MODUL PRAKTIKUM

BAHASA PEMROGRAMAN DASAR (PG168)

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS BUDI LUHUR

2015

DAFTAR ISI

PRAKTIKUM 1
PENGENALAN BAHASA C

1.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa memahami konsep dasar bahasa pemrograman serta mampu membuat dan menjalankan suatu program sederhana di komputer.

Tujuan Khusus

Mahasiswa dapat :

1. Menguraikan konsep dasar bahasa pemrograman dan kaitannya dengan algoritma.
2. Menyebutkan bermacam-macam jenis Bahasa Pemrograman
3. Menjelaskan cara kerja sebuah program dapat berjalan (kode sumber, kompilasi, executable)
4. Membuka dan memahami fitur dasar perangkat lunak editor pembuatan program (IDE).
5. Menulis, meng-kompilasi dan menjalankan (run) program sederhana

1.2 TEORI SINGKAT

Bahasa C dirancang oleh Dennis M. Ritchie, seorang pegawai Bell Telephone Laboratories, Inc. di Murray Hill, New Jersey, Amerika Serikat (sekarang dikenal dengan AT&T Bell Laboratories) pada tahun 1972. Ketika itu ia sedang bertugas membuat sebuah sistem operasi yang terbuka dan interaktif untuk Bell Laboratories. Sistem operasi tersebut dikemudian hari dikenal dengan nama sistem operasi UNIX.

Pada mulanya bahasa pemrograman C digunakan dan dikembangkan hanya terbatas pada lingkungan Bell Laboratories saja, bersama dengan sistem operasi UNIX. Setelah Dennis Ritchie dan Brian Kernighan menerbit buku yang berjudul "The C Programming Language" (Bahasa Pemrograman C) pada tahun 1978, barulah bahasa pemrograman C dikenal dan berkembang luas.

Dalam perkembangannya, muncul banyak varian dari bahasa pemrograman C. Untuk menjaga kompatibilitas dan fleksibilitasnya, lembaga standarisasi ANSI (American National Standards Institute) menetapkan standar unsur-unsur bahasa pemrograman C yang harus terdapat pada suatu varian dari bahasa pemrograman C. Versi standar ini dikenal dengan sebutan ANSI C. Beberapa varian bahasa pemrograman C yang dikenal antara lain Microsoft C, Microsoft Quick C, Borland Turbo C, Borland C, Symantec C, Run/C dan Lattice C.

Bahasa pemrograman C merupakan bahasa pemrograman tingkat tinggi tetapi berorientasi pada sistem operasi komputer yang menggunakan operasi tingkat rendah (bahasa C dikembangkan bersama dengan sistem operasi UNIX, bahkan sistem

operasi UNIX kemudian dibuat dengan menggunakan bahasa pemrograman C) sehingga banyak yang mengelompokkan bahasa pemrograman C sebagai bahasa pemrograman tingkat menengah (bukan tingkat tinggi, bukan pula tingkat rendah).

Salah satu ciri khas dari bahasa pemrograman C adalah programnya terdiri fungsi-fungsi serta seringnya digunakan tipe data pointer.

Berikut ini beberapa keunggulan Bahasa C:

- Bahasa C dapat dijalankan hampir di semua jenis komputer.
- Kode bahasa C sifatnya adalah portable dan fleksibel untuk semua jenis komputer.
- Bahasa hanya menyediakan sedikit kata-kata kunci, hanya terdapat 32 kata kunci.
- Proses executable program bahasa C lebih cepat
- Dukungan pustaka yang banyak.
- Bahasa C adalah bahasa yang terstruktur
- Bahasa C termasuk bahasa tingkat menengah

1.3 PELAKSANAAN PRAKTIKUM

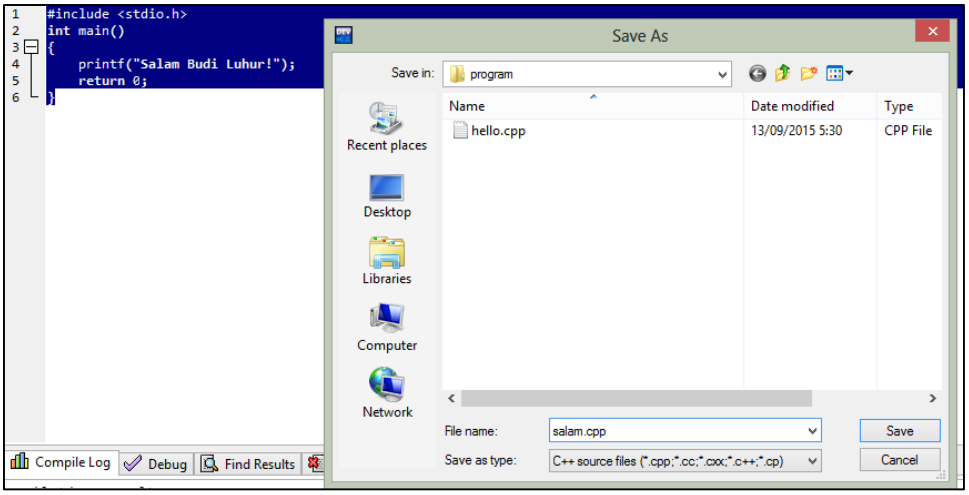
Langkah-langkah Praktikum

1. Buka Editor Bahasa C **Dev-C++ 5.11**.
2. Buatlah file baru dengan membuka menu **File > New > Source File** atau dengan shortcut **Ctrl + N**.
3. Tulislah Program 1.1 berikut ini.

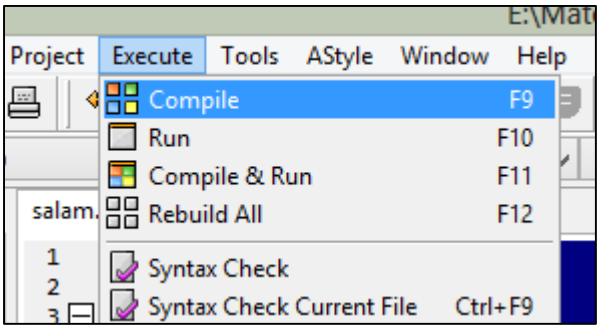
Program 1.1 salam.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      printf("Salam Budi Luhur!");
5      return 0;
6  }
```

4. Simpan program yang telah dituliskan dengan membuka menu **File > Save as...** . Pilih lokasi penyimpanan dan beri nama file dengan "**salam.cpp**".



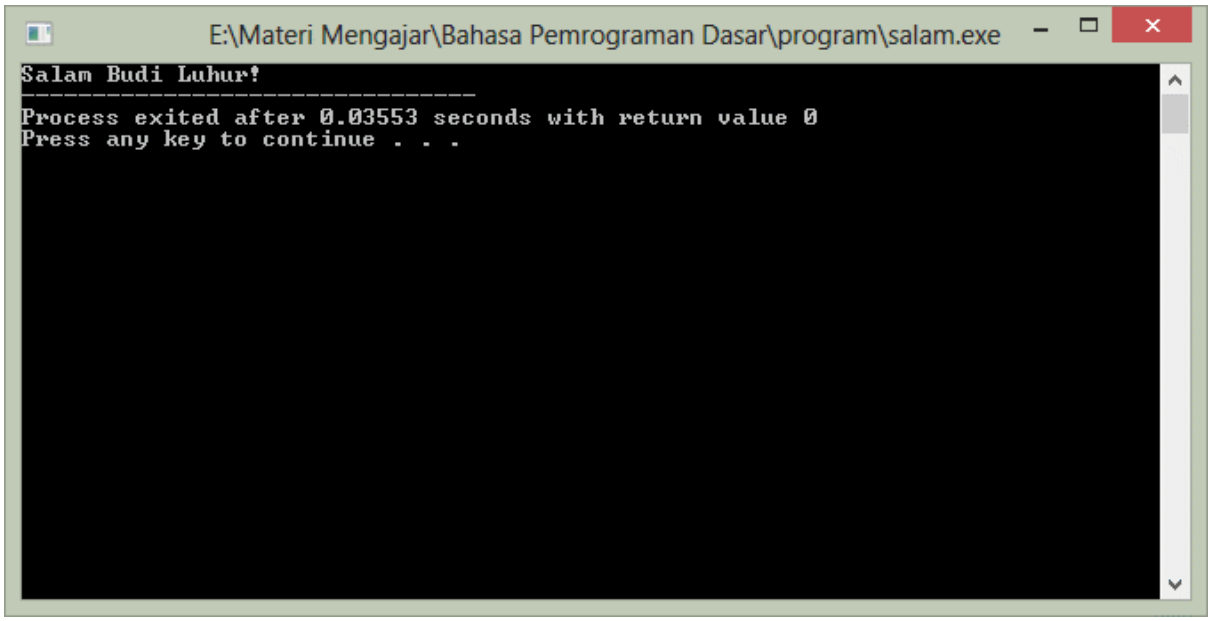
5. Lakukan kompilasi program melalui menu **Execute > Compile** atau dengan menekan shortcut **F9**.



6. Perhatikan hasil kompilasi program pada bagian "Compile Log". Jika kompilasi sukses, maka akan ditampilkan pesan kurang lebih sebagai berikut:

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: E:\Materi Mengajar\Bahasa Pemrograman Dasar\program\salam.exe
- Output Size: 127,931640625 KiB
- Compilation Time: 0,62s
```

