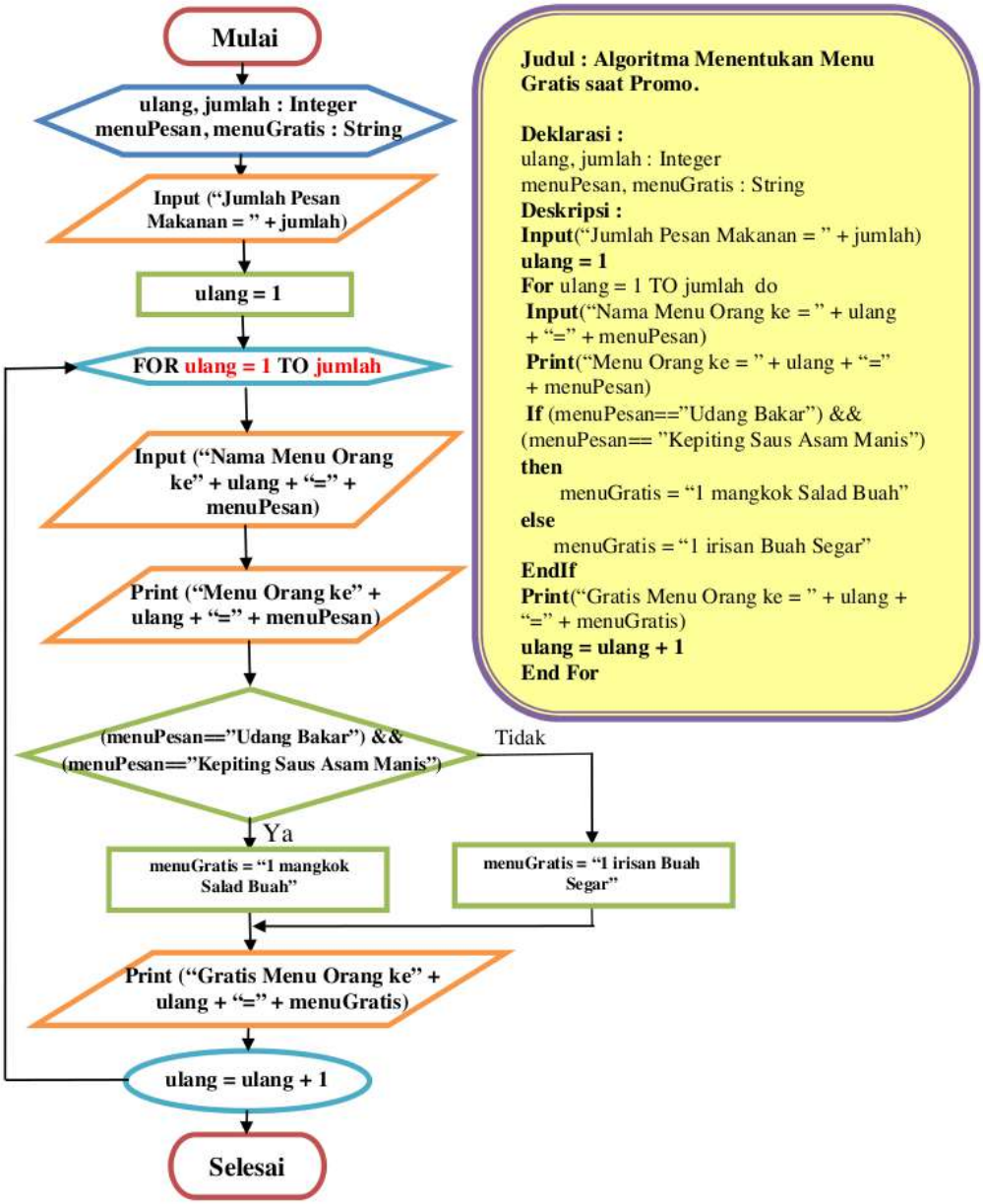
Deklarasi :
luasTanah = integer;
luasKamar = integer;
luasTeras = integer;
jmlhLuasKamar = integer;
jumlahKamar = integer;
sisaTanah = integer;

Deskripsi :
luasTanah ← 200
luasKamar ← 20
luasTeras ← 5
jmlahLuasKamar ← luasKamar + luasTeras
jumlahKamar ← luasTanah / jmlahLuasKamar
sisaTanah = luasTanah + (jumlahKamar / jmlhLuasKamar
Print (jumlahKamar, sisaTanah)

4. Restoran Cepat Saji Hoka-Hoka Hento sedang mengadakan promo. Titi mengajak Mas Didi dan keluarganya untuk makan siang di restoran tersebut. Promo tersebut berlaku dengan syarat dan ketentuan sebagai berikut :
   a. Promo diberikan hanya untuk 2 jenis makanan, yaitu Udang Bakar dan Kepiting Saus Asam Manis.
   b. Jika customer membeli salah satu dari menu tersebut, maka akan diberi bonus 1 mangkok salad buah secara gratis. Pemberian salad tidak berlaku kelipatan.
   c. Namun apabila cutomer tidak membeli salah satu dari menu promo tersebut, maka hanya diberi 1 irisan buah segar.

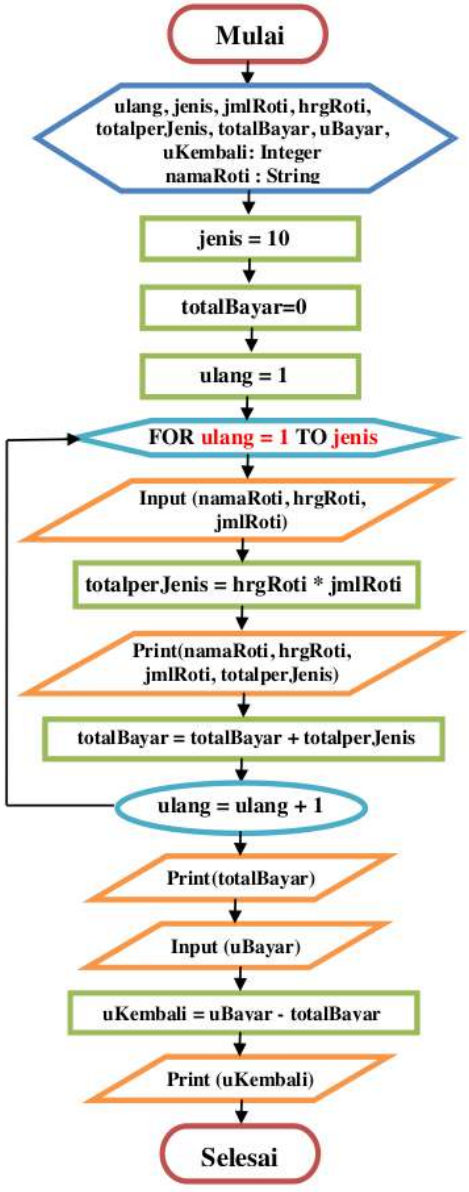
Jika Titi mengajak Mas Didi, ayah dan ibunya, masukkan jenis menu yang dipesan oleh setiap keluarga Titi, kemudian cetaklah bonus yang didapatkan baik untuk menu promo maupun menu diluar promo.

