

# **Modul Praktikum Pemrograman Komputer**

Oleh:

**Veny V. Ponggawa, SST., MT**



**POLITEKNIK NEGERI MANADO  
JURUSAN TEKNIK ELEKTRO  
PROGRAM STUDI D3 TEKNIK KOMPUTER  
2018**

## **KATA PENGANTAR**

Puji dan syukur patut patutlah dipanjatkan kehadirat Tuhan Yang Maha Besar, karena atas kasih dan anugerah\_NYA sehingga modul Praktikum Pemograman Komputer ini boleh terselesaikan dengan. Modul ini dibuat untuk melengkapi materi pembelajaran di Program Studi D3 Teknik Komputer di Jurusan Teknik Elektro Politeknik Negeri Manado. Didalamnya berisikan uraian materi yang sesuai dengan Rencana Pembelajaran Semester Mata Kuliah Praktikum Pemograman Komputer dan kegiatan praktikum yang harus dikerjakan oleh mahasiswa yang mengambil mata kuliah ini.

Pada kesempatan ini perkenalkan penulis mengucapkan terima kasih kepada :

1. Ir.Ever Slat,MT; sebagai Direktur Politeknik Negeri Manado
2. Dra.Maryke Alelo,MBA sebagai Wakil Direktur Bidang Akademik
3. Fanny Doringin,ST.,MT; sebagai Ketua Jurusan Teknik Elektro
4. Marson Budiman, SST., MT; sebagai Kaprodi D3 Teknik Komputer
5. Rekan – rekan di Prodi D3 Teknik Komputer

Akhirnya besar harapan kiranya dengan adanya modul ini diharapkan dapat menunjang kegiatan pembelajaran di Program Studi D3 Teknik Komputer.

Manado, Desember 2018

Penulis,

**DAFTAR ISI**

<b>LEMBAR PENGESAHAN</b> .....	<b>Error! Bookmark not defined.</b>
<b>KATA PENGANTAR</b> .....	<b>ii</b>
<b>DAFTAR ISI</b> .....	<b>iii</b>
<b>Bab 1</b> .....	<b>1</b>
<b>Algoritma</b> .....	<b>1</b>
<b>1.1. Definisi algoritma</b> .....	<b>1</b>
<b>1.2. Kriteria algoritma</b> .....	<b>2</b>
<b>1.3. Bentuk-bentuk dasar algoritma</b> .....	<b>3</b>
<b>1.4. Contoh kasus</b> .....	<b>5</b>
<b>Bab 2</b> .....	<b>6</b>
<b>Flowchart</b> .....	<b>6</b>
2.1. Definisi Flowchart.....	6
2.2. Simbol-Simbol Flowchart.....	8
2.2 Contoh Kasus.....	10
<b>Bab 3</b> .....	<b>11</b>
<b>Tipe Data, Variabel&amp; Konstanta</b> .....	<b>11</b>
3.1. Tipe Data.....	11
<b>3.2. Variabel &amp; Konstanta</b> .....	<b>12</b>
<b>3.3. Deklarasi</b> .....	<b>13</b>
<b>3.4. Contoh Kasus</b> .....	<b>18</b>
<b>Bab 4</b> .....	<b>19</b>
<b>Operator</b> .....	<b>19</b>
4.1. Operator Penugasan.....	19
4.2. Operator Aritmatika.....	20
4.3. Operator Pembandingan.....	22
4.4. Operator Logika.....	24
<b>4.5. Contoh Kasus</b> .....	<b>25</b>

## **PEMROGRAMAN KOMPUTER**

Bab 5.....	27
Struktur Kontrol Percabangan .....	27
Struktur if& if...else .....	27
5.1. Struktur if .....	27
<b>5.2. Struktur if... else</b> .....	28
<b>5.3. Contoh Kasus</b> .....	30
Bab 6.....	32
Struktur Kontrol Percabangan .....	32
Struktur if...else if & if dalam if .....	32
<b>6.1. Struktur if.....else if</b> .....	32
<b>6.2. Struktur if dalam if</b> .....	34
<b>6.3. Contoh Kasus</b> .....	35
Bab 7.....	37
Struktur Kontrol Percabangan .....	37
Struktur switch case .....	37
<b>7.1. Struktur switch case</b> .....	37
<b>7.2. Contoh Kasus</b> .....	42
Bab 8.....	43
Struktur kontrol pengulangan .....	43
Struktur for .....	43
<b>8.1. Struktur For</b> .....	43
<b>8.2. Contoh Kasus</b> .....	46
Bab 9.....	47
Struktur kontrol pengulangan .....	47
Struktur while .....	47
<b>9.1. Struktur Perulangan While</b> .....	47
9.2. Contoh Kasus .....	50
Bab 10 .....	51
Struktur kontrol pengulangan .....	51
Struktur do...while .....	51
<b>10.1. Struktur PerulanganDo.....While</b> .....	51
10.2. Contoh Kasus .....	54

## **Bab 1**

### **Algoritma**

Pokok Bahasan

1. Definisi algoritma
2. Kriteria algoritma
3. Bentuk-bentuk dasar algoritma
4. Contoh kasus

Tujuan pembelajaran:

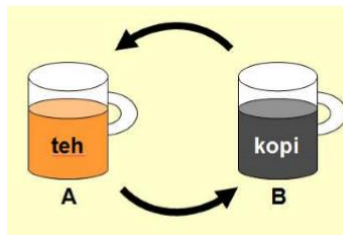
1. Menenal algoritma
2. Mengetahu kriteria algoritma
3. Mampu menyelesaikan masalah berdasarkan bentuk-bentuk algoritma

#### **1.1. Definisi algoritma**

Algoritma berasal dari nama seorang Ilmuwan Arab yang bernama Abu Jafar Muhammad Ibnu Musa Al Khuwarizmi penulis buku berjudul Al Jabar Wal Muqabala. Kata Al Khuwarizmi dibaca orang barat menjadi Algorism yang kemudian lambat laun menjadi Algorithm diserap dalam bahasa Indonesia menjadi Algoritma. Algoritma dapat diartikan urutan penyelesaian masalah yang disusun secara sistematis menggunakan bahasa yang logis untuk memecahkan suatu permasalahan.

Meski demikian terdapat beberapa definisi algoritma yang lain. Diantaranya menurut Rinaldi Munir, algoritma adalah urutan langkah-langkah logis penyelesaian masalah yang disusun secara sistematis. Sedang menurut Kamus Besar Bahasa Indonesia, definisi algoritma adalah urutan logis pengambilan keputusan untuk pemecahan masalah. Menurut tim Gunadarma:1988, algoritma adalah suatu himpunan berhingga dari instruksi-instruksi yang secara jelas memperinci langkah-langkah proses pelaksanaan, dalam pemecahan suatu masalah tertentu, atau suatu kelas masalah tertentu, dengan dituntut pula bahwa himpunan instruksi tersebut dapat dilaksanakan secara mekanik.

Untuk lebih mudah memahami arti dari algoritma dicontohkan sebuah permasalahan penukaran isi dari dua gelas. Diberikan dua buah gelas A dan B, gelas A berisi air teh dan gelas B berisi air kopi. Pertukarkan isi gelas tersebut sehingga menghasilkan gelas A yang semula berisi air teh menjadi berisi air kopi dan gelas B yang semula berisi air kopi menjadi berisi air teh. Ilustrasi permasalahan ini dapat dilihat pada Gambar 1.1.

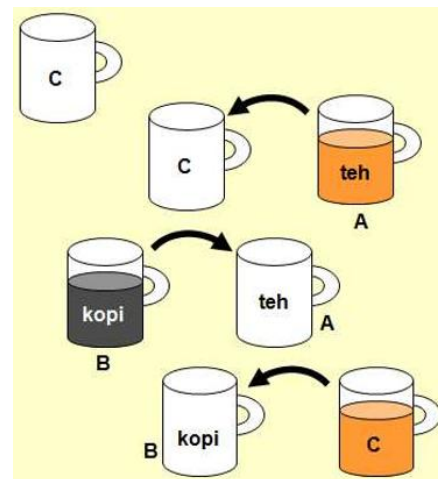


Gambar 1.1

Cara penyelesaian permasalahan ini adalah sebagai berikut. Untuk mempertukarkan isi gelas dengan benar, maka diperlukan gelas tambahan yang kita namakan gelas C sebagai tempat penampungan sementara. Berikut Algoritmanya:

1. Siapkan gelas cadangan C
2. Tuangkan air teh dari gelas A ke dalam gelas C (gelas A menjadi kosong).
3. Tuangkan air kopi dari gelas B ke dalam gelas A (gelas B menjadi kosong).
4. Tuangkan air teh dari gelas C ke dalam gelas B.

Ilustrasi langkah-langkah algoritma dapat dilihat pada Gambar 1.2



Gambar 1.2

### **1.2.Kriteria algoritma**

Menurut Donald E. Knuth, algoritma yang baik memiliki kriteria sebagai berikut :

1. Finiteness (keterbatasan), algoritma harus berakhir setelah mengerjakan sejumlah langkah proses.

2. Definiteness (kepastian), setiap langkah harus didefinisikan secara tepat dan tidak berarti ganda.
3. Input (masukan), algoritma memiliki nol atau lebih data masukan (input).
4. Output (keluaran), algoritma mempunyai nol atau lebih hasil keluaran (output).
5. Effectiveness (efektivitas), algoritma harus sangkil (efektif), langkah-langkah algoritma dikerjakan dalam waktu yang wajar.