7. Jalankan program (running) melalui menu **Execute > Run** atau dengan menekan shortcut **F10**. Hasil program akan ditampilkan pada window seperti pada gambar berikut ini.



8. Proses kompilasi dan running dapat dilaksanakan sekaligus melalu menu **Execute > Compile & Run** atau dengan shortcut **F11**.
9. Selesai.

1.4 LATIHAN

Tuliskan dan jalankan beberapa program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

Program 1.2: total1.cpp

```
1  #include "stdio.h"
2  int main()
3  {
4      printf("%i", (10 + 20));
5      return 0;
6  }
```

Hasil Program 1.2

Program 1.3: total2.cpp

```
1  #include "stdio.h"
2  int main()
3  {
4      int A, B;
5      A = 10;
6      B = 20;
7      printf("%i", (A+B));
8      return 0;
9  }
```

Hasil Program 1.3

Program 1.4: total3.cpp

```
 1  #include "stdio.h"
 2  int main()
 3  {
 4      int A, B, T;
 5      A = 10;
 6      B = 20;
 7      T = A + B;
 8      printf("%i", T);
 9      return 0;
10  }
```

Hasil Program 1.4

1.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Buatlah sebuah program Bahasa C yang menampilkan NIM, NAMA dan JURUSAN Anda di layar!
2. Buatlah sebuah program Bahasa C untuk menghitung luas persegi panjang dengan ukuran Panjang 10 cm dan lebar 7 cm!

PRAKTIKUM 2
STRUKTUR DASAR BAHASA C

2.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa dapat memahami:

1. Struktur penulisan bahasa pemrograman
2. Sintaks assignment statement dan output statement,
3. Keperluan sebuah variable,
4. Tipe data standar bahasa pemrograman.
5. Mengenal berbagai operator dalam Bahasa C

Tujuan Khusus

Mahasiswa dapat :

1. Menuliskan sintaks instruksi : assignment statement, dan output Statement
2. Mendeklarasikan dan menggunakan variabel dalam berbagai tipe data dalam sebuah program
3. Memilih tipe data sesuai dengan kegunaan data tersebut.
4. Menulis program untuk menampilkan isi dari suatu variabel
5. Menulis program untuk menampilkan string yang mengandung karakter khusus
6. Membuat program sederhana yang melibatkan berbagai operator
7. Memberi komentar program

2.2 TEORI SINGKAT

2.2.1 Tipe Data

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh komputer. 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.

Dalam bahasa C terdapat lima tipe data dasar, yaitu :

No	Tipe Data	Ukuran	Range (Jangkauan)	Format	Keterangan
1	char	1 byte	-128 s/d 127	%c	Karakter/string
2	int	2 byte	-32768 s/d 32767	%i , %d	Integer/bilangan bulat

No	Tipe Data	Ukuran	Range (Jangkauan)	Format	Keterangan
3	float	4 byte	-3.4E-38 s/d 3.4E+38	%f	Float/bilangan pecahan
4	double	8 byte	-1.7E-308 s/d 1.7+308	%lf	Pecahan presisi ganda
5	void	0 byte	-	-	Tidak bertipe

2.2.2 Variabel

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Nama dari suatu variabel dapat ditentukan sendiri oleh pemrogram dengan aturan sebagai berikut :

1. Terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Bahasa C bersifat case-sensitive artinya huruf besar dan kecil dianggap berbeda. Jadi antara **nim, NIM** dan **Nim** dianggap berbeda.
2. Tidak boleh mengandung spasi.
3. Tidak boleh mengandung simbol-simbol khusus, kecuali garis bawah (underscore). Yang termasuk symbol khusus yang tidak diperbolehkan antara lain : $, ?, %, #, !, &, *, (,), -, +, = dsb
4. Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

Contoh penamaan variabel yang benar :

NIM, a, x, nama_mhs, f3098, f4, nilai, budi.

Contoh penamaan variable yang salah :

%nilai_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!

2.2.3 Karakter Khusus (*Special Character*)

Pada Bahasa C, pada umumnya karakter atau string dapat ditampilkan dengan menuliskan karakter / string tersebut secara langsung. Namun demikian, terdapat beberapa karakter khusus yang penulisannya sedikit berbeda. Berikut ini karakter khusus yang dikenal di bahasa C beserta penjelasannya.

Karakter Khusus	Penjelasan
\a	Untuk bunyi bell (alert)
\b	Mundur satu spasi (backspace)
\f	Ganti halaman (form feed)
\n	Ganti baris baru (new line)
\r	Menuju ke kolom pertama, baris yang sama (carriage return)
\v	Tabulasi vertikal

Karakter Khusus	Penjelasan
\t	Tabulasi horizontal
\0	Nilai kosong (null)
\'	Kutip tunggal
\"	Kutip ganda
\\	Karakter garis miring (backslash)

2.2.4 Deklarasi

Deklarasi diperlukan bila kita akan menggunakan pengenal (identifier) dalam program. Identifier dapat berupa variable, konstanta dan fungsi.

Deklarasi Variabel

Bentuk umum pendeklarasian suatu variable adalah :

```
Nama_tipe nama_variabel;
```

Contoh :

```
int x;
char y, huruf, nim[10];
char float nilai;
double beta;
int array[5][4];
char *p;
```

Deklarasi Konstanta

Dalam bahasa C konstanta dideklarasikan menggunakan preprocessor #define. Contohnya :

```
#define PHI 3.14
#define nim "0111500382"
#define nama "Achmad Solichin"
```

Deklarasi Fungsi

Fungsi merupakan bagian yang terpisah dari program dan dapat diaktifkan atau dipanggil di manapun di dalam program. Fungsi dalam bahasa C ada yang sudah disediakan sebagai fungsi pustaka seperti printf(), scanf(), getch() dan untuk menggunakannya tidak perlu dideklarasikan. Fungsi yang perlu dideklarasikan terlebih dahulu adalah fungsi yang dibuat oleh programmer. Bentuk umum deklarasi sebuah fungsi adalah : Tipe_fungsi nama_fungsi(parameter_fungsi); Contohnya : float luas_lingkaran(int jari); void tampil(); int tambah(int x, int y);

2.2.5 Operator

Operator Penugasan

Operator Penugasan (*Assignment operator*) dalam bahasa C berupa tanda sama dengan ("="). Contoh :

```
nilai = 80;

A = x * y;
```

Artinya : variable "nilai" diisi dengan 80 dan variable "A" diisi dengan hasil perkalian antara x dan y.