Pembahasan :



5. Titi membeli 10 jenis roti di Toko Sweet Cakes & Bakery. Untuk memudahkan pencetakan nota penjualan roti, kasir harus memasukkan nama roti dan harga setiap roti yang dibeli Titi. Di akhir proses, cetaklah harga Total Bayar semua roti, Jumlah Uang Pembayaran Titi, dan Hasil Kembalian yang harus Titi terima apabila ada.

Pembahasan :



Judul : Algoritma Menghitung dan Mencetak Nota Penjualan Roti.

Deklarasi :
ulang, jenis, jmlRoti, hrgRoti, totalperJenis, totalBayar, uBayar, uKembali : Integer
namRoti : String

Deskripsi :
jenis = 10
totalBayar = 10
ulang = 1

For ulang = 1 TO jenis do
    Input(namaRoti, hrgRoti, jmlRoti)
    totalperJenis = hrgRoti * jmlRoti
    Print(namaRoti, hrgRoti, jmlRoti, totalperJenis)
    totalBayar = toralBayar + totalperJenis
    ulang = ulang + 1
End For

Print (totalBayar)
Input (uBayar)
uKembali = uBayar – totalBayar
Print (uKembali)

6. Menjelang libur Hari Raya Idul Fitri, semua orang di komplek perumahan Titi selalu melakukan mudik bersama. Mudik kali ini dilakukan menggunakan transportasi kereta

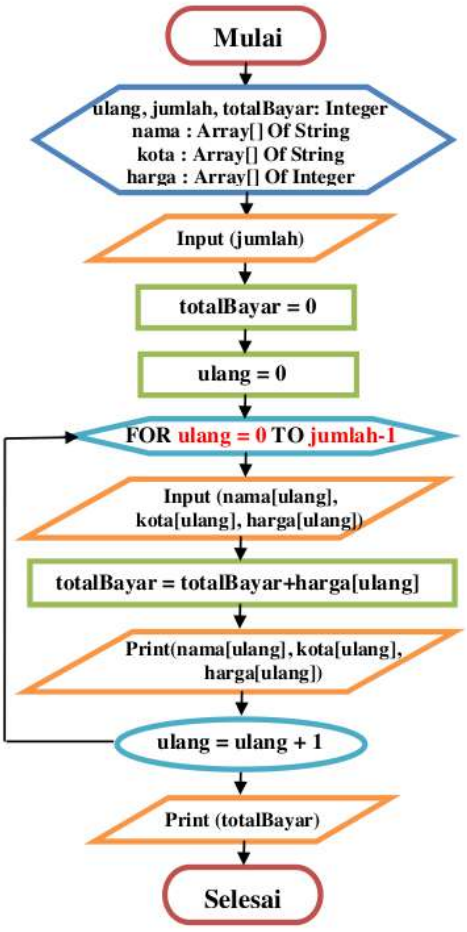
api. Sebelum hari H keberangkatan, Titi mengkoordinir membeli tiket kereta api sejumlah kepala keluarga yang tinggal di komplek perumahannya. Pada saat transaksi pembelian tiket, petugas melakukan pencatatan sejumlah data yang berkaitan dengan identitas dan tujuan penumpang, diantaranya sebagai berikut :

a. Berapa banyak tiket yang akan dibeli ?
b. Nama Penumpang setiap tiket.
c. Kota tujuan Penumpang.
d. Harga tiket yang disesuaikan dengan kota tujuan penumpang.

Setelah selesai memasukkan data, petugas akan mencetak seluruh data tiket penumpang yang dibeli Titi beserta harga total yang harus dibayar Titi. Selesaikan permasalahan ini menggunakan konsep array satu dimensi dan dua dimensi.

Pembahasan :

CARA I : MENGGUNAKAN ARRAY SATU DIMENSI



Judul : Algoritma Menghitung Total Harga Tiket Kereta Menggunakan Array 1D.

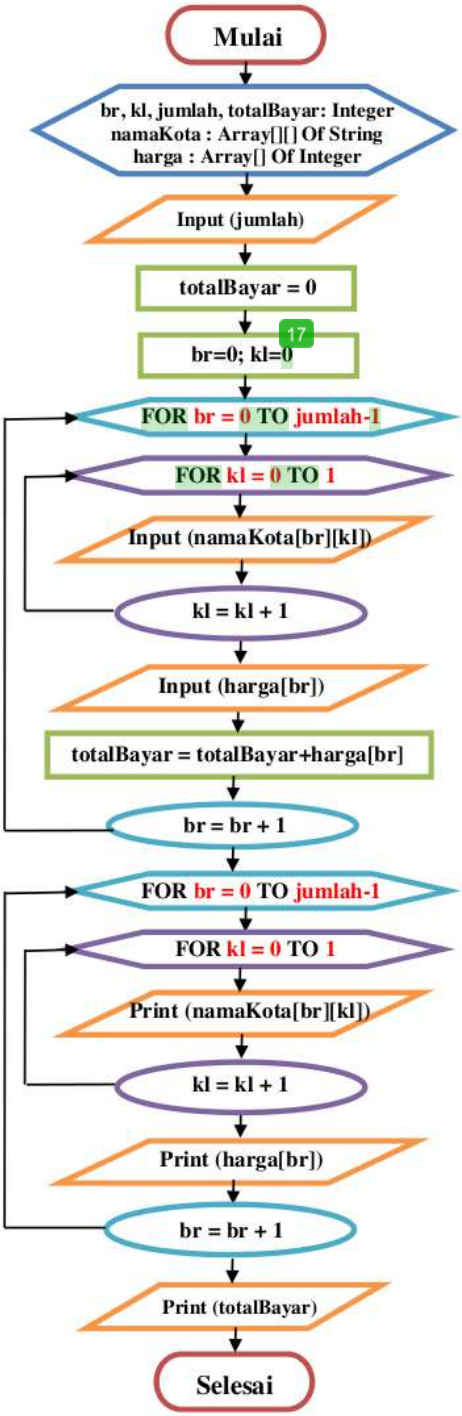
Deklarasi :
ulang,jumlah, totalBayar : Integer
nama : Array[] Of String
kota : Array[] Of String
harga : Array[] Of Integer

Deskripsi :
Input(jumlah)
totalBayar = 0
ulang = 0

For ulang = 0 TO jumlah-1 do
  Input(nama[ulang], kota[ulang], harga[ulang])
  totalBayar = totalBayar+harga[ulang]
  Print(nama[ulang], kota[ulang], harga[ulang])
  ulang = ulang + 1
End For

Print (totalBayar)

CARA II : MENGGUNAKAN ARRAY DUA DIMENSI



Judul : Algoritma Menghitung Total Harga Tiket Kereta Menggunakan Array 2D.