### **1.3. Bentuk-bentuk dasar algoritma**

#### **1. Algoritma sekunsial**

Algoritma sekuensial banyak dijumpai pada kehidupan sehari-hari, misalnya pada kasus tersebut terdapat langkah-langkah yang harus dilakukan, yaitu;

- a. Membeli amplop
- b. Membeli perangko
- c. Memasang perangko ke amplop
- d. Menuliskan alamat pengirim dan alamat tujuan
- e. Pergi ke kantor pos atau memasukkan surat itu ke dalam kotak surat
- f. Surat terkirim

Pada contoh di atas dapat dilihat bahwa langkah harus dilakukan secara runtut dan ada yang tidak boleh dibolak-balik urutannya. Misalnya, amplop harus dipasang perangko terlebih dahulu baru kemudian dikirim ke kantor pos. Surat tidak dapat dikirimkan tanpa dimasukkan ke amplop yang berperangko terlebih dahulu.

#### **2. Algoritma Percabangan**

Pada contoh kasus algoritma sekuensial “mengirimkan surat” dapat dilihat bahwa pada langkah ke-5, yang mana surat yang sudah siap dikirim dapat dikirimkan dengan pergi ke kantor pos atau dengan memasukkannya ke dalam kotak pos. Pada langkah tersebut terdapat kata **ATAU**, yang berarti pemilihan solusi. Ada dua solusi pilihan akan dilakukan jika kondisinya terpenuhi, dalam arti kondisinya menghasilkan nilai benar (TRUE). Jika kondisi benar maka solusi akan

dijalankan, dan jika salah maka tidak akan dijalankan; bisa jadi menjalankan solusi lain atau tidak bertindak sama sekali.

Contoh kasus Ibu Titin memasak kentang, algoritmanya adalah sebagai berikut:

- a. Ibu Titin mengambil kantong kentang dari dalam rak.
- b. Ibu Titin mengambil panci dari almari.
- c. Menurut kebiasaan Ibu Titin, pada hari biasa Ibu Titin menggunakan baju warna cerah sehingga dia harus menggunakan celemek. Pada hari Sabtu dan Minggu dia menggunakan baju warna hitam. Maka perlu diperhatikan, sebelum mengupas kentang, apakah Ibu Titin memakai celemek atau tidak, bergantung hari apa saat itu.
- d. Ibu Titin mengupas kentang.
- e. Ibu Titin mengembalikan sisa kentang dalam kantong ke dalam rak.
- f. Ibu Titin memasak kentang.
- g. Ibu Titin menghidangkan kentang yang sudah masak.

Pada langkah ke-3 (c) di atas juga terdapat algoritma pemilihan : jika hari senin – jumat, sebelum langkah mengupas kentang, Ibu Titin harus memakai celemek untuk melindungi bajunya. Selain itu, jika hari Sabtu dan Minggu, Ibu Titin tidak perlu menggunakan celemek karena warna bajunya tidak mudah kotor. Berarti algoritma percabangan adalah algoritma di mana programmer harus memilih langkah yang harus dia lakukan berdasarkan kondisi tertentu.

### **3. Algoritma Perulangan**

Berdasarkan contoh ibu Titin sebelumnya akan ditambahkan kasus baru, yaitu ibu Titin mendapatkan pesanan masak untuk katering kentang, sehingga dia wajib mengupas kentang cukup banyak. Jumlah kentang. Berdasarkan kasus baru ini akan diformulasikan kembali langkah-langkah yang harus dilakukan Ibu Titin, yaitu:

- a. Ibu Titin mengambil kantong kentang dari dalam rak.
- b. Ibu Titin mengambil panci dari almari.
- c. Menurut kebiasaan Ibu Titin, pada hari biasa Ibu Titin menggunakan baju warna cerah sehingga dia harus menggunakan celemek, sedangkan pada hari sabtu dan minggu dia menggunakan baju warna hitam. Oleh



sebab itu perlu diperhatikan apakah sebelum yang bergantung pada hari apa saat itu.

d. Ibu Titin mengupas kentang. Karena harus mengupas sebanyak 50 buah kentang maka langkah ini akan diulang sebanyak 50 kali.

e. Setelah 50 kali mengupas kentang, Ibu Titin mengembalikan sisa kentang dalam kantong ke dalam rak.

f. Ibu Titin memasak kentang.

g. Ibu Titin menghadirkan kentang yang sudah masak

Algoritma Perulangan berarti terdapat satu atau lebih kejadian/tindakan yang harus diulang terus-menerus sampai kondisinya tidak terpenuhi lagi. Pada contoh di atas Ibu Titin harus mengulang mengupas kentang secara terus-menerus sampai kondisi tercapai, yaitu kondisi di mana kentang mencapai 50 buah.

#### **1.4. Contoh kasus**

- Buatlah suatu algoritma untuk proses pembuatan kopi yang rasa manisnya tepat
- Buatlah algoritma untuk proses aktivitas yang anda kerjakan dari pagi hingga malam hari
- Buatlah algoritma menghitung nilai rata-rata dari 3 bilangan

## **Bab 2**

### **Flowchart**

Pokok Bahasan

1. Definisi flowchart
2. Simbol-simbol Flowchart
3. Contoh kasus

Tujuan pembelajaran:

1. Menenal Flowchart
2. Menenal simbol-simbol flowchart
3. Mampu menyelesaikan masalah lewat flowchart

#### **2.1. Definisi Flowchart**

Flowchart adalah cara penulisan algoritma dengan menggunakan notasi grafis. Flowchart merupakan gambar atau bagan yang memperlihatkan urutan atau langkah-langkah dari suatu program dan hubungan antar proses beserta pernyataannya. Gambaran ini dinyatakan dengan simbol. Dengan demikian setiap simbol menggambarkan proses tertentu. Sedangkan antara proses digambarkan dengan garis penghubung. Dengan menggunakan flowchart akan memudahkan kita untuk melakukan pengecekan bagian-bagian yang terlupakan dalam analisis masalah. Disamping itu flowchart juga berguna sebagai fasilitas untuk berkomunikasi antar pemrogram yang bekerja dalam suatu proyek.

Flowchart menolong analis dan programmer untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian.

Pada dasarnya terdapat berbagai macam flowchart, diantaranya yaitu Flowchart Sistem (System Flowchart), Flowchart Paperwork / Flowchart Dokumen (Document Flowchart), Flowchart Skematik (Schematic Flowchart), Flowchart Program (Program Flowchart), Flowchart Proses (Process Flowchart). Untuk keperluan pembuatan program maka digunakan Flowchart Program.

Flowchart program menggambarkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program.

Dalam flowchart program mengandung keterangan yang lebih rinci tentang bagaimana setiap langkah program atau prosedur seharusnya dilaksanakan. Flowchart ini menunjukkan setiap langkah program atau prosedur dalam urutan yang tepat saat terjadi. Programmer menggunakan flowchart program untuk menggambarkan urutan instruksi dari program komputer. Analis Sistem menggunakan flowchart program untuk menggambarkan urutan tugas-tugas pekerjaan dalam suatu prosedur atau operasi.



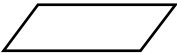
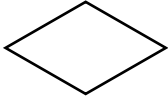




Dalam pembuatan flowchart program tidak ada rumus atau patokan yang bersifat mutlak. Karena flowchart merupakan gambaran hasil pemikiran dalam menganalisis suatu masalah yang nantinya akan diubah menjadi program komputer. Sehingga flowchart yang dihasilkan dapat bervariasi antara satu pemrogram dengan yang lainnya. Namun demikian terdapat beberapa anjuran yang harus diperhatikan, yaitu :

1. Flowchart digambarkan di suatu halaman dimulai dari sisi atas ke bawah dan dari sisi kiri kekanan.
2. Aktivitas yang digambarkan harus didefinisikan dengan menggunakan bahasa dan simbol yang tepat dan definisi ini harus dapat dimengerti oleh pembacanya.
3. Kapan aktivitas dimulai dan berakhir harus ditentukan secara jelas. Hanya terdapat satu titik awal dan satu titik akhir.
4. Setiap langkah dari aktivitas harus diuraikan dengan menggunakan deskripsi kata kerja, misalkan MENGHITUNG NILAI RATA-TARA.
5. Setiap langkah dari aktivitas harus berada pada urutan yang benar.
6. Lingkup dan range dari aktivitas yang sedang digambarkan harus ditelusuri dengan hati-hati. Percabangan-percabangan yang memotong aktivitas yang sedang digambarkan tidak perlu digambarkan pada flowchart yang sama. Simbol konektor harus digunakan dan percabangannya diletakkan pada halaman yang terpisah atau hilangkan seluruhnya bila percabangannya tidak berkaitan dengan sistem.
7. Gunakan simbol-simbol flowchart yang standar.