Operator Aritmatika

Bahasa C menyediakan lima operator aritmatika, yaitu :

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

Catatan : operator % digunakan untuk mencari sisa pembagian antara dua bilangan. Misalnya :

```
9 % 2 = 1
9 % 3 = 0
9 % 5 = 4
9 % 6 = 3
```

Operator Hubungan (Perbandingan)

Operator Hubungan digunakan untuk membandingkan hubungan antara dua buah operand (sebuah nilai atau variable. Operator hubungan dalam bahasa C :

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
!=	Tidah sama dengan	x != y	Apakah x tidak sama dengan y

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)

Operator Bitwise

Operator bitwise digunakan untuk memanipulasi bit-bit dari nilai data yang ada di memori. Operator bitwise dalam bahasa C :

- << : Pergeseran bit ke kiri
- >> : Pergeseran bit ke kanan
- & : Bitwise AND
- ^ : Bitwise XOR (exclusive OR)
- | : Bitwise OR
- ~ : Bitwise NOT

Operator Unary

Operator Unary merupakan operator yang hanya membutuhkan satu operand saja. Dalam bahasa C terdapat beberapa operator unary, yaitu :

Operator	**Arti/Maksud**	**Letak**	**Contoh**	**Equivalen**
-	Unary minus	Sebelum operator	A + -B * C	A + (-B) * C
++	Peningkatan dengan penambahan nilai 1	Sebelum dan sesudah	A++	A = A + 1
--	Penurunan dengan pengurangan nilai 1	Sebelum dan sesudah	A--	A = A – 1
sizeof	Ukuran dari operand dalam byte	Sebelum	sizeof(I)	-
!	Unary NOT	Sebelum	!A	-
~	Bitwise NOT	Sebelum	~A	-
&	Menghasilkan alamat memori operand	Sebelum	&A	-
*	Menghasilkan nilai dari pointer	Sebelum	*A	-

Catatan Penting ! :

Operator peningkatan ++ dan penurunan -- jika diletakkan sebelum atau sesudah operand terdapat perbedaan.

2.2.6 Kata Tercadang (Reserved Word)

Bahasa C standar ANSI memiliki 32 kata tercadang (reserved word) dan Turbo C menambahkannya dengan 7 kata tercadang. Semua *reserved word* tidak boleh digunakan dalam penamaan identifier (variable, nama fungsi dll). Kata tercadang yang tersedia dalam bahasa C adalah sbb (tanda * menunjukkan kata tercadang pada Turbo C):

```
*asm, default, for, *pascal, switch, auto, do, goto, register,
typedef, break, double, *huge, return, union, case, else, if,
short, unsigned, *cdecl, enum, int, signed, void, char, extern,
*interrupt, sizeof volatile const *far long static while
continue float *near struct
```

2.2.7 Komentar Program

Komentar program hanya diperlukan untuk memudahkan pembacaan dan pemahaman suatu program (untuk keperluan dokumentasi program). Dengan kata lain, komentar program hanya merupakan keterangan atau penjelasan program. Untuk memberikan komentar atau penjelasan dalam bahasa C digunakan pembatas /* dan */ atau menggunakan tanda // untuk komentar yang hanya terdiri dari satu baris. Komentar program tidak akan ikut diproses dalam kompilasi program (akan diabaikan).

2.3 PELAKSANAAN PRAKTIKUM

1. Tuliskan Program 2.1 berikut ini pada editor Dev-C++.

Program 2.1 variabel1.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10];
5      char nama[30];
6      int nilai;
7
8      printf("NIM : %s", nim);
9      printf("NAMA : %s", nama);
10     printf("NILAI : %i", nilai);
11
12     return 0;
13 }
```

2. Jalankan program 2.1 di atas dan tuliskan apa yang tercetak di layar.

3. Pada Program 2.1, variabel yang dideklarasikan belum diisi dengan nilai tertentu (belum diinisialisasi). Lakukan perubahan Program 2.1 dengan menambahkan perintah untuk mengisi variabel nim, nama dan nilai seperti pada Program 2.2 berikut ini.

Program 2.2 variabel2.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10];
5      char nama[30];
6      int nilai;
7
8      nim = "141150123";
9      nama = "Achmad Solichin";
10     nilai = 85;
11
12     printf("NIM : %s", nim);
13     printf("NAMA : %s", nama);
14     printf("NILAI : %i", nilai);
15
16     return 0;
17 }
```

4. Lakukan kompilasi Program 2.2 di atas dan perhatikan hasilnya. Apakah terjadi error? Apa error yang akan ditampilkan? Tuliskan error yang ditampilkan dan baris berapa terjadi error!

Error yang ditampilkan: Error terjadi pada baris ke:

5. Kesalahan pada Program 2.2 terjadi karena cara pengisian variabel nim dan nama yang kurang tepat. Variabel nim dan nama dideklarasikan sebagai sebuah variabel bertipe char dan berupa array (ditunjukkan dengan adanya tanda kurung siku []). Pembahasan mengenai array akan dilakukan secara khusus pada Pertemuan ke-9.
6. Sekarang kita perbaiki kesalahan pada Program 2.2 dengan mengisi nim dan nama menggunakan fungsi strcpy(). Penggunaan fungsi strcpy() harus menyertakan header fungsi <string.h>. Ubahlah Program 2.2 menjadi Program 2.3 berikut ini.

Program 2.3 variabel3.cpp

```
 1  #include <stdio.h>
 2  #include <string.h>
 3  int main()
 4  {
 5      char nim[10];
 6      char nama[30];
 7      int nilai;
 8
 9      strcpy(nim, "141150123");
10      strcpy(nama, "Achmad Solichin");
11      nilai = 85;
12
13      printf("NIM : %s", nim);
14      printf("NAMA : %s", nama);
15      printf("NILAI : %i", nilai);
16
17      return 0;
18  }
```

7. Jalankan Program 2.3 dan tuliskan apa yang tercetak di layar!

8. Tampilan Program 2.3 sedikit berantakan bukan? Mari kita buat lebih rapi dengan menambahkan karakter khusus \n (pindah baris) dan \t (tabulasi horizontal). Perhatikan Program 2.4 berikut ini dan jalankan!

Program 2.4 variabel4.cpp

```
1  #include <stdio.h>
2  #include <string.h>
3  int main()
4  {
5      char nim[10];
6      char nama[30];
7      int nilai;
8
9      strcpy(nim, "141150123");
10     strcpy(nama, "Achmad Solichin");
11     nilai = 85;
12
13     printf("NIM \t: %s", nim);
14     printf("\nNAMA \t: %s", nama);
15     printf("\nNILAI \t: %i", nilai);
16
17     return 0;
18 }
```

9. Jalankan Program 2.4 dan tuliskan apa yang tercetak di layar!

10. Selanjutnya kita akan ubah Program 2.4 menjadi Program 2.5 dengan menambahkan nilai uts, uas, tugas dan kehadiran serta menghitung nilai akhir yang diperoleh mahasiswa. Nilai akhir diperoleh dengan menggunakan rumus sebagai berikut:

    NILAI AKHIR = 10% kehadiran + 20% tugas + 30% uts + 40% uas

    Kita akan mendeklarasikan variabel kehadiran, tugas, uts dan uas bertipe integer dan variabel nilai_akhir bertipe float (pecahan).