Deklarasi :
br, kl, jumlah, totalBayar : Integer
namaKota : Array[][] Of String
harga : Array[] Of Integer

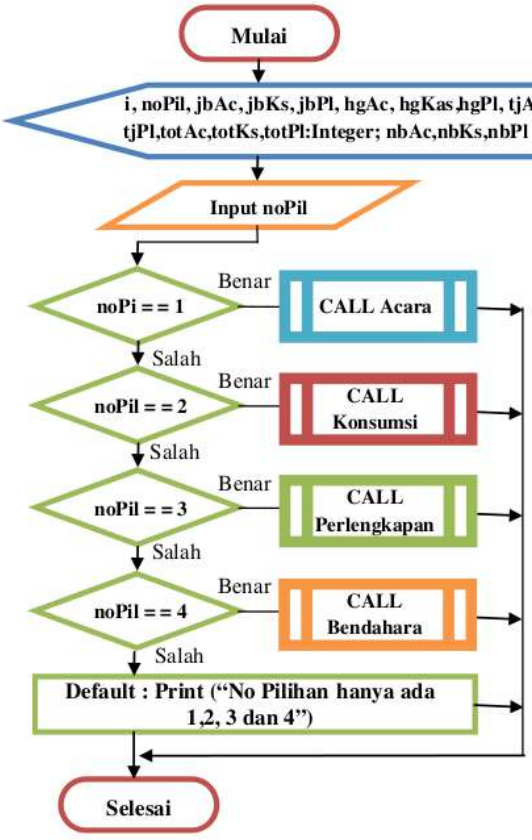
Deskripsi :
Read(jumlah)
totalBayar = 0
br = 0
kl = 0

```
For br = 0 TO jumlah-1 do
 For kl = 0 TO 1 do
    Input(namaKota[br][kl])
     kl = kl + 1
 EndFor
  Input(harga[br])
  totalBayar = totalBayar+harga[br]
  br = br + 1
End For

For br = 0 TO jumlah-1 do
 For kl = 0 TO 1 do
    Print(namaKota[br][kl])
     kl = kl + 1
 EndFor
 Print(harga[br])
 br = br + 1
End For

Print (totalBayar)
```

7. Demi kelancaran berlangsungnya acara tasyakuran yang diadakan di rumah Titi dalam rangka lulus kuliah, Titi membuat sebuah team kecil dari anggota keluarga. Adapun daftar tugas team tersebut adalah sebagai berikut :
   a. Pengurus Acara diserahkan pada Mas Didi.
   b. Pengurus Konsumsi diserahkan pada Ibu Titi.
   c. Pengurus Perlengkapan diserahkan pada Ayah Titi.
   d. Pengurus Bendahara dipegang sendiri oleh Titi.

   Sebagai bendahara, Titi bertugas mengelola semua uang kebutuhan tasyakuran. Tugas Titi adalah mengumpulkan daftar barang yang dibutuhkan setiap pengurus (maks 10 barang) dan memberikan uang sesuai harga barang dan jumlah barang yang dibeli. Buatlah algoritma dengan menggunakan konsep Prosedur dan Fungsi, dimana Prosedur diperuntukkan untuk menginputkan dan mencetak kebutuhan barang setiap pengurus, sedangkan Fungsi digunakan untuk menghitung kebutuhan uang yang harus Titi berikan kepada seluruh pengurus.

Pembahasan :



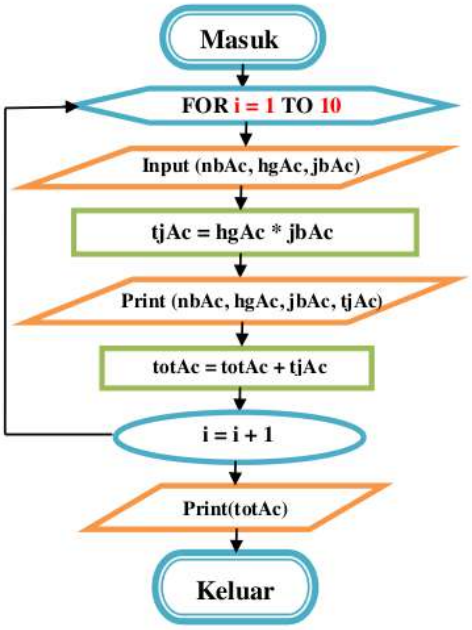
Algoritma Utama

Judul : Algoritma Utama Menentukan Anggaran Setiap Pengurus Acara.

Deklarasi :
i, noPil, jbAc, jbKs, jbPl, hgAc, hgKas, hgPl, tjAc, tjKs, tjPl,totAc,totKs, totPl:Integer;
nbAc,nbKs,nbPl : String

Deskripsi :
Input (noPil)
switch(noPil) {
case 1 : CALL Acara
break;
case 2 : CALL Konsumsi
break;
case 3 : CALL Perlengkapan"
break;
case 4 : CALL Bendahara"
break;
default :
Print ("No Pilihan hanya ada no 1, 2, 3, dan 4")
}

Procedure Acara



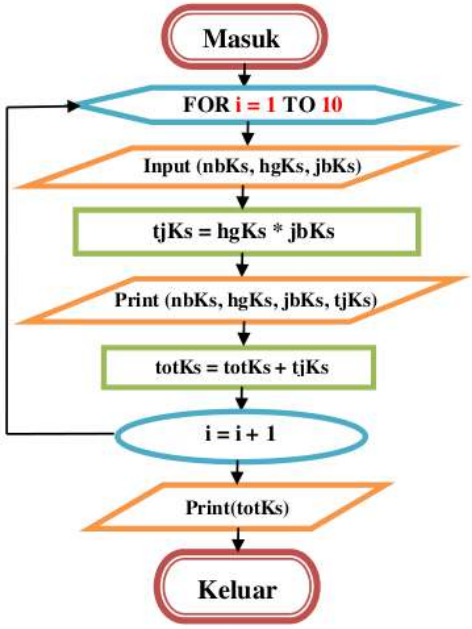
PROCEDURE Acara
{Menghitung Kebutuhan Pengurus Acara}

Deklarasi :
- *(tidak ada)*

Deskripsi :
For i= 1 TO 10 do
  Input(nbAc, hgAc, jbAc)
  tjAc = hgAc * jbAc
  Print(nbAc, hgAc, jbAc)
  totAc = totAc + tjAc
  i= i + 1
End For

Print (totAc)