## **2.2. Simbol-Simbol Flowchart**

Simbol-simbol flowchart yang biasanya dipakai adalah simbol-simbol flowchart standar yang dikeluarkan oleh ANSI dan ISO. Tabel 2.1 merupakan beberapa simbol flowchart yang digunakan dalam menggambar suatu flowchart:

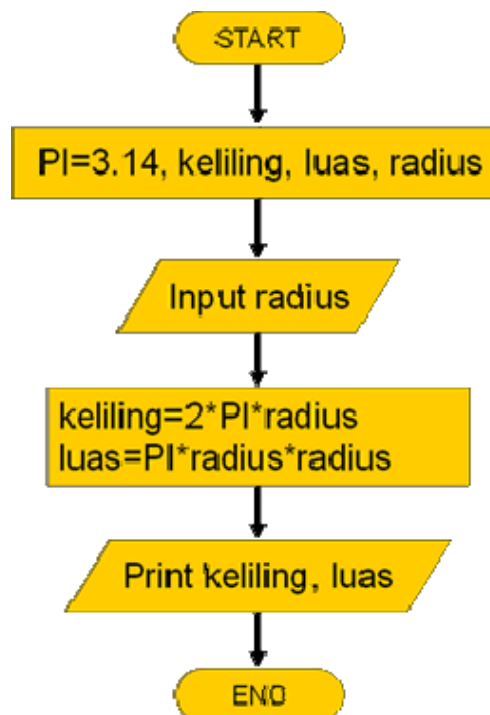
Tabel 2.1 Simbol-simbol flowchart

Simbol	Nama	Fungsi
	Terminator	Simbol Awal (Start) / Simbol Akhir (End)
	Flow Line	Simbol aliran / penghubung
	Input / Output Data	Mempresentasikan pembacaan/penulisan data
	Decision	Simbol pernyataan pilihan, berisi suatu kondisi yang selalu menghasilkan 2 nilai keluaran yaitu benar atau salah
	Preparation	Inisialisasi / pemberian nilai awal
	Predefined Process (subprogram)	Proses menjalankan sub program / fungsi / prosedur
	On Page Connector	Penghubung Flowchart pada satu halaman
	Off Page Connector	Penghubung Flowchart pada halaman berbeda

Untuk memahami lebih dalam mengenai flowchart ini, dibuat

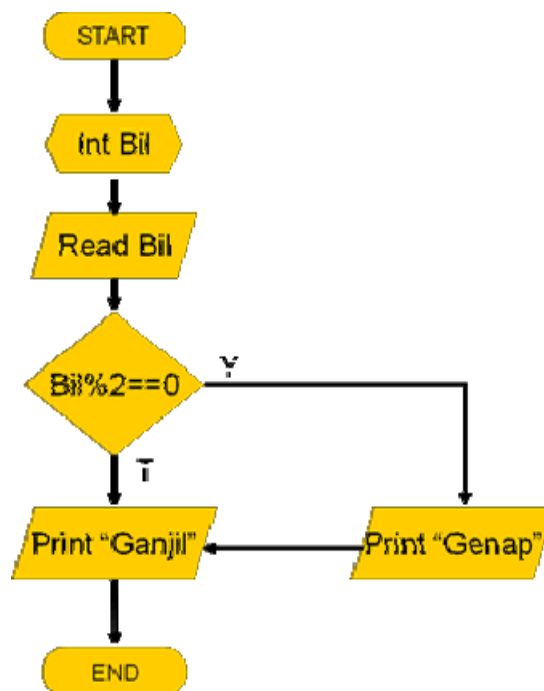
sebuahkasussederhana.

Misalnya buatlah sebuah rancangan program dengan menggunakan flowchart untukmenentukan keliling dan luas lingkaran.Perumusan untuk menentukan luas lingkaranadalah:  $luas = \pi * radius * radius$ , dan keliling lingkaran adalah  $keliling = 2 * \pi * radius$ , dengan  $\pi$  adalah sebuah konstanta 3.14. Flowchart permasalahan ini dapat dilihat di Gambar2.1.



Gambar 2.1. Flowchart luas dan keliling lingkaran

Selanjutnya akan dibuat contoh flowchart yang mengandung percabangan atau decision. Misalnya untuk permasalahan penentuan apakah suatu bilangan termasuk bilangan ganjil atau genap. Cara menyelesaikan permasalahan ini adalah dengan membagi bilangan dengan angka 2. Jika nilai sisa pembagian nya adalah 0 maka bilangan tersebut adalah bilangan genap, jika nilai sisa pembagiannya adalah 1 maka bilangan tersebut adalah bilangan ganjil. Operasi aritmatika yang digunakan untuk menentukan nilai sisa pembagian adalah operasi modulo (%). Flowchart permasalahan ini dapat dilihat di Gambar 2.2. Dalam hal ini Bil adalah bilangan yang akan di cek status ganjil atau genapnya.



Gambar 2.2. Flowchart Penentuan Bilangan Ganjil-Genap

## 2.2 Contoh Kasus

1. Buatlah algoritma untuk menentukan nilai terbesar dari bilangan bulat yang dibaca dari keyboard dan menuliskan hasilnya ke layar! Algoritma dibuat dalam bentuk kalimat deskriptif dan flowchart.
2. Buat algoritma dalam bahasa deskriptif dan flowchart untuk kegiatan mengambil dan menabung uang di bank melalui teller!
3. Buat algoritma dalam flowchart untuk menentukan apakah suatu bilangan merupakan bilangan genap atau ganjil!

## **Bab 3**

### **Tipe Data, Variabel & Konstanta**

Pokok Bahasan

1. Tipe Data
2. Variabel & Konstanta
3. Deklarasi
4. Contoh kasus

Tujuan pembelajaran:

1. Mengetahui tipe data
2. Mengetahui variabel dan konstanta
3. Mengetahui cara mendeklarasikan variabel dan konstanta
4. Mampu membuat program dengan menggunakan variabel dan konstanta

#### **3.1. Tipe Data**

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh computer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

Tabel 3.1 Bentuk Tipe data

No	Tipe Data	Ukuran	Range (Jangkauan)	Format	Keterangan
1	Char	1 byte	-128 s/d 127	%c	Karakter
2	Int	2 byte	-32768 s/d 32767	%i, %d	Bilangan bulat
3	Float	4 byte	-3.4E-38 s/d 3.4E+38	%f	Bilangan pecahan
4	Double	8 byte	1.7E-308 s/d 1.7E+308	%lf	Pecahan presisi ganda
5	Void	0 byte	-	-	Tidak bertipe

### **3.2. Variabel & Konstanta**

Konstanta dan variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu didalam proses program. Berbeda dengan konstanta yang nilainya tidak bisa diubah atau selalu tetap selang eksekusi berlangsung, nilai dari suatu variabel dapat berubah sesuai kebutuhan. Konstanta dan variabel merupakan tempat di memori komputer untuk menyimpan data berupa nilai dengan tipe data tertentu. Konstanta dan variabel harus diberi nama sebagai identifikasi.

Secara logika dapat dibayangkan sebuah konstanta atau variabel sebagai sebuah kotak kosong yang dapat diisi dengan sesuatu tipe data tertentu, misal kita membuat sebuah variabel berupa bilangan bulat, maka dalam logika, kita sedang membuat kotak kosong yang hanya dapat diisi dengan kertas bertuliskan bilangan bulat, tidak boleh jenis bilangan selain bilangan bulat. Ilustrasi logika ini dapat dilihat pada Gambar 3.1. Pada Gambar 3.1 dimisalkan membuat dua buah konstanta atau variabel dengan nama identifier **nilai** dan **X** yang masing-masing dapat digunakan untuk menyimpan suatu nilai dalam memori sesuai dengan tipe data yang telah ditentukan.

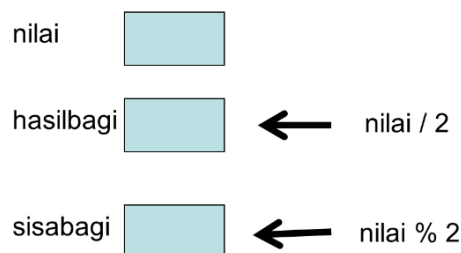


Gambar 3.1. Ilustrasi pembuatan konstanta atau variabel

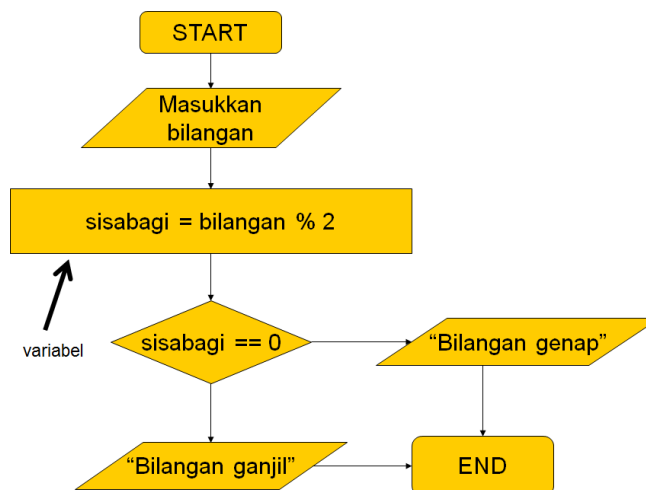
Semisal **nilai** di Gambar 3.1 adalah sebuah variabel. Maka variabel nilai yang telah dibuat ini selanjutnya dapat digunakan dalam program, semisal dilakukan operasi aritmatika berupa operasi pembagian (/) atau modulo (%). Gambar 3.2 menunjukkan ilustrasi operasi yang terjadi pada variabel nilai. Dalam hal ini variabel nilai dibagi dengan angka 2, dan hasil operasi pembagian disimpan dalam variabel baru yang bernama **hasilbagi**. Operasi aritmatika lain yang terjadi



pada Gambar 3.2 adalah variabel nilai dimodulo dengan angka 2 dan hasilnya disimpan dalam variabel baru yang bernama **sisabagi**. Operasi pada suatu variabel tidak hanya terbatas pada operasi aritmatika saja, tetapi juga dapat berupa operasi perbandingan. Misalnya nilai apakah lebih besar dari angka 10 ( $\text{nilai} > 10$ ) dan lain- lain. Contoh penggunaan variabel nilai dalam permasalahan sederhana adalah penentuan suatu bilangan termasuk dalam kategori ganjil atau genap seperti terlihat pada Gambar3.3.