Program 2.5 variabel5.cpp

```
1  #include <stdio.h>
2  #include <string.h>
3  int main()
4  {
5      char nim[10];
6      char nama[30];
7      int kehadiran, tugas, uts, uas;
8      float nilai_akhir;
9
10     strcpy(nim, "141150123");
11     strcpy(nama, "Achmad Solichin");
12     kehadiran  = 100;
13     tugas      = 90;
14     uts        = 83;
15     uas    = 86;
16
17     nilai_akhir = (0.1 * kehadiran) + (0.2 * tugas) + (0.3
   * uts) + (0.4 * uas);
18
19     printf("NIM \t: %s", nim);
20     printf("\nNAMA \t: %s", nama);
21     printf("\nKEHADIRAN \t: %i", kehadiran);
22     printf("\nTUGAS \t: %i", tugas);
23     printf("\nUTS \t: %i", uts);
24     printf("\nUAS \t: %i", uas);
25     printf("\nNILAI AKHIR \t: %.2f", nilai_akhir);
26
27     return 0;
28 }
```

11. Jalankan Program 2.5 dan tuliskan apa yang tercetak!

12. Untuk lebih mempermudah pembacaan program (dan untuk keperluan belajar), kita dapat menambahkan komentar-komentar program. Ubah Program 2.5 menjadi Program 2.6 berikut ini.

Program 2.6 variabel6.cpp

```
1  #include <stdio.h>
2  #include <string.h>
3  int main()
4  {
5      char nim[10];
6      char nama[30];
7      int kehadiran, tugas, uts, uas;
8      float nilai_akhir;
9
10     strcpy(nim, "141150123");
11     strcpy(nama, "Achmad Solichin");
12     kehadiran  = 100;     //nilai kehadiran
13     tugas      = 90;      //nilai tugas
14     uts        = 83;      //nilai uts
15     uas    = 86;      //nilai uas
16
17     /* perhitungan nilai akhir
18        sesuai peraturan di UBL
19     */
20     nilai_akhir = (0.1 * kehadiran) + (0.2 * tugas) + (0.3
   * uts) + (0.4 * uas);
21
22     //tampilkan data
23     printf("NIM \t: %s", nim);
24     printf("\nNAMA \t: %s", nama);
25     printf("\nKEHADIRAN \t: %i", kehadiran);
26     printf("\nTUGAS \t: %i", tugas);
27     printf("\nUTS \t: %i", uts);
28     printf("\nUAS \t: %i", uas);
29     printf("\nNILAI AKHIR \t: %.2f", nilai_akhir);
30
31     return 0;
32 }
```

13. Jalankan Program 2.6 dan lihat hasilnya. Bandingkan dengan hasil Program 2.5. Sama bukan? Hal tersebut menunjukkan bahwa komentar program tidak mempengaruhi hasil program. Komentar program diperlukan oleh programmer sendiri agar dalam mempermudah pembacaan program.

2.4 LATIHAN

Tuliskan dan jalankan beberapa program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

Program 2.7 unary.cpp

```
 1  #include <stdio.h>
 2  int main()
 3  {
 4      int A, B;
 5      A = 5;
 6      printf("A = %i", A);
 7      printf("\nA = %i", A++);
 8      printf("\nA = %i", A);
 9
10      B = 10;
11      printf("\n\nB = %i", B);
12      printf("\nB = %i", ++B);
13      printf("\nB = %i", B);
14
15      return 0;
16  }
```

Hasil Program 2.7

Program 2.8 lingkaran.cpp

```
 1  #include <stdio.h>
 2  int main()
 3  {
 4      int jari;
 5      float luas, keliling;
 6      jari = 7;
 7
 8      luas = 3.14 * jari * jari;
 9      keliling = 2 * 3.14 * jari;
10
11      printf("LUAS dan KELILING LINGKARAN");
12      printf("\nJari-jari = %i", jari);
13      printf("\nLUAS = %.3f", luas);
14      printf("\nKELILING = %.2f", keliling);
15
16      return 0;
17  }
```

Hasil Program 2.8

2.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Diketahui sebuah segitiga memiliki panjang alas = 8 cm dan tinggi = 5 cm. Buatlah sebuah program dalam Bahasa C untuk menghitung dan menampilkan luas dari segitiga tersebut!
2. Diketahui sebuah bangun ruang berbentuk bola dan memiliki diameter 15 cm. Buatlah sebuah program dalam Bahasa C untuk menghitung dan menampilkan isi (volume) dari bangun ruang bola tersebut!
3. Diketahui sebuah segitiga siku-siku dengan panjang sisi alas 4 cm dan sisi tinggi 5 cm. Dengan menggunakan rumus Phitagoras, buatlah sebuah program Bahasa C untuk menghitung sisi miring segitiga tersebut dan menampilkannya di layar!

PRAKTIKUM 3
MASUKAN DAN KELUARAN PROGRAM

3.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa dapat memahami:

1. Penggunaan perintah Input (masukan) pada sebuah program.
2. Penggunaan perintah Output (keluaran) untuk menampilkan berbagai data pada sebuah program.
3. Penggunaan format cetakan berbagai variabel.

Tujuan Khusus

Mahasiswa dapat :

1. Menuliskan perintah intruksi input pada sebuah program.
2. Mengetahui dan menggunakan berbagai format inputan
3. Menyusun berbagai program yang menginput dan menampilkan berbagai data tipe data.
4. Membaca dan menjelaskan maksud dari suatu program.

3.2 TEORI SINGKAT

3.2.1 Perintah Masukan

Dalam bahasa C proses memasukkan suatu data bisa menggunakan beberapa fungsi pustaka yang telah tersedia. Beberapa fungsi pustaka yang bisa digunakan adalah :

Fungsi scanf()

Fungsi pustaka scanf() digunakan untuk menginput data berupa data numerik, karakter dan string secara terformat.

Hal-hal yang perlu diperhatikan dalam pemakaian fungsi scanf() :

- Fungsi scanf() memakai penentu format
- Fungsi scanf() memberi pergantian baris secara otomatis
- Fungsi scanf() tidak memerlukan penentu lebar field
- Variabelnya harus menggunakan operator alamat &

NO	FORMAT	KETERANGAN	CONTOH
1	%c	Membaca sebuah karakter	scanf("%c", &N);
2	%s	Membaca sebuah string	scanf("%s", &N);
3	%i, %d	Membaca sebuah bilangan bulat (integer)	scanf("%i", &N);

NO	FORMAT	KETERANGAN	CONTOH
4	%f, %e	Membaca sebuah bilangan pecahan (real)	scanf("%f", &N);
5	%o	Membaca sebuah bilangan basis octal	scanf("%o", &N);
6	%x	Membaca sebuah bilangan basis heksadesimal	scanf("%x", &N);
7	%u	Membaca sebuah bilangan tak bertanda	scanf("%u", &N);

Fungsi gets()

- Fungsi gets() digunakan untuk memasukkan data bertipe karakter dan tidak dapat digunakan untuk memasukkan data numerik.
- Harus diakhiri dengan penekanan tombol enter
- Cursor secara otomatis akan pindah baris
- Tidak memerlukan penentu format

Fungsi getchar()

- Fungsi getchar() digunakan untuk membaca data yang bertipe karakter
- Harus diakhiri dengan penekanan tombol enter
- Karakter yang dimasukkan terlihat pada layar
- Pergantian baris secara otomatis

Fungsi getch() dan getche()

- Fungsi getch() dan getche() digunakan untuk membaca data karakter.
- Karakter yang dimasukkan tidak perlu diakhiri dengan penekanan tombol enter.
- Tidak memberikan efek pergantian baris secara otomatis
- Jika menggunakan fungsi getch() karakter yang dimasukkan tidak akan ditampilkan pada layar sehingga sering digunakan untuk meminta inputan berupa password.
- Sedangkan pada getche() karakter yang dimasukkan akan ditampilkan pada layar.

3.2.2 Perintah Keluaran

Menampilkan data ke layar monitor

- Menggunakan fungsi **printf()**, **puts()**, dan **putchar()**.
- Fungsi printf() digunakan untuk menampilkan semua jenis data (numerik dan karakter)
- Fungsi puts() digunakan untuk menampilkan data string dan secara otomatis akan diakhiri dengan perpindahan baris.
- Fungsi putchar() digunakan untuk menampilkan sebuah karakter.