Procedure Konsumsi



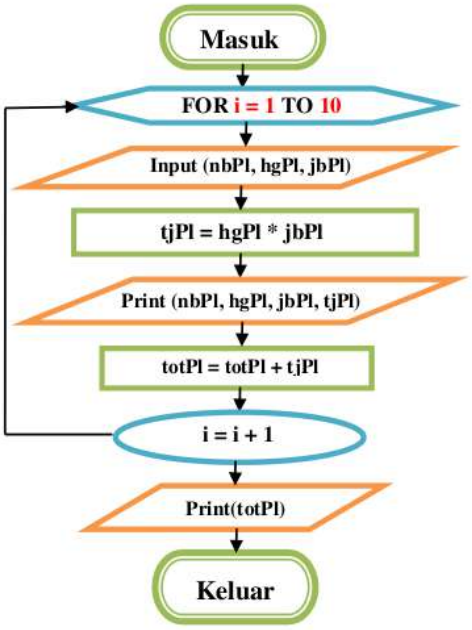
PROCEDURE Konsumsi
{Menghitung Kebutuhan Pengurus Konsumsi}

Deklarasi :
- *(tidak ada)*

Deskripsi :
For i= 1 TO 10 do
  Input(nbKs, hgKs, jbKs)
  tjKs = hgKs * jbKs
  Print(nbKs, hgKs, jbKs)
  totKs = totKs + tjKs
  i= i + 1
End For

Print (totKs)

Procedure Perlengkapan



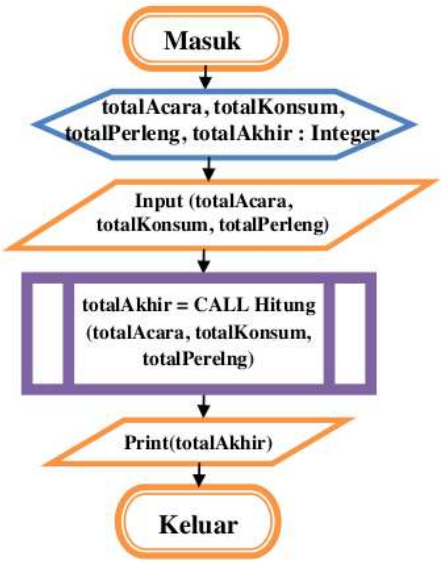
PROCEDURE Perlengkapan
{Menghitung Kebutuhan Pengurus Perlengkapan}

Deklarasi :
- *(tidak ada)*

Deskripsi :
For i= 1 TO 10 do
  Input(nbPl, hgPl, jbPl)
  tjPl = hgPl * jbPl
  Print(nbPl, hgPl, jbPl)
  totPl = totPl + tjPl
  i= i + 1
End For

Print (totPl)

Procedure Bendahara

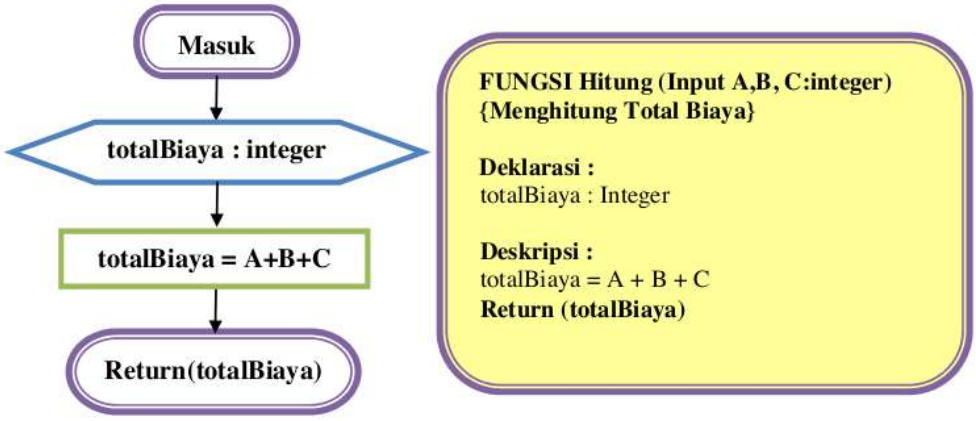


PROCEDURE Bendahara
{Menghitung Pembayaran Bendahara}

Deklarasi :
totalAcara, totalKonsum, totalPerleng, totalAkhir : Integer

Deskripsi :
Input (totalAcara, totalKonsum, totalPerleng)
totalAkhir = CALL Hitung(totalAcara, totalKonsum, totalPerleng)
Print (totalAkhir)

Fungsi Hitung (Input A, B, C)



12.2 Kumpulan Soal Latihan

Buatlah Flowchart dan Pseudocode untuk semua Persoalan di bawah ini. ☺

1. Mas Didi membeli 25 galon air mineral di Toserba depan rumahnya. Kegiatan seperti ini rutin dilakukan Mas Didi 1 bulan sekali. Hitunglah berapa liter air yang dihabiskan Mas Didi dalam waktu sehari, apabila 1 galon tersebut berisi 3,8 liter air ?
2. Buatlah algoritma yang menerima inputan 3 buah bilangan. Kemudian Tampilkan hasil dari penjumlahan, pengurangan, perkalian, pembagian, sisa bagi, dan pangkat ketiga bilangan tersebut.
3. Buatlah program yang dapat memecahkan digit angka dari 5 digit bilangan yang diinputkan. **Perhatikan contoh dibawah ini :**

   38
   Masukkan 5 digit angka : 85372 [inputan]
   Digit Puluh Ribuan = 8 [output]
   Digit Ribuan = 5 [output]
   Digit Ratusan = 3 [output]
   Digit Puluhan = 7 [output]
   Digit Satuan = 2 [output]
4. Titi membeli sejumlah kebutuhan pokok di Toko Sembako Murah. Harga yang ditawarkan untuk sejumlah barang adalah sebagai berikut :
   a. Harga Beras Pandan = Rp 50.000/ 5 kg
   b. Harga Minyak Goreng Murni = Rp 60.000/ 5 liter
   c. Harga Gula Pasir Manis = Rp 75.000/ 10 kg

Buatlah Nota Penjualan dengan inputan berapa banyak jumlah masing-masing barang sembako yang dibeli Titi, lengkap dengan Total Bayar, Uang Pembayaran, dan uang Kembalian yang harus diterima Titi.

5. Buatlah algoritma yang dapat mengkonversi Mata Uang dari inputan berupa Rupiah ke Singapura, jika diketahui sebagai berikut :
   a. 1 Dollar = 11.800 Rupiah
   b. 1 Dollar Amerika = 1,25 Dollar Singapura
6. Mas Didi diminta oleh atasannya untuk membuat sebuah program yang dapat menghitung besarnya Gaji Pokok seluruh karyawan berdasarkan masa kerja. Untuk keperluan tersebut, Mas Didi harus membuat algoritma terlebih dahulu yang dapat menerima inputan berupa nama karyawan, bagian kerja, dan masa kerja karyawan. Adapun kriteria yang harus dipenuhi adalah sebagai berikut :
   a. Jika Masa Kerja Karyawan dibawah 2 tahun, maka Gaji Pokok yang diterima adalah sebesar Rp 5.000.000,-
   b. Jika Masa Kerja Karyawan antara 2-5 tahun, maka Gaji Pokok yang diterima adalah sebesar Rp 7.000.000,-
   c. Jika Masa Kerja Karyawan diatas 5 tahun, maka Gaji Pokok yang diterima adalah sebesar Rp 10.000.000,-
7. Selama event Surabaya Shopping Festival berlangsung, harga beli sejumlah pakaian mendapatkan diskon besar-besaran. Adapun diskon tersebut berlaku untuk jenis pakaian sebagai berikut :
   a. Jika Jenis Pakaian adalah Kemeja dengan Harga Awal dibawah Rp 100.000, maka mendapat diskon sebesar 10% + 20%.
   b. Jika Jenis Pakaian adalah Kemeja dengan Harga Awal diatas Rp 100.000, maka mendapat diskon sebesar 20% + 30%
   c. Jika Jenis Pakaian adalah Celana dengan Harga Awal dibawah Rp 200.000, maka mendapat diskon sebesar 10% + 30%.
   d. Jika Jenis Pakaian adalah Kemeja dengan Harga Awal diatas Rp 200.000, maka mendapat diskon sebesar 20% + 40%