Gambar 3.2. Ilustrasi penggunaanvariabel



Gambar 3.3. Contoh penggunaanvariabel

### 3.3. Deklarasi

#### Deklarasi variabel

Deklarasi diperlukan bila kita akan menggunakan pengenal (identifier) dalam

program.

Bentuk umum pendeklarasian suatu variable

adalah : Nama\_tipe nama\_variabel;

Contoh :

`int x; // Deklarasi x bertipe integer`

`char y, huruf, nim[10]; // Deklarasi variable bertipe char`

`float nilai; // Deklarasi variable bertipe float`

Contoh program 1 : membuat program dengan menggunakan tipe data int untuk mengalikan dua buah bilangan

Algoritma :

1. Masukkan  $x = 2$
2. Masukkan  $y = 8$
3. Hitung  $z$  dengan rumus  $z = x * y$
4. Tampilkan  $z$

Kode sumber :

```
#include<stdio.h>

main()
{
    //Deklarasi Variabel
    int x;
    int y,z;
    //input
    x=2;
    y=8;
    z=x*y;
    printf("Nilai z = %d",z);

}
```

Contoh program 2: sama dengan program pertama tetapi di modifikasi dengan melakukan input lewat keyboard

Kode sumber :

```
#include<stdio.h>
main()
{
    //Deklarasi Variabel
    int x;
    int y,z;
    //input
    printf("Masukkan nilai x = ");
    scanf("%d",&x);
    printf("Masukkan nilai y = ");
    scanf("%d",&y);
    z=x*y;
    printf("Nilai z = %d",z);
}
```

Contoh program 3: melakukan operasi penjumlahan dan pengurangan dari 3 buah data.

Algoritma :

1. Masukkan nilai A dan B
2. Hitung dengan menggunakan rumus Hasil = A – B + 15
3. Tampilkan nilai Hasil

Kode sumber :

```
#include<stdio.h>
main()
{
    int Hasil,A,B;
    printf("A = "); scanf("%d",&A);
    printf("B = "); scanf("%d",&B);
```

```
    Hasil=A-B+15;
    printf("Hasilnya adalah = %d.",Hasil);
}
```

Contoh program 4: Membaca dan menampilkan karakter huruf yang diinputkan lewat keyboard

Algoritma :

1. Masukkan huruf
2. Tampilkan huruf

Kode Sumber

```
//Program untuk jenis data karakter
#include<stdio.h>
main()
{
    char huruf;
    printf("Masukkan sembarang huruf = ");
    scanf("%c",&huruf);
    printf("Huruf yang dimasukkan adalah = %c",huruf);
}
```

### **DeklarasiKonstanta**

Dalam bahasa C konstanta dideklarasikan menggunakan preprocessor #define.

Contohnya :

```
#define PHI3.14
#define nim "0111500382"
#define nama "SriWidhiyanti"
```

Contoh program 5: Membuat program untuk menghitung luas lingkaran

Algoritma :

1. Tentukan konstanta  $\phi = 3.14$
2. Masukkan r

3. Hitung luas dengan rumus  $Luas = \phi * r * r$

4. Cetak Luas

Kode sumber :

```
#include<stdio.h>
main()
{
    float Luas,r,x=2;
    const float phi=3.14;
    //input
    r=x;

    printf("radius = ");
    scanf("%f",&r);

    Luas=phi*r*r;
    printf("Luas Lingkaran = %g",Luas);
}
```

Contoh program 6: Konversi satuan panjang. Program untuk konversikan panjang dalam yard, kaki, dan inch ke dalam meter

1 inch = 0,0254 meter

1 kaki = 0,3048 meter

1 yard = 0,9144 meter

Kostanta :

mi = 0,0254

mk = 0,3048

my = 0,9144

Rumus :  $meter = my * yard + mk * kaki + mi * inch$

Algoritma :

1. Tentukan konstanta  $mi = 0,0254$ ,  $mk = 0,3048$ ,  $my = 0,9144$
2. Masukkan yard, kaki dan inch
3. Hitung konversi dengan rumus  $meter = my * yard + mk * kaki + mi * inch$
4. Cetak meter

Kode sumber :

```
#include<stdio.h>

main()
{
    //deklarasi konstanta
    const float mi=0.0254, mk=0.3048, my=0.9144;
    //deklarasi variabel input dan output
    float yard, kaki, inch, meter;
    //input data
    printf("masukkan yard = ");
    scanf("%f",&yard);
    printf("masukkan kaki = ");
    scanf("%f",&kaki);
    printf("masukkan inch = ");
    scanf("%f",&inch);
    //proses
    meter=my*yard+mk*kaki+mi*inch;
    printf("%g yard %g kaki %g inch = %g
meter",yard,kaki,inch,meter);
}
```

### **3.4. Contoh Kasus**

1. Buatlah program untuk menghitung luas persegi panjang
2. Buatlah program untuk menghitung jarak tempuh dengan formula  $s = v * t$
3. Buatlah program untuk menghitung konversi suhu dari Celcius ke Fahrenheit
4. Buatlah program konversi detik ke hari, jam, menit, detik
5. Buatlah program untuk menghitung luas & keliling lingkaran dengan nilai phi tetapkan sebagai konstanta
6. Buatlah program untuk menghitung Energi dengan formula  $E=mc^2$  dengan  $c = 2,99 \times 10^8$
7. Buatlah program untuk menghitung gravitasi newton dari 3 buah benda

## **Bab 4**

### **Operator**

#### **Pokok Bahasan**

1. Operator Penugasan
2. Operator Aritmatika
3. Operator Pembanding
4. Operator Logika
5. Contoh kasus

#### **Tujuan pembelajaran:**

1. Mampu menggunakan operator penugasan
2. Mampu menggunakan operator aritmatika
3. Mampu menggunakan operator pembanding
4. Mampu menggunakan operator logika
5. Mampu membuat program dengan menggunakan operator.

#### **4.1. Operator Penugasan**

Operator penugasan (assignment operator) adalah operator yang digunakan untuk memberi nilai pada sebuah variabel. Operator penugasan dilambangkandengan tanda sama dengan (=).

Contoh :

$A = 6$

$A = B$

$C = A + B$

Nilai di sebelah kanan bisa berupa nilai data, variabel atau ekspresi. Berikut adalah contoh program dengan menggunakan operator penugasan :

Contoh program 1: program yang menggunakan dua buah variabel dalam operasi yang menggunakan operator penugasan

Algoritma :

1. Masukkan nilai A dan B
2. Lakukan operasi  $A = B$

### 3. Cetak A dan B

Kode sumber:

```
#include<stdio.h>
main()
{
    int A,B;
    printf("A = "); scanf("%d",&A);
    printf("B = "); scanf("%d",&B);
    A = B;
    printf("New value\n");
    printf("%d\n",A);
    printf("%d",B);
}
```

#### **4.2. Operator Aritmatika**

Bahasa C menyediakan lima operator aritmatika, yaitu :

- \* : untuk perkalian
- / : untuk pembagian
- % : untuk sisa pembagian(modulus)
- + : untuk penjumlahan
- - : untuk pengurangan

Contoh program 2: membuat program yang menggunakan operator aritmatika yaitu \*, /, +, -

Algoritma :

1. Masukkan x dan y
2. Hitung  $z=x+y$
3. Cetak z
4. Hitung  $z=x-y$
5. Cetak z
6. Hitung  $z=x*y$
7. Cetak z



8. Hitung  $z=x/y$

9. Cetak z

Kode sumber :

```
#include<stdio.h>
main()
{
    float x,y,z;
    printf("x = ");
    scanf("%f",&x);
    printf("y = ");
    scanf("%f",&y);
    z=x+y;
    printf("%g + %g = %g\n",x,y,z);
    z=x-y;
    printf("%g - %g = %g\n",x,y,z);
    z=x*y;
    printf("%g * %g = %g\n",x,y,z);
    z=x/y;
    printf("%g / %g = %g",x,y,z);
}
```

Contoh program 3 : buatlah program yang melakukan operasi modulus

Algoritma :

1. Masukkan variabel A dan B
2. Lakukan operasi  $C = A \% B$
3. Cetak C

Kode sumber :

```
#include<stdio.h>
main()
{
    int A,B,C;
    printf("A = ");
```

```
scanf ("%d", &A) ;  
printf ("B = ");  
scanf ("%d", &B) ;  
C=A%B;  
printf ("%d %% %d = %d", A, B, C) ;  
  
}
```

Contoh program 4 : buatlah program untuk menggabungkan operator penugasan an operator aritmatika

Algortima :