Mengatur tampilan bilangan pecahan (float).

Bentuk umum :

```
printf("%m.nf", var);
```

dimana:

- m : menyatakan panjang range
- n : menyatakan jumlah digit di belakang koma.
- var : nilai atau variable yang akan ditampilkan.

Contoh :

```
printf("%5.2f", nilai);
```

artinya variable nilai akan ditampilkan sebanyak 5 digit dengan 2 digit di belakang koma.

Menampilkan data ke printer

- Untuk menampilkan data ke printer dapat menggunakan fungsi fprintf(), fputs() dan fputc().
- Fungsi fprintf() digunakan untuk mencetak semua jenis tipe data ke printer dan secara otomatis memberikan efek perpindahan baris.
- Fungsi fputs() digunakan untuk mencetak tipe data string ke printer
- Fungsi fputc() digunakan untuk mencetak tipe data karakter ke printer

3.3 PELAKSANAAN PRAKTIKUM

1. Tuliskan Program 3.1 berikut ini pada editor Dev-C++.

Program 3.1 inout1.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10], nama[30];
5      printf("INPUT DATA MAHASISWA\n");
6      printf("NIM : "); scanf("%s", &nim);
7      printf("NAMA : "); scanf("%s", &nama);
8
9      //tampilkan
10     printf("\nNIM : %s", nim);
11     printf("\nNAMA : %s", nama);
12     return 0;
13 }
```

2. Jalankan Program 3.1 di atas dan inputkan NIM dan NAMA Anda masing-masing. Tuliskan apa yang tercetak di layar! Apakah ada hasil yang "tidak sesuai" ?

3. Sekarang ubah Program 3.1 menjadi Program 3.2 berikut ini. Fungsi scanf() untuk menginput NAMA diganti dengan fungsi gets().

Program 3.2 inout2.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10], nama[30];
5      printf("INPUT DATA MAHASISWA\n");
6      printf("NIM : "); scanf("%s", &nim);
7      printf("NAMA : "); gets(nama);
8
9      //tampilkan
10     printf("\nNIM : %s", nim);
11     printf("\nNAMA : %s", nama);
12     return 0;
13 }
```

4. Lakukan kompilasi dan jalankan Program 3.2 lalu inputkan kembali NIM dan NAMA Anda. Apa yang terjadi? Apakah berhasil?
5. Sekarang tambahkan perintah `fflush(stdin);` setelah perintah untuk menginput NIM. Perhatikan Program 3.3 berikut ini.

Program 3.3 inout3.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10], nama[30];
5      printf("INPUT DATA MAHASISWA\n");
6      printf("NIM : "); scanf("%s", &nim);
7      fflush(stdin);
8      printf("NAMA : "); gets(nama);
9
10     //tampilkan
11     printf("\nNIM : %s", nim);
12     printf("\nNAMA : %s", nama);
13     return 0;
14 }
```

6. Jalankan Program 3.3 di atas dan inputkan NIM dan NAMA Anda masing-masing. Tuliskan apa yang tercetak di layar! Perintah `fflush(stdin)` berfungsi menghapus buffer I/O di dalam memori. Fungsi dapat ditambahkan setelah perintah inputan.

7. Selanjutnya ditambahkan variabel "nilai" pada Program 3.3 untuk menginput dan menyimpan nilai mahasiswa. Variabel "nilai" dideklarasikan sebagai variabel yang bertipe float (pecahan). Perhatikan Program 3.4 di bawah.

Program 3.4 inout4.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      char nim[10], nama[30];
5      float nilai;
6      printf("INPUT DATA MAHASISWA\n");
7      printf("NIM : "); scanf("%s", &nim);
8      fflush(stdin);
9      printf("NAMA : "); gets(nama);
10     printf("NILAI : "); scanf("%f", &nilai);
11
```

```
12      //tampilkan
13      printf("\nNIM : %s", nim);
14      printf("\nNAMA : %s", nama);
15      printf("\nNILAI : %f", nilai);
16      printf("\nNILAI (PEMBULATAN) : %.2f", nilai);
17
18      return 0;
19  }
```

8. Jalankan Program 3.4 dan pada inputan nilai, masukkan beberapa nilai berikut ini: 80, 78.253, 87.243, 90.55. Tuliskan hasil tampilan program!

3.4 LATIHAN

Tuliskan dan jalankan beberapa program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

Program 3.5 lingkaran_in.cpp

```
1   #include <stdio.h>
2   int main()
3   {
4       int jari;
5       float luas, keliling;
6
7       printf("Input jari-jari lingkaran : ");
8       scanf("%i", &jari);
9
10      luas = 3.14 * jari * jari;
11      keliling = 2 * 3.14 * jari;
12
13      printf("LUAS dan KELILING LINGKARAN");
14      printf("\nJari-jari = %i", jari);
15      printf("\nLUAS = %.3f", luas);
16      printf("\nKELILING = %.2f", keliling);
17
18      return 0;
19  }
```

Hasil Program 3.5

Program 3.6 volumebola.cpp

```
 1  #include <stdio.h>
 2  #define PHI 3.14
 3  int main()
 4  {
 5      float jari;
 6      float volume;
 7
 8      printf("PROGRAM MENGHITUNG VOLUME BOLA\n\n");
 9      printf("Input jari-jari bola (cm) : ");
10      scanf("%f", &jari);
11
12      //hitung volume
13      volume = 4/3 * PHI * jari * jari * jari;
14
15      printf("\nVolume bola dg jari-jari %.2f cm adalah %.3f
    cm3.", jari, volume);
16      return 0;
17  }
```

Hasil Program 3.6

3.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Buatlah sebuah program Bahasa C untuk menginput panjang alas dan tinggi segitiga (dalam cm). Lalu hitung dan tampilkan luas dan keliling dari segitiga tersebut!
2. Buatlah sebuah program Bahasa C untuk menginput sebuah nilai bilangan bulat yang menyatakan suhu dalam satuan Celcius. Selanjutnya hitung dan tampilkan nilai sudut dalam derajat Fahrenheit dan Reamur!
3. Buatlah sebuah program Bahasa C untuk menginput sisi alas dan sisi tinggi sebuah segitiga siku-siku (dalam cm). Dengan menggunakan rumus Phitagoras, buatlah sebuah program Bahasa C untuk menghitung sisi miring segitiga tersebut dan menampilkannya di layar!
4. Buatlah sebuah program Bahasa C untuk menginput sebuah nilai bilangan bulat positif, lalu tampilkan keterangan "GANJIL" jika bilangan tersebut adalah ganjil dan "GENAP" jika bilangan tersebut adalah genap.

PRAKTIKUM 4
STRUKTUR KONDISI IF DAN IF…ELSE

4.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa dapat memahami:

1. Bentuk umum struktur kondisi IF dan IF…ELSE
2. Penggunaan struktur kondisi IF dan IF…ELSE pada sebuah program.

Tujuan Khusus

Mahasiswa dapat :

1. Menentukan nilai TRUE atau FALSE suatu kondisi yang ditulis dalam bemacam-macam variasi penulisan kondisi pada statement if.
2. Mengenal dan menggunakan bermacam-macam operator relational dalam program.
3. Dapat memilih statement if yang menggunakan else (if-then-else) atau statement if yang tidak menggunakan else (if - then) pada sebuah program
4. Menulis program untuk alur yang dinyatakan dalam bentuk Flowchart atau pseudocode.

4.2 TEORI SINGKAT

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 lebih kompleks.