   Buatlah algoritma menggunakan konsep IF Bersarang untuk menyelesaikan persoalan tersebut, dengan inputan berupa jenis pakaian yang dibeli pengunjung dan harga awal pakaian tersebut.

Contoh :

Masukkan Jenis Pakaian : Kemeja [inputan]

Masukkan Harga Awal : Rp 150000 [inputan]

Anda Mendapat Diskon 20% + 30% [output]

Diskon 20% = Rp 30000 **(20% x 150000)** [output]

Harga Setelah Diskon 20% = Rp 120000 **(150000-30000)** [output]

Diskon 30% = Rp 36000 **(30% x 120000)** [output]

Harga Setelah Diskon 30% = Rp 84000 **(120000-36000)** [output]

8. Buatlah sebuah algoritma untuk menentukan pembayaran tiket bus dengan kriteria berdasarkan usia penumpang, diantaranya sebagai berikut :
   a. Jika usia penumpang dibawah 5 tahun mendapat diskon 100%.
   b. Jika usia penumpang antara 5 – 10 tahun, mendapat diskon 50 %.
   c. Jika usia penumpang diatas 10 tahun, tidak mendapat diskon.

   Data yang harus diinputkan adalah tahun kelahiran penumpang dan harga awal tiket.

   Contoh :

   Saat ini adalah Tahun 2014 [inputan]

   Masukkan Tahun Kelahiran Penumpang = 2013 [inputan]

   Masukkan Harga Awal Tiket = Rp 80000 [inputan]

   Diskon yang didapat = 100% [output]

   Harga tiket yang harus dibayar = Rp 0 [output]
9. Sebuah Kalkulator Limited Edition sedang Titi ciptakan untuk memenuhi kebutuhan belajarnya. Kalkulator ini berbeda dengan kalkulator pada umumnya. Perbedaan tersebut terletak pada beberapa proses di bawah ini :
   a. Proses penjumlahan dapat dilakukan jika menginputkan huruf ‘JM’.
   b. Proses pengurangan dapat dilakukan jika menginputkan huruf ‘KR’.
   c. Proses perkalian dapat dilakukan jika menginputkan huruf ‘KL’.
   d. Proses pembagian dapat dilakukan jika menginputkan huruf ‘BG’.

   Buatlah algoritma yang dapat menyelesaikan persoalan tersebut dengan baik dan benar.

   Contoh :

   Masukkan bilangan pertama : 9 [inputan]

   Masukkan kode operator : KR [inputan]

   Masukkan bilangan kedua : 3 [inputan]

   Hasil Pengurangan = 9 – 3 = 6 [output]

10. Titi mempunyai usaha peternakan ayam. Usaha tersebut dirintis sejak ayahnya masih muda. Saat ini terdapat ribuan ayam di pertenakan milik Titi. Untuk lebih meningkatkan usahanya, Titi harus rajin memberi makan setidaknya 30 kg makanan ayam setiap harinya. Dengan 30 kg makanan tersebut, ayam-ayam Titi menghasilkan 200 butir telur per hari yang berasal dari 3 jenis kandang yang berbeda, dengan daftar sebagai berikut :
    a. Kandang A dengan 10 kg makanan dapat menghasilkan 60 butir telur.
    b. Kandang B dengan 10 kg makanan dapat menghasilkan 100 butir telur
    c. Kandang C dengan 10 kg makanan dapat menghasilkan 40 butir telur.

    Buatlah algoritma yang dapat menerima inputan berupa jenis kandang dan banyaknya makanan yang diberikan Titi, sehingga Titi dapat menghitung berapa banyak jumlah telur yang dihasilkan setiap harinya.

    Contoh :

    Masukkan Jenis Kandang = B [inputan]

    Masukkan banyak makanan (kg) = 30 [inputan]

    Jumlah Telur yang dihasilkan = 300 butir telur. [output]
11. Karena tidak mengerjakan Tugas Sekolah, Didi dihukum oleh Sang Guru untuk menulis rasa penyesalannya di selembar kertas. Kata yang harus ditulis adalah "[No.] Saya akan lebih rajin mengerjakan Tugas Sekolah". Tulisan tersebut harus ditulis sebanyak 200 kali. Untuk memudahkan Didi menulis kalimat penyesalannya tersebut, Didi meminta bantuan kepada Titi untuk membuat sebuah program yang dapat mencetak sebanyak yang ia inginkan.

    Contoh :

    Berapa kali kalimat penyesalan akan dicetak = 200 [inputan]

    1. Saya akan lebih rajin mengerjakan Tugas Sekolah. [output]
    2. Saya akan lebih rajin mengerjakan Tugas Sekolah. [output]
    3. Saya akan lebih rajin mengerjakan Tugas Sekolah. [output]

    …..

    200. Saya akan lebih rajin mengerjakan Tugas Sekolah. [output]
12. Buatlah algoritma untuk mencetak deret angka dibawah ini :
    a. 10, 20, 30, 40,…. N
    b. 100, 90, 80, ….. N
    c. 3, 5, 7, 9, 11,…. N
    d. 10, 5, 20, 10, 30, 15, 40, 20, ….. N
    e. 7, 14, 21, 28, …. N

13. Dalam waktu dekat, Titi akan menghadiri acara Penerimaan Raport bersama orang tuanya. Sebelum hari-H penerimaan raport, Titi ingin menghitung sendiri nilainya dirumah. Pada Raport Titi akan ditampilkan nilai rata-rata dari 5 mata pelajaran yang diterima di kelas.