1. Inisialisasikan x = 1
2. Masukkan y
3. Hitung x dengan rumus x += y
4. Cetak x

Kode sumber :

```
#include<stdio.h>  
main()  
{  
    int x,y;  
    x=1;  
    printf("y = ");  
    scanf ("%d", &y) ;  
    x+=y;  
    printf("x = %d", x) ;  
}
```

#### **4.3. Operator Pembandingan**

Operator hubungan digunakan untuk membandingkan hubungan antara dua buah operand /sebuah nilai atau variable. Operasi majemuk seperti pada tabel dibawah ini:

**Tabel 4.1**Operator Hubungan

Operator	Arti	Contoh	
<	Kurang dari	X<Y	Apakah X kurang dari Y
<=	Kurang dari sama dengan	X<=Y	Apakah X Kurang dari sama dengan Y
>	Lebih dari	X>Y	Apakah X Lebih dari Y
>=	Lebih dari sama dengan	X==Y	Apakah X Lebih dari sama dengan Y
==	Sama dengan	X==Y	Apakah X Sama dengan Y
!=	Tidak sama dengan	X!= Y	Apakah X Tidak sama denganY

Contoh program 5 : buatlah program yang menggunakan operator pembandingan

Algoritma :

1. Masukkan a dan b
2. Hitung c=a<b
3. Cetak c
4. Hitung c=a>b
5. Cetak c
6. Hitung c=a<=b
7. Cetak c
8. Hitung c=a>=b
9. Cetak c
10. Hitung c=a==b
11. Cetak c
12. Hitung c=a!=b
13. Cetak c

Kode sumber :

```
#include<stdio.h>
main()
{
```

```
int a,b,c;
printf("a = "); scanf("%d",&a);
printf("b = "); scanf("%d",&b);
c=a<b;
printf("%d < %d = %d\n",a,b,c);
c=a>b;
printf("%d > %d = %d\n",a,b,c);
c=a<=b;
printf("%d <= %d = %d\n",a,b,c);
c=a>=b;
printf("%d >= %d = %d\n",a,b,c);
c=a==b;
printf("%d == %d = %d\n",a,b,c);
c=a!=b;
printf("%d != %d = %d\n",a,b,c);
}
```

#### **4.4. Operator Logika**

Jika operator hubungan membandingkan hubungan antara dua buah operand, maka operator logika digunakan untuk membandingkan logika hasil dari operator-operator hubungan.

Operator logika ada tiga macam, yaitu :

- && : Logika AND(DAN)
- || : Logika OR(ATAU)
- ! : Logika NOT (INGKARAN)

Operasi AND akan bernilai benar jika dua ekspresi bernilai benar. Operasi OR akan bernilai benar jika dan hanya jika salah satu ekspresinya bernilai benar. Sedangkan operasi NOT menghasilkan nilai benar jika ekspresinya bernilai salah, dan akan bernilai salah jika ekspresinya bernilaibenar.

Contoh program 6 : buatlah program yang menggunakan operator logika AND dan NOT

Algortima :

1. Inisialisasikan a=1 dan b=0
2. Hitung c=a&&a

3. Cetak c
4. Hitung  $c=a\&\&b$
5. Cetak c
6. Hitung  $c=b\&\&a$
7. Cetak c
8. Hitung  $c=b\&\&b$
9. Cetak c
10. Hitung  $c=!a$
11. Cetak c
12. Hitung  $c=!b$
13. Cetak c

Kode sumber:

```
#include<stdio.h>
main()
{
    int a=1,b=0,c;
    c=a&&a;
    printf("c = %d\n",c);
    c=a&&b;
    printf("c = %d\n",c);
    c=b&&a;
    printf("c = %d\n",c);
    c=b&&b;
    printf("c = %d\n",c);
    c=!a;
    printf("c = %d\n",c);
    c=!b;
    printf("c = %d\n",c);
}
```

#### **4.5. Contoh Kasus**

1. Buatlah program untuk menghitung formula berikut ini

$$hasil = \frac{\frac{a}{b} + 6}{x - \frac{y}{z}}$$

2. Buatlah program untuk menghitung sisi miring dari sebuah segitiga siku-siku
3. Buatlah program untuk menghitung hasil akhir dari rumus berikut
  - a.  $C = (A + B - 2) \neq (B * 20)$
  - b.  $C = (14 - A) \leq (B * 5)$
4. Buatlah program berikut yang menggunakan operator logika
  - a.  $z = a < 2 * y \parallel x \neq y$
  - b.  $c = !(a == b) \&\& (a > b);$

## **Bab 5**

### **Struktur Kontrol Percabangan**

#### **Struktur if & if...else**

Pokok Bahasan

1. Struktur if
2. Struktur if...else
3. Contoh kasus

Tujuan pembelajaran:

1. Menenal dan mampu menggunakan struktur if
2. Menenal dan mampu menggunakan struktur if...else
3. Mampu membuat program dengan menggunakan struktur kontrol percabangan if dan if...else

Di dalam pemrograman terstruktur, semua perintah dikerjakan secara sekuensial dari atas ke bawah secara berurutan, sehingga disebut aliran sequence. Namun ada kalanya kita harus memilih dari alternatif-alternatif yang ada sesuai dengan kondisi yang ada. Struktur kontrol percabangan disebut juga penyeleksian kondisi.

Penyeleksian kondisi digunakan untuk mengarahkan perjalanan suatu proses. Penyeleksian kondisi dapat diibaratkan sebagai katup atau kran yang mengatur jalannya air. Bila katup terbuka maka air akan mengalir dan sebaliknya bila katup tertutup air tidak akan mengalir atau akan mengalir melalui tempat lain. Fungsi penyeleksian kondisi penting artinya dalam penyusunan bahasa C, terutama untuk program yang kompleks.

#### **5.1. Struktur if**

Struktur if dibentuk dari pernyataan if dan sering digunakan untuk menyeleksi suatu kondisi tunggal. Bila proses yang diseleksi terpenuhi atau bernilai benar, maka pernyataan yang ada di dalam blok if akan diproses dan dikerjakan.

Bentuk umum struktur kondisi if adalah :

```
if(kondisi)
    pernyataan;
```

Contoh program 1 :Menampilkan bilangan absolut dari sebuah bilangan negatif

Algoritma :

1. Inisialisasikan bil = -54,321
2. Jika bil < 0 maka hitung absolut = -bil

## **PEMROGRAMAN KOMPUTER**

3. Cetak absolut

4. Akhir jika

Kode sumber :

```
#include<stdio.h>
main()
{
    float bil,absolut;
    bil=-54.321;
    if(bil<0)
    {
        absolut=-bil;
        printf("Angka mutlak dari %.3f = %.3f",bil,absolut);
    }
}
```

Contoh program 2: Jika misalnya x bernilai 100 dan y bernilai 10, maka tulisan “x lebih besar y” akan ditampilkan.

Algoritma :

1. Inisialisasikan x = 100
2. Inisialisasikan y = 10
3. Jika  $x > y$  maka cetak “x lebih besar y”
4. Akhir jika

Kode sumber :

```
#include<stdio.h>
main()
{
    int x,y;
    x=100;
    y=10;
    if(x>y)
        printf("x lebih besar y");
}
```

### **5.2. Struktur if... else**

Dalam struktur kondisi if.....else minimal terdapat dua pernyataan. Jika kondisi yang diperiksa bernilai benar atau terpenuhi maka pernyataan pertama yang dilaksanakan dan jika kondisi yang diperiksa bernilai salah maka pernyataan yang kedua yang dilaksanakan.



## **PEMROGRAMAN KOMPUTER**

Bentuk umumnya adalah sebagai berikut:

```
if(kondisi)
    pernyataan-1
else
    pernyataan-2
```

Contoh program 3: Menampilkan bilangan terbesar dari dua bilangan

Algoritma :

1. Masukkan A dan B
2. Jika  $A > B$  maka cetak A Terbesar
3. Jika sebaliknya maka cetak B Terbesar
4. Akhir jika

Kode sumber :

```
#include<stdio.h>
main()
{
    int A,B;
    printf("A = "); scanf("%d",&A);
    printf("B = "); scanf("%d",&B);
    if(A>B)
        printf("%d Terbesar",A);
    else
        printf("%d terbesar",B);
}
```

Contoh program 4: Memilih menampilkan tahanan seri atau paralel dari 3 tahanan

Algoritma :

1. Masukkan R1, R2 dan R3
2. Masukkan pilih dengan karakter s atau yang lain
3. Jika pilih == 's' atau pilih == 'S' maka hitung tahanan seri dengan rumus  
 $RS = R1 + R2 + R3$
4. Cetak RS
5. Jika sebaliknya maka hitung tahanan seri dengan rumus  
 $RP = 1 / (1/R1 + 1/R2 + 1/R3)$
6. Cetak RP

## **PEMROGRAMAN KOMPUTER**

Kode sumber :

```
#include<stdio.h>
main()
{
    float R1,R2,R3,RS,RP;
    char pilih;
    printf("R1 = ");
    scanf("%f",&R1);
    printf("R2 = ");
    scanf("%f",&R2);
    printf("R3 = ");
    scanf("%f",&R3);
    printf("Daftar Menu :\n");
    printf("(s). Tahanan Seri\n");
    printf("(p). Tahanan Paralel\n");
    printf("Hitung Seri atau paralel? ");
    scanf("%c%c",&pilih,&pilih);
    if(pilih=='s' || pilih=='S')
    {
        RS=R1+R2+R3;
        printf("R Seri = %g",RS);
    }
    else
    {
        RP=1/(1/R1+1/R2+1/R3);
        printf("R Paralel = %g",RP);
    }
}
```