4.2.1 Struktur Kondisi IF

Struktur kondisi IF dibentuk dari pernyataan IF dan 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) {
    perintah-jika-benar;
}
```

Catatan:

- Kondisi dalam struktur IF dapat berupa kondisi tunggal maupun kondisi gabungan (jamak) yang bernilai TRUE / FALSE.
- Penggunaan tanda kurawal { dan } bersifat WAJIB jika blok perintah terdiri dari dua perintah atau lebih. Tanda kurawal dapat dihilangkan jika blok perintah hanya terdiri dari satu perintah.

4.2.2 Struktur Kondisi IF…ELSE

Dalam struktur kondisi IF…ELSE minimal terdapat dua blok perintah. Jika kondisi yang diperiksa bernilai benar atau terpenuhi maka blok perintah pertama yang dilaksanakan dan jika kondisi yang diperiksa bernilai salah maka blok perintah yang kedua yang dilaksanakan.

Bentuk umumnya adalah sebagai berikut :

```
if (kondisi) {
    perintah-jika-benar;
} else {
    perintah-jika-salah;
}
```

Catatan:

- Kondisi dalam struktur IF…ELSE dapat berupa kondisi tunggal maupun kondisi gabungan (jamak) yang bernilai TRUE / FALSE.
- Penggunaan tanda kurawal { dan } bersifat WAJIB jika blok perintah terdiri dari dua perintah atau lebih. Tanda kurawal dapat dihilangkan jika blok perintah hanya terdiri dari satu perintah.

4.3 PELAKSANAAN PRAKTIKUM

1. Tuliskan Program 4.1 berikut ini pada editor Dev-C++.

Program 4.1 if_nilai.cpp

```
 1  #include <stdio.h>
 2  int main()
 3  {
 4      int nilai;
 5      printf("Input nilai matakuliah: ");
 6      scanf("%i", &nilai);
 7
 8      if (nilai >= 60) {
 9          printf("\nLULUS");
10      }
11      printf("\n\nProgram Selesai.");
12      return 0;
13  }
```

2. Jalankan Program 4.1 di atas dan inputkan beberapa **nilai** berikut ini!. Catat hasilnya!

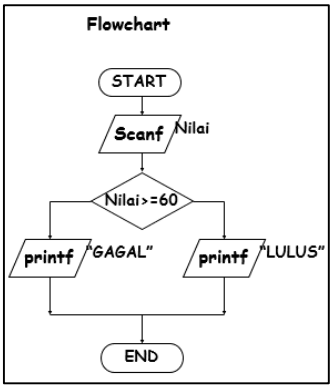
NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

3. Ubah program sebelumnya sehingga menjadi Program 4.2 berikut ini.

Program 4.2 ifelse_nilai.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4      int nilai;
5      printf("Input nilai matakuliah: ");
6      scanf("%i", &nilai);
7
8      if (nilai >= 60) {
9          printf("\nLULUS");
10     } else {
11         printf("\nGAGAL");
12     }
13     printf("\n\nProgram Selesai.");
14
15     return 0;
16 }
```



4. Jalankan Program 4.2 di atas dan inputkan beberapa **nilai** berikut ini!. Catat hasilnya!

NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

5. Hapus tanda kurawal { dan } pada baris 8, 10 dan 12. Jalankan program. Inputkan kembali beberapa nilai berikut ini dan catat hasil / keluarannya!

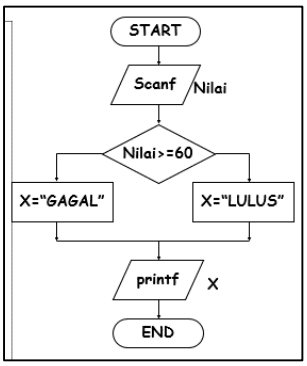
NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

6. Program 4.3 berikut ini merupakan program berbeda dari program sebelumnya namun memiliki hasil yang sama. Tuliskan, kompilasi dan jalankan!

Program 4.3 ifelse_nilai_cara2.cpp

```
1  #include <stdio.h>
2  #include <string.h>
3  main()
4  {
5      int nilai;
6      char X[10];
7      printf("\nInputkan sebuah nilai : ");
8      scanf("%i", &nilai);
9      if (nilai >=60)
10         strcpy(X, "LULUS");
11     else
12         strcpy(X, "GAGAL");
13     printf("\n\n %s", X);
14
15     printf("\nProgram Selesai.");
16     return 0;
17 }
```



7. Jalankan Program 4.3 di atas dan inputkan beberapa **nilai** berikut ini!. Catat hasilnya!

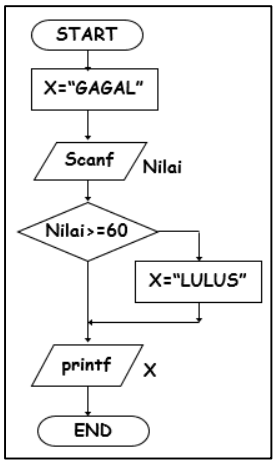
NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

8. Ubahlah Program 4.3 di atas menjadi Program 4.4 berikut ini. Kompilasi dan jalankan program 4.4 dan inputkan beberapa nilai. Apakah hasilnya sama?

Program 4.4 ifelse_nilai_cara3.cpp

```
1  #include <stdio.h>
2  #include <string.h>
3  main()
4  {
5       int nilai;
6       char X[10];
7       strcpy(X, "GAGAL");
8       printf("\n Inputkan sebuah nilai : ");
9       scanf("%i", &nilai);
10      if (nilai >=60)
11           strcpy(X, "LULUS");
12      printf("\n %s", X);
13      printf("\nProgram Selesai.");
14      return 0;
15 }
```



9. Tuliskan Program 4.5 berikut ini.

Program 4.5 ifelse_grade.cpp

```
1  #include <stdio.h>
2  int main()
3  {
4       int nilai;
5       printf("Input nilai matakuliah: ");
6       scanf("%i", &nilai);
7
8       if (nilai >= 85 && nilai <= 100) {
9            printf("\nLULUS");
10           printf("\nGRADE A");
11      }
12      if (nilai >= 75 && nilai < 85) {
13           printf("\nLULUS");
14           printf("\nGRADE B");
15      }
16      if (nilai >= 60 && nilai < 75) {
17           printf("\nLULUS");
18           printf("\nGRADE C");
19      }
20      if (nilai >= 45 && nilai < 60) {
21           printf("\nGAGAL");
22           printf("\nGRADE D");
23      }
24      if (nilai >= 0 && nilai < 45) {
25           printf("\nGAGAL");
26           printf("\nGRADE E");
27      }
28      if (nilai < 0 || nilai > 100) {
29           printf("\nInput nilai antara 0-100");
30      }
31
32      return 0;
33 }
```

10. Jalankan Program 4.5 di atas dan inputkan beberapa **nilai** berikut ini!. Catat hasilnya!

NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

11. Hapus tanda kurawal { dan } pada seluruh blok IF. Kompilasi dan jalankan Program 4.5. Inputkan kembali beberapa nilai berikut ini dan catat hasil / keluarannya!