    Contoh :

    Masukkan Nilai Mata Pelajaran ke-1 : 70 [inputan]
    Masukkan Nilai Mata Pelajaran ke-2 : 80 [inputan]
    Masukkan Nilai Mata Pelajaran ke-3 : 60 [inputan]
    Masukkan Nilai Mata Pelajaran ke-4 : 90 [inputan]
    Masukkan Nilai Mata Pelajaran ke-5 : 100 [inputan]
    Jumlah total = 400 [output]
    Rata-rata = 80 [output]

14. Sebuah Restoran Cepat Saji Sate Ayam sedang mengadakan promo untuk pembelian minimal 3 porsi Sate Ayam. Dengan membeli 3 porsi, maka pengunjung Restoran akan mendapatkan 1 porsi Gratis. Porsi Gratis berlaku untuk kelipatannya, seperti membeli 3 porsi, 6 porsi, 9 porsi, dan seterusnya. Buatlah algoritma yang dapat menampilkan jumlah porsi gratis yang diterima oleh pengunjung.

    Contoh :

    Masukkan jumlah porsi Sate Ayam yang dibeli = 12 [inputan]
    3 porsi Sate Ayam GRATIS 1 porsi. [output]
    6 porsi Sate Ayam GRATIS 2 porsi. [output]
    9 porsi Sate Ayam GRATIS 3 porsi. [output]
    12 porsi Sate Ayam GRATIS 4 porsi. [output]

15. Buatlah algoritma menggunakan konsep Array 1D yang dapat menampung sejumlah Data Mahasiswa, yang terdiri dari NIM, Nama, Alamat, TTL, Usia, dan Jurusan. Kemudian tampilkan seluruh isi elemen array tersebut secara urut.

    Contoh :

    Masukkan Jumlah Mahasiswa = 3 [inputan]
    Masukkan NIM Mahasiswa ke-1 = 1241010121 [inputan]
    Masukkan Nama Mahasiswa ke-1 = Dian [inputan]
    Masukkan Alamat Mahasiswa ke-1 = Jl. Surabayan No. 10 [inputan]
    Masukkan TTL Mahasiswa ke-1 = 12 Juli 1995 [inputan]
    Masukkan Usia Mahasiswa ke-1 = 19 [inputan]
    Masukkan Jurusan Mahasiswa ke-1 = Ekonomi [inputan]

Lakukan inputan seterusnya sampai Mahasiswa ke-3.

Hasil Data Mahasiswa Keseluruhan :

Data Mahasiswa ke-1

NIM : 1241010121

Nama : Dian

Alamat : Jl. Surabayan No. 10

TTL : 12 Juli 1995

Usia : 19 Tahun

Jurusan : Ekonomi

Data Mahasiswa ke-2

…….

Data Mahasiswa ke-3

…….....

16. Buatlah algoritma menggunakan konsep Array 2D untuk penyimpanan Data Mahasiswa pada no.15 diatas.
17. Buat algoritma menggunakan **konsep Prosedur dengan 2 macam parameter yaitu parameter masukan dan keluaran** untuk menghitung keliling persegi, luas persegi, dan volume kubus dengan **inputan satu variabel yang sama yaitu, panjang sisi.**

    Catatan :

    Rumus keliling persegi = 4 x sisi

    Rumus luas persegi = sisi x sisi

    Rumus volume kubus = 6 x sisi x sisi
18. Di perpustakaan terdapat 1 rak besar untuk menumpuk buku sesuai dengan jenis kelompok buku. Penumpukan buku diatur berdasarkan ketentuan sebagai berikut :
    a. Bagian Rak Paling Bawah digunakan untuk menumpuk **buku jenis Kamus.**
    b. Bagian Rak Tengah digunakan untuk menumpuk **buku jenis Legenda/Cerita Rakyat.**
    c. Bagian Rak Paling Atas digunakan untuk menumpuk **buku jenis Romansa**.

    Buatlah algoritma dengan konsep Stack yang dapat menampung inputan 3 judul buku pada setiap jenis buku, sehingga apabila ada pembaca yang ingin membaca salah satu jenis buku, maka judul buku paling atas yang harus diambil terlebih dahulu.

Contoh :

Rak Buku Paling Bawah **untuk jenis Kamus**

Masukkan Judul Buku ke-1 : Kamus Bahasa Indonesia **[inputan]**

Masukkan Judul Buku ke-2 : Kamus Bahasa Inggris **[inputan]**

Masukkan Judul Buku ke-3: Kamus Bahasa Jerman **[inputan]**

Rak Buku Paling Bawah **untuk jenis Legenda**

…… **[inputan]**

Rak Buku Paling Bawah **untuk jenis Romansa**

…… **[inputan]**

Masukkan jenis buku yang ingin diambil : **Kamus [inputan]**

Sebaiknya ambil **Kamus Bahasa Jerman** terlebih dahulu. **[output]**

19. Sekelompok orang di kompleks perumahan Titi mengadakan Arisan Khusus. Arisan ini akan diberikan secara bergiliran untuk pendaftar pertama terlebih dahulu, disusul pendaftar kedua, ketiga, dan seterusnya. Buatlah algoritma dengan menggunakan konsep Queue, sehingga permasalahan tersebut dapat diatasi.

    Contoh :

    Masukkan Jumlah Pendaftar Arisan = 5 **[inputan]**

    Masukkan Nama Pendaftar ke-1 = Ibu Titi **[inputan]**

    Masukkan Jumlah uang Arisan ke-1 = Rp 50000 **[inputan]**

    Masukkan Nama Pendaftar ke-2 = Ibu Hasan **[inputan]**

    Masukkan Jumlah uang Arisan ke-1 = Rp 30000 **[inputan]**

    Masukkan Nama Pendaftar ke-3 = Ibu Rudi **[inputan]**

    Masukkan Jumlah uang Arisan ke-1 = Rp 40000 **[inputan]**

    Masukkan Nama Pendaftar ke-4 = Ibu Didi **[inputan]**

    Masukkan Jumlah uang Arisan ke-1 = Rp 100000 **[inputan]**

    Masukkan Nama Pendaftar ke-5 = Ibu Budi **[inputan]**

    Masukkan Jumlah uang Arisan ke-1 = Rp 70000 **[inputan]**

    Yang mendapat Giliran Arisan Khusus **adalah Ibu Titi** dengan uang **sebesar Rp 500000 [output]**

20. Buatlah sebuah algoritma yang dapat menampilkan deret angka berupa hasil penjumlahan bilangan pangkat dua dan bilangan pangkat tiga. Gunakan konsep rekursif untuk menampilkan deret angkat bilangan pangkat dua dan pangkat tiga.

Masukkan banyak angka yang ingin ditampilkan= 5 **[inputan]**

Deret Angka Dasar = 1, 2, 3, 4, 5 **[output]**

Deret Pangkat dua = 1, 4, 9, 16, 25 **[output]**

Deret Pangkat tiga = 1, 8, 27, 84, 125 **[ouput]**

Jumlah Deret Pangkat dua dan Pangkat tiga = 2, 12, 36, 100, 150 **[output]**

Daftar Pustaka

Kristanto, Andi. ***"Algoritma & Pemrograman Dengan C++ Edisi 2".*** Graha Ilmu, Yogyakarta. 2009.

Munir, R. ***"Algoritma dan Pemrograman dalam Bahasa Pascal dan C Edisi Revisi".*** Informatika, Bandung. 2011.