### **5.3. Contoh Kasus**

1. Buatlah program untuk menyatakan seorang mahasiswa dapat lulus jika nilai tugasnya diatas 60 dan nilai ujian diatas 70
2. Buatlah program untuk memeriksa bilangan genap atau ganjil
3. Buatlah program untuk menentukan apakah sebuah segitiga sama sisi atau tidak sama sisi

## **PEMROGRAMAN KOMPUTER**

4. Buatlah program untuk menentukan huruf vokal atau konsonan

## **Bab 6**

### **Struktur Kontrol Percabangan**

#### **Struktur if...else if & if dalam if**

Pokok Bahasan

1. Struktur if...else if
2. Struktur if dalam if
3. Contoh kasus

Tujuan pembelajaran:

1. Menenal dan mampu menggunakan struktur if...else if
2. Menenal dan mampu menggunakan struktur if dalam if
3. Mampu membuat program dengan menggunakan struktur kontrol percabangan if...else if dan if dalam if

#### **6.1. Struktur if.....else if**

Struktur kondisi if ..... else if diperuntukkan bagi 3 kondisi atau lebih. Jika kondisi pertama bernilai benar maka pernyataan pertama yang akan dilaksanakan, tetapi jika kondisi pertama bernilai salah maka masih ada pilihan kedua atau ketiga dan seterusnya yang akan di cek yang bernilai benar.

Bentuk umumnya adalah sebagai berikut:

```
if(kondisi1)
    pernyataan-1
else if(kondisi2)
    pernyataan-2
else
    pernyataan-3
```

Contoh program 1: Buatlah program untuk menentukan apakah bilangan yang dimasukkan lewat keyboard adalah angka 0 atau bilangan positif atau juga bilangan negatif

Algoritma :

1. Masukkan angka
2. Jika angka  $> 0$  maka cetak "Bilangan Positif"
3. Sebaliknya jika angka  $< 0$  maka cetak "Bilangan Negatif"
4. Jika sebaliknya dari pilihan dua kondisi di atas maka cetak "Bilangan Nol (0)"

## **PEMROGRAMAN KOMPUTER**

### 5. Akhir jika

Kode sumber :

```
#include<stdio.h>

main()
{
    float angka;
    printf("Masukkan sembarang angka = ");
    scanf("%f",&angka);
    if(angka>0)
        printf("%g Bilangan Positif",angka);
    else if(angka<0)
        printf("%g Bilangan Negatif",angka);
    else
        printf("Bilangan Nol(0)");
}
```

Contoh program 2: program untuk memilih salah satu dari 4 prodi tetapi jika sebaliknya tidak memilih dari ke 4 prodi maka akan tampil pesan “kode prodi salah”

Algoritma :

1. Masukkan Prodi
2. Jika Prodi == 1 maka cetak pesan D3 T. Listrik
3. Sebaliknya Jika Prodi == 2 maka cetak pesan D4 T. Listrik
4. Sebaliknya Jika Prodi == 3 maka cetak pesan D3 T. Komputer
5. Sebaliknya Jika Prodi == 4 maka cetak pesan D4 T. Informatika
6. Sebaliknya selain dari 4 pilihan di atas maka cetak pesan Kode Prodi Salah

Kode sumber :

```
#include<stdio.h>

main()
{
    int Prodi;
    printf("Apa pilihan prodi anda ? ");
    scanf("%d",&Prodi);
    printf("Pilihan anda adalah :\n");
    if(Prodi==1)
```

## **PEMROGRAMAN KOMPUTER**

```
        printf("D3 T. Listrik");
    else if (Prodi==2)
        printf("D4 T. Listrik");
    else if (Prodi==3)
        printf("D3 T.Komputer");
    else if (Prodi==4)
        printf("D4 T.Informatika");
    else
        printf("Kode Prodi Salah");
}
```

### **6.2. Struktur if dalam if**

Struktur kondisi if dalam if digunakan untuk penyeleksian bersarang. Bentuk ini dianggap sebagai percabangan komplek, karena pemilihan tidak hanya satu tetapi bisa tersiri atas banyak alternatif. Bentuk umumnya adalah sebagai berikut:

```
if(kondisi1)
    if(kondisi2)
        pernyataan-1
        pernyataan-2
    else
        pernyataan-3
else
    pernyataan-4
```

Contoh program 3: buatlah program untuk menentukan apakah mahasiswa dinyatakan LULUS, MENGULANG atau TIDAK LULUS

Algoritma :

1. Masukkan nilai data dari variabel ujian
2. Jika ujian == 'Y' atau ujian == 'y' maka masukkan nilai
3. Jika nilai  $\geq 60$  maka cetak pesan "Anda LULUS"
4. Jika sebaliknya maka cetak pesan "Anda diberi kesempatan MENGULANG"
5. Akhir jika
6. Jika sebaliknya maka cetak pesan "Anda TIDAK LULUS"
7. Akhir jika

Kode sumber :

```
#include<stdio.h>
```

## **PEMROGRAMAN KOMPUTER**

```
main()
{
    char ujian;
    float nilai;
    printf("Apakah Anda ikut ujian ? ");
    scanf("%c",&ujian);
    if(ujian=='Y' || ujian=='y')
    {
        printf("Berapa nilai Anda ? ");
        scanf("%f",&nilai);
        if(nilai>=60)
            printf("Anda LULUS");
        else
            printf("Anda diberi kesempatan
            MENGULANG");
    }
    else
        printf("Anda TIDAK LULUS");
}
```

### **6.3. Contoh Kasus**

1. Buatlah program yang menerima input 3 buah panjang sisi segitiga kemudian :
  - a. Jika segitiga tersebut sama sisi, katakan SEGITIGA SAMA SISI
  - b. Jika segitiga tersebut sama kaki, katakan SEGITIGA SAMA KAKI
  - c. Jika bukan keduanya, katakan SEGITIGA SEMBARANG
2. Mencari bilangan terbesar dari 3 bilangan
3. Buatlah program diskon :

Input adalah jumlah total pembelian :

Jika total pembelian  $\geq 500000$  maka bonusnya setrika

Jika total pembelian  $\geq 100000$  maka bonusnya payung

Jika total pembelian  $\geq 50000$  maka bonusnya ballpoint

Selain itu tidak mendapat bonus.
4. Buatlah konversi nilai huruf dari nilai yang dimasukkan user!

$80 < A \leq 100$

$65 < B \leq 80$

## **PEMROGRAMAN KOMPUTER**

$$55 < C \leq 65$$

$$40 < D \leq 55$$

$$0 < E \leq 40$$

5. Membuat konversi dari bilangan yang dimasukkan sebagai berikut:

Misal 101 : bilangan ratusan

Misal 1200 : bilangan ribuan

Misal 11 : bilangan puluhan



## **Bab 7**

### **Struktur Kontrol Percabangan**

#### **Struktur switch case**

Pokok Bahasan

1. Struktur switch case
2. Contoh kasus

Tujuan pembelajaran:

1. Mengetahui dan mampu menggunakan struktur switch case
2. Mampu membuat program dengan menggunakan struktur kontrol percabangan switch case

#### **7.1. Struktur switch case**

Struktur kondisi switch....case....default digunakan untuk penyeleksian kondisi dengan kemungkinan yang terjadi cukup banyak. Struktur ini akan melaksanakan salah satu dari beberapa pernyataan 'case' tergantung nilai kondisi yang ada di dalam switch. Selanjutnya proses diteruskan hingga ditemukan pernyataan 'break'. Jika tidak ada nilai pada case yang sesuai dengan nilai kondisi, maka proses akan diteruskan kepada pernyataan yang ada di bawah 'default'.

Bentuk umum dari struktur kondisi ini adalah :

```
switch(kondisi)
{
    case 1 : pernyataan-1;
    break;
    case 2 : pernyataan-2;
    break;
    ....
    ....
    case n : pernyataan-n;
    break;
    default : pernyataan-m
}
```

Contoh program 1 : Buatlah program yang membaca data kode hari dari keyboard dan

## **PEMROGRAMAN KOMPUTER**

kemudian menampilkan nama hari.

Algoritma :

1. Masukkan hari
2. Jika hari sama dengan 1 maka cetak “Senin”
3. Jika hari sama dengan 2 maka cetak “Selasa”
4. Jika hari sama dengan 3 maka cetak “Rabu”
5. Jika hari sama dengan 4 maka cetak “Kamis”
6. Jika hari sama dengan 5 maka cetak “Jumat”
7. Jika hari sama dengan 6 maka cetak “Sabtu”
8. Jika hari sama dengan 7 maka cetak “Minggu”
9. Sebaliknya cetak “kode hari salah”

Kode sumber :

```
#include<stdio.h>
main()
{
    int hari;
    printf("Hari ini adalah hari? ");
    scanf("%d",&hari);

    switch(hari)
    {
        case 1:
            printf("Hari Senin");
            break;
        case 2:
            printf("Hari Selasa");
            break;
        case 3:
            printf("Hari Rabu");
            break;
        case 4:
            printf("Hari Kamis");
            break;
        case 5:
```

## **PEMROGRAMAN KOMPUTER**

```
        printf("Hari Jum'at");
        break;
    case 6:
        printf("Hari Sabtu");
        break;
    case 7:
        printf("Hari Minggu");
        break;
    default:
        printf("Kode hari salah");
}

}
```

Contoh program 2 : Membuat program kalkulator sederhana untuk memilih operasi perkalian, pembagian, penjumlahan atau pengurangan.