NILAI	KELUARAN
80	
100	
40	
50	

NILAI	KELUARAN
60	
61	
150	
-80	

12. Catatlah beberapa kesimpulan yang Anda peroleh dari beberapa langkah praktikum di atas!

4.4 LATIHAN

1. Tuliskan dan jalankan program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

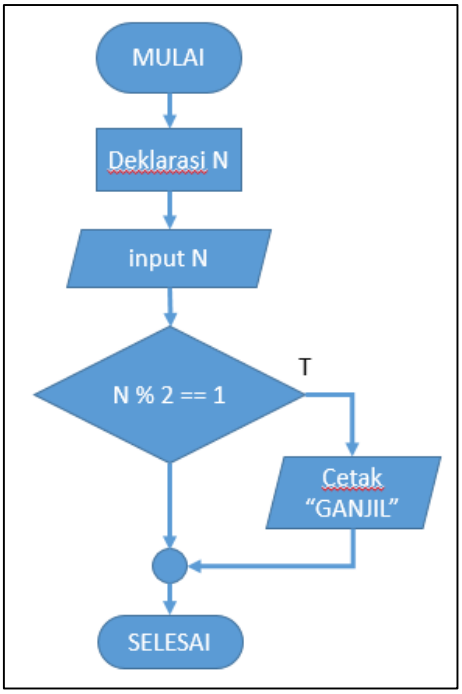
Program 4.6 terbesar_cara2.cpp

```
#include <stdio.h>
int main()
{
    int A, B, max;
    printf("PROGRAM NILAI TERBESAR 2 BILANGAN\n\n");
    printf("Input Bilangan 1: ");
    scanf("%i", &A);
    printf("Input Bilangan 2: ");
    scanf("%i", &B);

    if (A > B) {
        max = A;
    } else {
        max = B;
    }
    printf("\nBilangan terbesar = %i", max);

    return 0;
}
```

Hasil Program 4.6

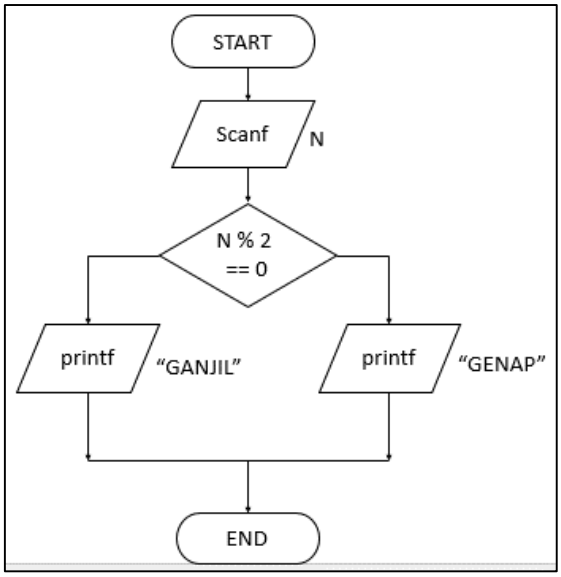
2. Buatlah program bahasa C berdasarkan beberapa flowchart / algoritma berikut ini!

Flowchart 1



Tuliskan Program Flowchart 1

Flowchart 2

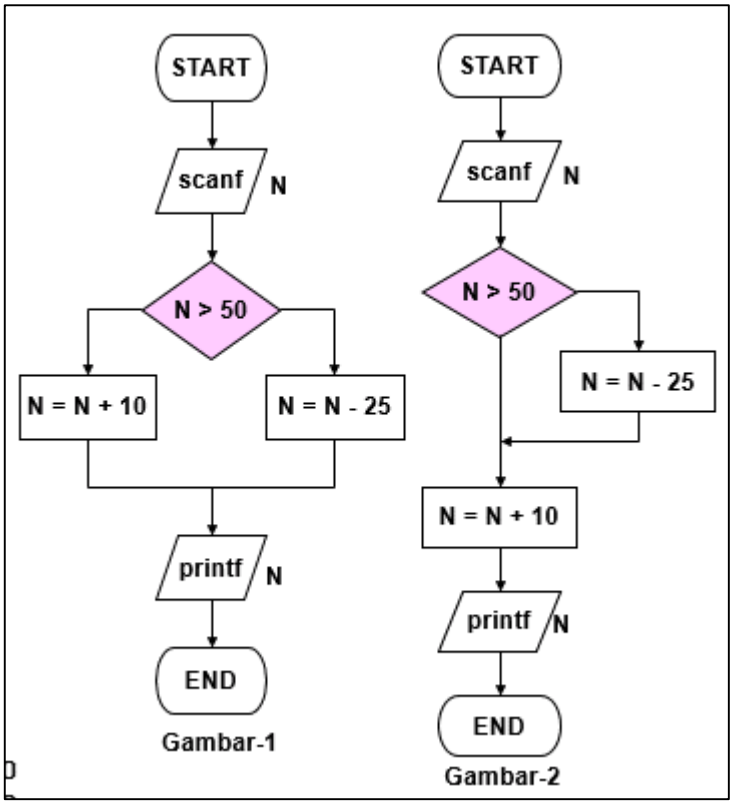


Tuliskan Program Flowchart 2

4.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Buatlah sebuah program Bahasa C untuk menginput sebuah nilai bilangan bulat positif, lalu tampilkan keterangan "GANJIL" jika bilangan tersebut adalah ganjil dan "GENAP" jika bilangan tersebut adalah genap.



2. Perhatikan flowchart pada **Gambar 1** di atas. Buatlah sebuah program Bahasa C berdasarkan flowchart tersebut! Selanjutnya jalankan dan input beberapa bilangan berikut ini!
    a. 30
    b. 50
    c. 65
3. Perhatikan flowchart pada **Gambar 2** di atas. Buatlah sebuah program Bahasa C berdasarkan flowchart tersebut! Selanjutnya jalankan dan input beberapa bilangan berikut ini!
    a. 30
    b. 50
    c. 65

PRAKTIKUM 5
STRUKTUR KONDISI IF BERTINGKAT DAN SWITCH…CASE

5.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa dapat memahami:

1. Bentuk umum struktur kondisi IF bertingkat dan SWITCH…CASE.
2. Penggunaan struktur kondisi IF bertingkat dan SWITCH…CASE pada sebuah program.

Tujuan Khusus

Mahasiswa dapat :

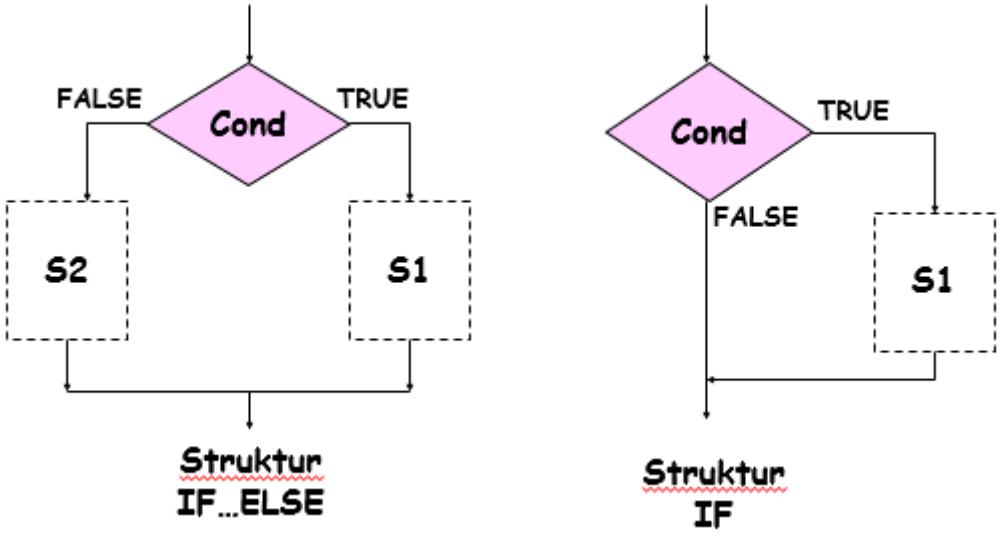
1. Menuliskan program yang menggunakan IF bertingkat.
2. Menuliskan program yang menggunakan struktur SWITCH…CASE.
3. Menulis program untuk alur yang dinyatakan dalam bentuk Flowchart atau pseudocode.

5.2 TEORI SINGKAT

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 lebih kompleks.