Munir, Rinaldi dan Lidya, Leoni. ***"Algoritma dan Pemrograman, Buku 2".*** Informatika, Bandung. 2002.

Sholiq. ***"Logika dan Algoritma".*** Perpustakaan STIKOM Surabaya, Surabaya. 2004.

Sjukani, Moh. ***"Algoritma dan Struktur Data 1 dengan C, C++, dan Java".*** Mitra Wacana Media. 2010.

Suarga. ***"Algoritma dan Pemrograman".*** Andi, Yogyakarta. 2012.

Swastika, Windra dan Lucky, Paulus. ***"Dasar Algoritma dan Pemrograman Menggunakan C dan Java".*** Prestasi Pustaka. 2012.

Glosarium

A

Algoritma adalah sebuah prosedur langkah demi langkah untuk memecahkan masalah atau menyelesaikan suatu tugas.

Ambiguous adalah sesuatu yang mempunyai arti ganda.

Array atau larik adalah variabel yang terdiri dari kumpulan beberapa nilai yang mempunyai tipe data yang sama.

B

Bahasa pemrograman merupakan prosedur atau tata cara penulisan program.

Boolean adalah sebuah tipe data yang digunakan untuk menyimpan data yang hanya mempunyai 2 nilai, yaitu true dan false.

C

Camel case adalah aturan penulisan variabel dimana selalu diawali huruf kecil dan huruf besar di bagian tengah, seperti punuk unta.

Char adalah sebuah tipe data yang digunakan untuk menyimpan data berupa karakter (huruf alphabet/angka/tanda baca.

Code adalah kode program.

Counter adalah variabel yang berfungsi sebagai penambah satu angka menuju tingkat perulangan berikutnya.

D

Definiteness adalah aturan pada algoritma bahwa Setiap langkah algoritma harus didefinisikan dengan tepat dan tidak menimbulkan makna ambigu.

E

Effectiveness adalah aturan pada algoritma bahwa langkah-;angkah pada algoritma dikerjakan dalam waktu yang wajar.

Ekspresi adalah pernyataan atau sebuah proses yang mentransformasikan nilai menjadi hasil (keluaran) yang diinginkan melalui proses perhitungan (komputasi).

F

Finiteness adalah aturan pada algoritma untuk dapat berakhir (terminate) setelah melakukan sejumlah langkah proses.

Flowchart adalah suatu bagan terurut untuk menggambarkan alur yang terjadi pada suatu proses, dengan menggunakan simbol – simbol tertentu.

Flowchart program adalah bagan yang menggambarkan arus logika dari data yang akan diproses dalam suatu program dari awal sampai akhir.

Flowchart sistem adalah bagan yang menggambarkan arus data dari sebuah sistem.

Fungsi adalah bagian dari program yang dibuat terpisah untuk melaksanakan fungsi tertentu yang menghasilkan suatu nilai untuk dikembalikan ke program utama *(return)*.

H

High Level Language bahasa pemrograman yang menggunakan aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar bahasa yang langsung dapat dipahami oleh manusia.

I

Indeks adalah bentuk penomoran atau tanda yang digunakan pada array/larik.

Input adalah data masukan pada algoritma, baik berupa deklarasi variabel yang akan digunakan selama proses berjalan maupun variabel yang nilainya berasal dari inputan user.

Integer adalah sebuah tipe data yang digunakan untuk menyimpan bilangan bulat.

K

Konstanta adalah variabel yang nilainya tetap.

L

Logika dapat didefinisikan sebagai ilmu yang harus menggunakan prinsip-prinsip tertentu agar dapat berfikir secara logis (tepat dan benar) menurut kaidah/aturan tertentu.

Logika Algoritma adalah suatu cabang khusus dalam ilmu komputer yang mempelajari bagaimana para peserta didik dapat berpikir secara logis dalam pembuatan sebuah bentuk algoritma, baik menggunakan flowchart maupun pseudocode.

Looping Process adalah sebuah proses algoritma dimana sebuah perintah/instruksi dapat dijalankan secara berulang sampai kondisi/batas tertentu.

Low Level Language adalah bahasa yang berorientasi pada mesin.

M

Middle Level Language adalah bahasa pemrograman yang menggunakan aturan-aturan gramatikal dalam penulisan ekspresi atau pernyataan dengan standar yang mudah dipahami manusia.

O

Operand adalah nilai yang diberikan, dapat berupa variabel, konstanta, nilai, dan nilai balik dari fungsi.

Operator adalah penghubung antar operand.

Out of space adalah sebuah kondisi pada memori komputer yang diakibatkan adanya proses perulangan yang tanpa batas.

Output adalah data keluaran yang dihasilkan berdasarkan inputan dan proses yang terjadi pada sebuah algoritma.

P

Pemrograman adalah proses mengimplementasikan urutan langkah untuk menyelesaikan suatu masalah dengan menggunakan suatu bahasa pemrograman.

Pemrograman Berorientasi Objek adalah proses mengimplementasikan urutan langkah-langkah untuk menyelesaikan suatu masalah dalam bentuk program yang berorientasi pada objek dan class.

Pemrograman Terstruktur adalah proses mengimplementasikan urutan langkah-langkah untuk menyelesaikan suatu masalah dalam bentuk program yang memiliki rancang bangun yang terstruktur.

Program adalah merupakan susunan instruksi (kata, ekspresi, pernyataan atau kombinasinya) yang dirangkai dan disusun manjadi satu kesatuan prosedur.

Prosedur adalah bagian dari suatu program yang disusun secara terpisah untuk melakukan suatu tugas khusus/fungsi tertentu, tanpa mengembalikan nilai dari hasil proses tersebut.

Proses adalah tahap pengolahan data, baik menggunakan logika proses maupun model matematika yang sesuai.

Pseudo adalah imitasi/mirip/menyerupai.

Pseudocode adalah bentuk penyajian algoritma dengan menggunakan struktur bahasa tertentu.

Q

Queue adalah sekumpulan elemen yang disusun secara horizontal dengan konsep kerja menggunakan metode FIFO (First In First Out).

R

Real adalah sebuah tipe data yang digunakan untuk menyimpan bilangan pecahan/desimal.

Rekursif adalah proses perulangan dimana pengulangan terjadi dengan cara mengulang kembali dalam memanggil dirinya sendiri.

S

Selection Process adalah sebuah proses pada algoritma dimana sebuah perintah/instruksi dapat dijalankan dengan syarat/kondisi tertentu harus dipenuhi terlebih dahulu.

Semantik adalah aturan-aturan untuk menyatakan suatu arti.

Sequence Process adalah sebuah proses pada algoritma dimana semua semua perintah/instruksi dijalankan secara berurutan, satu persatu, mulai dari yang pertama sampai yang terakhir.

Sintak adalah aturan-aturan gramatikal yang mengatur tata cara penulisan kata, ekspresi dan pernyataan.