Algoritma :

1. Inisialisasikan valid\_operator = 1
2. Masukkan bil1, operator dan bil2
3. Jika operator sama dengan '\*' maka hitung hasil = bil1 \* bil2
4. Jika operator sama dengan '/' maka hitung hasil = bil1 / bil2
5. Jika operator sama dengan '+' maka hitung hasil = bil1 + bil2
6. Jika operator sama dengan '-' maka hitung hasil = bil1 - bil2
7. Sebaliknya maka valid\_operator = 0
8. Akhir jika
9. Jika valid\_operator bernilai TRUE maka cetak hasil
10. Jika sebaliknya maka cetak pesan "Invalid Operator"

Kode sumber :

```
#include<stdio.h>

main()
{
    int valid_operator=1;
    char Operator;
    float bil1,bil2,hasil;
```

## **PEMROGRAMAN KOMPUTER**

```
printf("Masukkan 2 buah bilangan & operator\n");
printf("dengan format : bil1 Operator bil2\n");
scanf("%f %c %f",&bil1,&Operator,&bil2);

switch(Operator)
{
    case '*':
        hasil=bil1*bil2;
        break;
    case '/':
        hasil=bil1/bil2;
        break;
    case '+':
        hasil=bil1+bil2;
        break;
    case '-':
        hasil=bil1-bil2;
        break;
    default:
        valid_operator=0;
}
if(valid_operator)
    printf("\n%g          %c          %g          =
    %g\n",bil1,Operator,bil2,hasil);
else
    printf("Invalid Operator");
}
```

Contoh program 3 : Buatlah program yang meminta data bulan (1...12) dimasukkan dari keyboard dan kemudian menentukan jumlah hari dalam bulan tersebut.

Algoritma :

1. Masukkan kode\_bulan
2. Jika kode\_bulan sama dengan 2 maka cetak pesan “Jumlah hari 28 atau 29”
3. Jika kode\_bulan sama dengan 1 maka cetak pesan “Jumlah hari 31”

## **PEMROGRAMAN KOMPUTER**

4. Jika kode\_bulan sama dengan 3 maka cetak pesan “Jumlah hari 31”
5. Jika kode\_bulan sama dengan 5 maka cetak pesan “Jumlah hari 31”
6. Jika kode\_bulan sama dengan 7 maka cetak pesan “Jumlah hari 31”
7. Jika kode\_bulan sama dengan 8 maka cetak pesan “Jumlah hari 31”
8. Jika kode\_bulan sama dengan 10 maka cetak pesan “Jumlah hari 31”
9. Jika kode\_bulan sama dengan 12 maka cetak pesan “Jumlah hari 31”
10. Jika kode\_bulan sama dengan 4 maka cetak pesan “Jumlah hari 30”
11. Jika kode\_bulan sama dengan 6 maka cetak pesan “Jumlah hari 30”
12. Jika kode\_bulan sama dengan 9 maka cetak pesan “Jumlah hari 30”
13. Jika kode\_bulan sama dengan 11 maka cetak pesan “Jumlah hari 30”
14. Jika sebaliknya maka cetak pesan “Salah kode bulan”

Kode sumber :

```
#include<stdio.h>
main()
{
    int kode_bulan;
    printf("Masukkan kode bulan (1...12) : ");
    scanf("%d",&kode_bulan);
    switch(kode_bulan)
    {
        case 2:
            printf("Jumlah hari 28 atau 29");
            break;
        case 1:
        case 3:
        case 5:
        case 7:
        case 10:
        case 12:
            printf("Jumlah hari 31");
            break;
        case 4:
        case 6:
        case 9:
```

## **PEMROGRAMAN KOMPUTER**

```
        case 11:
            printf("Jumlah hari 30");
            break;
        default:
            printf("Salah kode bulan");
    }
}
```

### **7.2. Contoh Kasus**

6. Buatlah program untuk menyatakan seorang mahasiswa dapat lulus jika nilai tugasnya diatas 60 dan nilai ujian diatas 70
7. Buatlah program untuk memeriksa bilangan genap atau ganjil
8. Buatlah program yang menerima input 3 buah panjang sisi segitiga kemudian :
  - a. Jika segitiga tersebut sama sisi, katakan SEGITIGA SAMA SISI
  - b. Jika segitiga tersebut sama kaki, katakan SEGITIGA SAMA KAKI
  - c. Jika bukan keduanya, katakan SEGITIGA SEMBARANG
9. Mencari bilangan terbesar dari 3 bilangan
10. Buatlah program diskon :

Input adalah jumlah total pembelian :

Jika total pembelian  $\geq 500000$  maka bonusnya setrika

Jika total pembelian  $\geq 100000$  maka bonusnya payung

Jila total pembelian  $\geq 50000$  maka bonusnya ballpoint

Selain itu tidak mendapat bonus.
11. Buatlah konversi nilai huruf dari nilai yang dimasukkan user!
  - 80 < A  $\leq$  100
  - 65 < B  $\leq$  80
  - 55 < C  $\leq$  65
  - 40 < D  $\leq$  55
  - 0 < E  $\leq$  40
12. Membuat konversi dari bilangan yang dimasukkan sebagai berikut:

Misal 101 : bilangan ratusan

Misal 1200 : bilangan ribuan

Misal 11 : bilangan puluhan

## **Bab 8**

### **Struktur kontrol pengulangan**

#### **Struktur for**

##### **Pokok Bahasan**

1. Struktur for
2. Contoh kasus

##### **Tujuan pembelajaran:**

1. Menenal dan mampu menggunakan struktur pengulangan for
2. Mampu membuat program dengan menggunakan struktur kontrol pengulangan for

Dalam bahasa C tersedia suatu fasilitas yang digunakan untuk melakukan proses yang berulang-ulang sebanyak keinginan kita. Misalnya saja, bila kita ingin menginput dan mencetak bilangan dari 1 sampai 100 bahkan 1000, tentunya kita akan merasa kesulitan. Namun dengan struktur perulangan proses, kita tidak perlu menuliskan perintah sampai 100 atau 1000 kali, cukup dengan beberapa perintah saja. Struktur perulangan dalam bahasa C mempunyai bentuk yang bermacam-macam.

#### **8.1. Struktur For**

Struktur perulangan for biasa digunakan untuk mengulang suatu proses yang telah diketahui jumlah perulangannya. Dari segi penulisannya, struktur perulangan for tampaknya lebih efisien karena susunannya lebih simpel dan sederhana. Bentuk umum perulangan for adalah sebagai berikut :

```
for (inisialisasi; syarat; penambahan)
    pernyataan;
```

##### **Keterangan:**

***Inisialisasi*** : pernyataan untuk menyatakan keadaan awal dari variabel kontrol.

***syarat*** : ekspresi relasi yang menyatakan kondisi untuk keluar dari perulangan.

***penambahan*** : pengatur perubahan nilai variabel kontrol.

Contoh program 1 : Menampilkan deret bilangan 1 2 3 4 5 6 7 8 9 10

##### **Algoritma :**

1. Untuk  $i = 1$  sampai dengan 10 tampilkan  $i$
2. Akhir untuk

## **PEMROGRAMAN KOMPUTER**

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    for(i=1;i<=10;i++)
    {

        printf("%d ",i);

    }

}
```

Contoh program 2: Menampilkan tabel bilangan triangular

Algoritma :

1. Inisialisasikan jumlah = 0
2. Masukkan n
3. Untuk i = 1 sampai dengan n hitung jumlah=jumlah+i
4. Cetak jumlah
5. Akhir untuk

Kode sumber :

```
#include<stdio.h>
main()
{
    int i,n,jumlah;
    jumlah=0;

    printf("Masukkan n = ");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        jumlah=jumlah+i;
        printf(" %d = %d\n",i,jumlah);
    }

}
```



## **PEMROGRAMAN KOMPUTER**

Contoh 3: program untuk menampilkan tabel harga fotokopian dari 1 – 100 lembar, dimana harga per lembar kertas adalah 250 rupiah

Algoritma :

1. Inisialisasikan  $h = 250$
2. Untuk  $lbr = 1$  sampai dengan 20 hitung  $harga = lbr * h$
3. Cetak  $lbr$  dan harga
4. Akhir untuk

Kode sumber :

```
#include<stdio.h>
main()
{
    int lbr,h=250,harga;
    printf("lembar harga\n");
    printf("-----\n");
    for(lbr=1;lbr<=20;lbr++)
    {
        harga=lbr*h;
        printf("%d      %d\n",lbr,harga);
    }
}
```

Contoh 4: buatlah program untuk menampilkan semua bilangan genap dari 1 sampai 20

Algoritma :

1. Untuk  $i = 2$  sampai dengan 20 dengan langkah +2
2. Cetak  $i$
3. Akhit untuk

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    for(i=2;i<=20;i+=2)
    {

        printf("%d ",i);
    }
}
```

}

}

## **8.2. Contoh Kasus**

1. Buatlah program untuk menghitung faktorial dengan menggunakan struktur for. Perlu diketahui suatu faktorial didefinisikan seperti berikut :

$$n! = n \times (n-1) \times (n-2) \times \dots \times 2 \times 1$$

Sebagai contoh :

$$2! = 2 \times 1 = 2$$

$$3! = 3 \times 2 \times 1 = 6$$

2. Daniel menabung sebesar 5 juta dan setiap tahun mendapatkan bunga majemuk sebesar 7%. Buatlah algoritma dan program untuk menampilkan nilai uang Daniel dari akhir tahun pertama hingga tahun kedelapan.
3. Sebuah bilangan prima adalah bilangan bulat yang hanya habis dibagi dengan 1 atau dirinya sendiri. Berdasarkan ketentuan ini, buatlah algoritma dan program untuk menentukan bilangan yang dimasukkan lewat keyboard, termasuk bilangan prima atau bukan.

## **Bab 9**

### **Struktur kontrol pengulangan**

#### **Struktur while**

Pokok Bahasan

1. Struktur while
2. Contoh kasus

Tujuan pembelajaran:

1. Menenal dan mampu menggunakan struktur while
2. Mampu membuat program dengan menggunakan struktur kontrol pengulangan while

#### **9.1. Struktur Perulangan While**

Perulangan WHILE banyak digunakan pada program yang terstruktur. Perulangan ini banyak digunakan bila jumlah perulangannya belum diketahui. Proses perulangan akan terus berlanjut selama kondisinya bernilai benar (true) dan akan berhenti bila kondisinya bernilaisalah.

Bentuk umum dari struktur kondisi ini adalah:

```
While (kondisi)
{
    Pernyataan_1
    Pernyataan_2
}
```

Seandainya dalam tanda { dan } hanya terdapat satu pernyataan, pasangan tanda tersebut bisa dihilangkan.

```
While (kondisi)
    Pernyataan
```

Contoh program 1 : tampilkan deret bilangan 1 2 3 4 5 6 7 8 9 10 dengan menggunakan struktur while

Algoritma :

1. Inisialisasikan  $i = 1$
2. Ulang selama  $i \leq 10$  maka cetak  $i$
3. Hitung  $i = i + 1$

## **PEMROGRAMAN KOMPUTER**

### 4. Akhir ulang

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    i=1;
    while (i<=10)
    {
        printf("%d ",i);
        i=i+1;
    }
}
```

Contoh program 2 : buatlah algoritma dan program untuk menampilkan 6 buah baris yang berisi tulisan “Bahasa C”

Algoritma :

1. Inisialisasikan i = 1
2. Ulang selama i<=6 maka cetak kalimat “Bahasa C”
3. Hitung i++
4. Akhir ulang

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    i=1;
    while (i<=6)
    {
        printf("Bahasa C\n");
        i++;
    }
}
```

Contoh program 3 : Buatlah algoritma dan program untuk menampilkan deret berikut dengan menggunakan struktur pengulangan.

## **PEMROGRAMAN KOMPUTER**

100000000  
10000000  
1000000  
100000  
10000  
1000  
100  
10  
1

Algoritma :

1. Inisialisasikan nilai = 100000000
2. Ulang selama nilai  $\geq 1$  maka cetak nilai
3. Hitung nilai = nilai / 10
4. Akhir ulang

Kode sumber :

```
#include<stdio.h>

main()
{
    long int nilai;
    nilai=100000000;
    while (nilai>=1)
    {
        printf("%ld\n",nilai);
        nilai=nilai/10;
    }
}
```

Contoh program 4 : Buatlah algoritma dan program untuk menampilkan deret bilangan hasil perpangkatan dua dari mulai angka 1 sampai dengan 5. Untuk pangkat gunakan fungsi pow()

Algoritma :

1. Inisialisasikan i = 1
2. Ulang selama i  $\leq 5$  maka hitung pangkat = pow (i,2)
3. Cetak pangkat
4. Hitung i++

## **PEMROGRAMAN KOMPUTER**

### 5. Akhir ulang

Kode sumber :

```
#include<stdio.h>
#include<math.h>
main()
{
    int i,pangkat;
    i=1;
    while (i<=5)
    {
        pangkat=pow(i,2);
        printf("%d ",pangkat);
        i++;
    }
}
```

### **9.2. Contoh Kasus**

1. Buatlah algoritma dan program yang menampilkan deretan seperti berikut :

10 9 8 7 6 5 4 3 2 1

2. Konstanta  $\pi$  dapat diperoleh melalui pendekatan seperti berikut :

$$3. \frac{\pi^2}{6} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \dots + \frac{1}{n^2}$$

Untuk n yang besar (di atas 100)

Buatlah algoritma dan program yang mula-mula meminta data n dimasukkan dari keyboard dan kemudian hitunglah  $\pi$

## **Bab 10**

### **Struktur kontrol pengulangan**

#### **Struktur do...while**

Pokok Bahasan

1. Struktur do...while
2. Contoh kasus

Tujuan pembelajaran:

1. Menenal dan mampu menggunakan struktur do...while
2. Mampu membuat program dengan menggunakan struktur kontrol pengulangan do while

#### **10.1. Struktur PerulanganDo.....While**

Pada dasarnya struktur perulangan do....while sama saja dengan struktur while, hanya saja pada proses perulangan dengan while, seleksi berada di while yang letaknya di atas sementara pada perulangan do....while, seleksi while berada di bawah batas perulangan. Jadi dengan menggunakan struktur do...while sekurang- kurangnya akan terjadi satu kali perulangan.

Bentuk umum dari struktur kondisi ini adalah:

Do

```
{  
    Pernyataan_1  
    Pernyataan_2  
}
```

While (kondisi)

Contoh program 1 : tampilkan deret bilangan 1 2 3 4 5 6 7 8 9 10 dengan menggunakan struktur do...while

Algoritma :

1. Inisialisasikan  $i = 1$
2. Cetak  $i$
3. Hitung  $i++$
4. Ulang selama  $i \leq 10$

## **PEMROGRAMAN KOMPUTER**

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    i=1;
    do
    {
        printf("%d ",i);
        i++;
    }while(i<=10);
}
```

Contoh program 2 :Buatlah algoritma dan program untuk menampilkan bilangan ganjil yang terletak antara 1 sampai dengan 10

Algoritma :

1. Inisialisasikan i = 1
2. Cetak i
3. Hitung i+=2
4. Ulang selama i <=10

Kode sumber :

```
#include<stdio.h>
main()
{
    int i;
    i=1;
    do
    {
        printf("%d ",i);
        i+=2;
    }while(i<=10);
}
```

Contoh program 3 : Buatlah algoritma dan program untuk menampilkan bilangan rata-rata dari n data yang dimasukkan lewat keyboard.



## **PEMROGRAMAN KOMPUTER**

Algoritma :

1. Inisialisasikan jumlah=0
2. Masukkan n
3. Tentukan nilai awal dari i = 1
4. Masukkan data
5. Hitung jumlah += data
6. Hitung i++
7. Ulang mulai langkah ke 4 selama i <=n
8. Hitung rata2=jumlah/n
9. Cetak jumlah dan rata2

Kode sumber :

```
#include<stdio.h>

main()
{
    int i,n;
    float data,rata2,jumlah=0;
    printf("Masukkan jumlah data = ");
    scanf("%d",&n);
    i=1;
    do
    {
        printf("Data ke-%d = ",i);
        scanf("%f",&data);
        jumlah+=data;
        i++;
    }while(i<=n);
    rata2=jumlah/n;
    printf("Jumlah = %g\n",jumlah);
    printf("Nilai rata-rata = %g",rata2);
}
```

Contoh program 4 : Buatlah algoritma dan program untuk menampilkan bilangan fibonacci

Seperti berikut ini : 0 1 1 2 3 5 8

Algoritma :

1. Inisialisasikan f1 = 0

## **PEMROGRAMAN KOMPUTER**

2. Inisialisasikan  $f2 = 1$
3. Cetak  $f1$  dan  $f2$
4. Inisialisasikan  $i = 1$
5. Hitung  $fibo = f1 + f2$
6. Lakukan operasi  $f1 = f2$
7. Lakukan operasi  $f2 = fibo$
8. Cetak  $fibo$
9. Hitung  $i++$
10. Ulang selama  $I \leq 10$

Kode sumber :

```
#include<stdio.h>
main()
{
    int f1=0,f2=1,fibo,i;
    printf ("%d %d ",f1,f2);
    i=1;
    do
    {
        fibo=f1+f2;
        f1=f2;
        f2=fibo;
        printf("%d ",fibo);
        i++;
    }while(i<=10);
}
```

### **10.2. Contoh Kasus**

1. Buatlah program untuk menampilkan perkalian 5. Contoh tampilan keluarannya adalah sebagai berikut :

```
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
4 x 5 = 20
5 x 5 = 25
6 x 5 = 30
7 x 5 = 35
```

## **PEMROGRAMAN KOMPUTER**

$$8 \times 5 = 40$$

$$9 \times 5 = 45$$

$$10 \times 5 = 50$$

2. Buatlah program dengan menggunakan struktur do...while untuk menghitung

$$\frac{1}{2} + \frac{2}{3} + \frac{3}{4} + \dots + \frac{99}{100}$$