Stack adalah sekumpulan elemen yang disusun secara vertikal dengan konsep kerja menggunakan metode LIFO (Last In First Out).

String adalah sebuah tipe data yang digunakan untuk menyimpan kata/kalimat.

Sub Program adalah kode program yang ditulis pada main program menjadi program-program kecil, tanpa mengubah tugas utamanya.

T

Tipe Data adalah pola penyajian data dalam memori komputer, seperti alamat yang dapat menunjukkan dimana objek data disimpan.

V

Variabel adalah suatu lokasi memori komputer yang digunakan untuk menampung dan menyimpan data yang akan diolah.

Indeks

Buku Ajar Logika

ORIGINALITY REPORT

14%	13%	2%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.uny.ac.id Internet Source	2%
2	www.scribd.com Internet Source	1%
3	id.scribd.com Internet Source	1%
4	rplsapra2.blogspot.com Internet Source	1%
5	sisfo.itp.ac.id Internet Source	<1%
6	sakurachan89.blogspot.com Internet Source	<1%
7	sarifahparifdah.blogspot.com Internet Source	<1%
8	nyohongo.wordpress.com Internet Source	<1%
9	albertsally7.blogspot.com Internet Source	<1%

10	sugi4u.blogspot.com Internet Source	<1%
11	es.scribd.com Internet Source	<1%
12	dragtegal.blogspot.com Internet Source	<1%
13	andisantoso504.blogspot.com Internet Source	<1%
14	core.ac.uk Internet Source	<1%
15	pt.scribd.com Internet Source	<1%
16	hsnetkomputer.blogspot.com Internet Source	<1%
17	Henning Mittelbach. "Programmierkurs TURBO-PASCAL Version 7.0", Springer Science and Business Media LLC, 1998 Publication	<1%
18	selamat-berkreasi.blogspot.com Internet Source	<1%
19	issuu.com Internet Source	<1%
20	110.139.54.25 Internet Source	<1%

21	B. Marincek, J. L. Marais, E. Zeller. "Oberon", Springer Nature, 1999 Publication	<1%
22	karindpermata.wordpress.com Internet Source	<1%
23	repository.usu.ac.id Internet Source	<1%
24	mohammadkafi.blogspot.com Internet Source	<1%
25	rochmafebraqius.blogspot.com Internet Source	<1%
26	de.scribd.com Internet Source	<1%
27	www.hendiwicaksono.com Internet Source	<1%
28	www.slideshare.net Internet Source	<1%
29	redaksi.pens.ac.id Internet Source	<1%
30	adhetomcatcleverley.wordpress.com Internet Source	<1%
31	ghishacitanndah.blogspot.com Internet Source	<1%
	wahyuikalestari.blogspot.com	

32	Internet Source	<1%
33	www.kepanen.com Internet Source	<1%
34	fx-teknikomputer.blogspot.com Internet Source	<1%
35	pt.slideshare.net Internet Source	<1%
36	www.smkn1gorontalo.sch.id Internet Source	<1%
37	fti.unand.ac.id Internet Source	<1%
38	www.slideserve.com Internet Source	<1%
39	media.neliti.com Internet Source	<1%
40	widyantorowasito.blogspot.com Internet Source	<1%
41	anna_fitria.staff.gunadarma.ac.id Internet Source	<1%
42	www.souverindo.com Internet Source	<1%
43	ficktriinapratiwi.blogspot.com Internet Source	<1%

44	si.stikom.edu Internet Source	<1%
45	id.123dok.com Internet Source	<1%
46	Submitted to Universitas Brawijaya Student Paper	<1%
47	smknduawng.blogspot.com Internet Source	<1%
48	Henning Mittelbach. "Einführung in TURBO-PASCAL", Springer Nature, 1988 Publication	<1%
49	vidykanovianda.blogspot.com Internet Source	<1%
50	www.nadnet.ch Internet Source	<1%
51	mu-indo.blogspot.com Internet Source	<1%
52	catatananalgo.wordpress.com Internet Source	<1%
53	downloads.omron.se Internet Source	<1%
54	infosuryo.blogspot.com Internet Source	<1%

55	yoursilentbell.blogspot.com Internet Source	<1%
56	kumaha-aingk.blogspot.com Internet Source	<1%
57	edoc.pub Internet Source	<1%
58	frengkyonline.wordpress.com Internet Source	<1%
59	docslide.us Internet Source	<1%
60	113028stmik.blogspot.com Internet Source	<1%
61	eprints.upnjatim.ac.id Internet Source	<1%
62	loveninx.wordpress.com Internet Source	<1%
63	edoc.site Internet Source	<1%
64	tia-sopyan.blogspot.com Internet Source	<1%
65	trianjaharya23.blogspot.com Internet Source	<1%
66	ronirosandi.blogspot.com Internet Source	

		<1%
67	Submitted to Walsall College Student Paper	<1%
68	backingflash.blogspot.com Internet Source	<1%
69	leopard.blog.binusian.org Internet Source	<1%
70	elsaelsinda.blogspot.com Internet Source	<1%
71	www.bahasapemrograman.com Internet Source	<1%
72	zeealkatiry.blogspot.com Internet Source	<1%
73	repository.uinsu.ac.id Internet Source	<1%
74	fatkurbelajar.wordpress.com Internet Source	<1%
75	mikidinarti.blogspot.com Internet Source	<1%
76	beams.ulb.ac.be Internet Source	<1%
77	susirifqie.wordpress.com Internet Source	<1%

78	share.its.ac.id Internet Source	<1%
79	akristiyanti.blogspot.com Internet Source	<1%
80	fransgrand.blogspot.com Internet Source	<1%
81	laptop-pekanbaru.blogspot.com Internet Source	<1%
82	bingungnamablogg.blogspot.co.id Internet Source	<1%
83	algoritma-program.blogspot.com Internet Source	<1%
84	ml.scribd.com Internet Source	<1%
85	widuri.raharja.info Internet Source	<1%
86	itsathayaahm.blogspot.com Internet Source	<1%
87	ria-sevtiany.blogspot.com Internet Source	<1%
88	manajement-info.blogspot.com Internet Source	<1%
89	etheses.uinmataram.ac.id	

	Internet Source	<1%
90	wwwdienz.blogspot.com Internet Source	<1%
91	www.klikdokter.com Internet Source	<1%
92	thyqhy.blogspot.com Internet Source	<1%
93	achmatim.net Internet Source	<1%
94	asepmultimedia.wordpress.com Internet Source	<1%
95	ekafirgiawan.blogspot.com Internet Source	<1%
96	katamantap.blogspot.com Internet Source	<1%
97	M. Ridwan Nur Septian, Agus Sidiq Purnomo. "Sistem Penilaian Pegawai Menggunakan Metode Fuzzy Multiple Attribute Decision Making (FMADM) dan Weighted Product (WP)", JMAI (Jurnal Multimedia & Artificial Intelligence), 2017 Publication	<1%
98	Submitted to Universitas Islam Indonesia Student Paper	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		