5.2.1 Struktur Kondisi IF Bertingkat

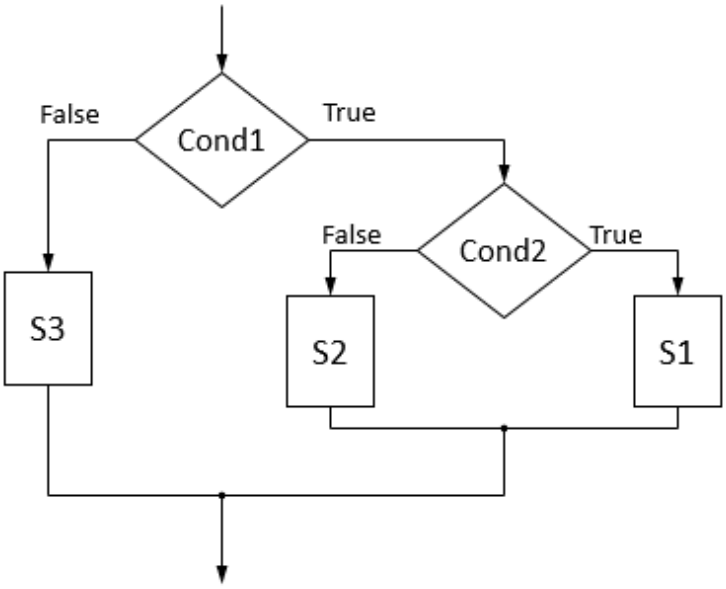
IF Bertingkat sering disebut juga dengan IF Bersarang atau Nested IF. Pada dasarnya IF Bertingkat merupakan struktur IF atau IF…ELSE. Perhatikan kembali flowchart struktur IF dan IF…ELSE seperti digambarkan sebagai berikut:



Dari ilustrasi struktur diatas, S1 dan S2 adalah satu atau sekelompok statement. Di dalam kelompok S1 dan S2 mungkin terdapat statement IF sehingga terjadi IF secara berjenjang atau secara tersarang yang biasa disebut Nested IF (nest = sarang).

Contoh Struktur IF Bertingkat / IF Bersarang:

```
if (cond1)
  {if (cond2)
      {-
       -  S1
       -
      }
    else {-
          - S2
          -
         }
   }
 else {-
       - S3
       -
      }
```



5.2.2 Struktur SWITCH...CASE

Struktur kondisi SWITCH...CASE digunakan untuk penyeleksian kondisi dengan kemungkinan yang terjadi cukup banyak. Struktur ini akan memeriksa isi dari 'variabel' yang berada di dalam SWITCH dan melaksanakan salah satu dari beberapa pernyataan 'CASE'. 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 struktur SWITCH...CASE adalah sebagai berikut :

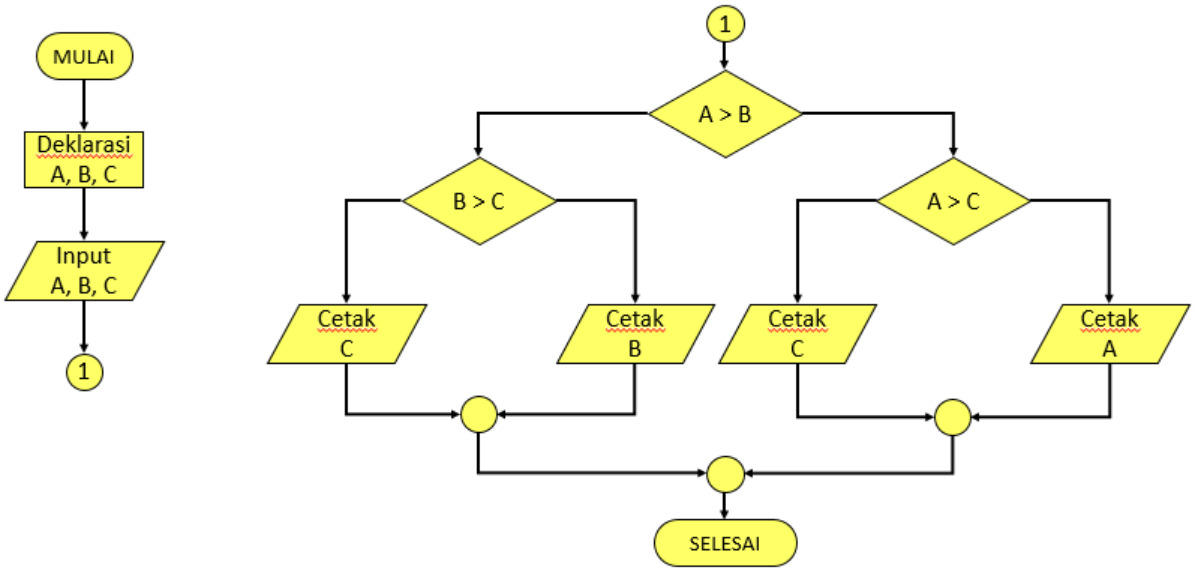
```
switch(variabel)
{
    case 1 : pernyataan-1; break;
    case 2 : pernyataan-2; break;
    .....
    .....
    case n : pernyataan-n; break;
    default : pernyataan-m;
}
```

Catatan:

- Variabel 'variabel' yang berada di dalam SWITCH harus bertipe **INT** atau **CHAR**.
- Nilai pada CASE harus menyesuaikan tipe data dari 'variabel' dan harus bernilai tunggal.
- Compiler akan memeriksa kebenaran kondisi dari mulai case ke-1 hingga ke-n.

5.3 PELAKSANAAN PRAKTIKUM

1. Perhatikan flowchart berikut ini dan tuliskan Program 5.1 berikut ini pada editor Dev-C++.



Program 5.5.1 terbesar3bil.cpp

```
 1  #include <stdio.h>
 2  main()
 3  {
 4      int A, B, C;
 5      printf("Input 3 buah bilangan\n");
 6      printf("Bilangan 1: "); scanf("%i", &A);
 7      printf("Bilangan 2: "); scanf("%i", &B);
 8      printf("Bilangan 3: "); scanf("%i", &C);
 9
10      printf("\nBilangan terbesar: ");
11      if (A > B) {
12          if (A > C) {
13            printf("%i", A);
14          } else {
15            printf("%i", C);
16          }
17      } else {
18          if (B > C) {
19            printf("%i", B);
20          } else {
21            printf("%i", C);
22          }
23      }
24      return 0;
25  }
```

2. Jalankan Program 5.1 di atas dan inputkan beberapa kombinasi **nilai A, B dan C** berikut ini!. Catat hasilnya!

A	B	C	KELUARAN
3	7	9	
9	3	7	
7	9	3	

A	B	C	KELUARAN
9	7	3	
3	9	7	
7	3	9	

Untuk memeriksa keberanan dari program mencari nilai terbesar, semua kombinasi nilai A, B dan C di atas harus menghasilkan nilai yang benar.

3. Selanjutnya, tuliskan dan jalankan Program 5.2 berikut ini dan inputkan beberapa kombinasi **nilai A, B dan C** seperti pada tabel. Catat hasilnya!

Program 5.2 terbesar3bil_alt2.cpp

```
 1  #include <stdio.h>
 2  main()
 3  {
 4      int A, B, C, max;
 5      printf("Input 3 buah bilangan\n");
 6      printf("Bilangan 1: "); scanf("%i", &A);
 7      printf("Bilangan 2: "); scanf("%i", &B);
 8      printf("Bilangan 3: "); scanf("%i", &C);
 9
10      max = 0;
11      printf("\nBilangan terbesar: ");
12      if (A > max) {
13          max = A;
14      }
15      if (B > max) {
16          max = B;
17      }
18      if (C > max) {
19          max = C;
20      }
21      printf("%i", max);
22
23      return 0;
24  }
```

Ujicoba Program

A	B	C	KELUARAN
3	7	9	
9	3	7	
7	9	3	

A	B	C	KELUARAN
9	7	3	
3	9	7	
7	3	9	

Apakah Anda menemukan kelemahan/kekurangan dari program di atas? Tuliskan jika ada.

4. Sekarang tuliskan dan jalankan Program 5.3 berikut ini dan inputkan beberapa kombinasi **nilai A, B dan C** seperti pada tabel sebelumnya. Catat hasilnya!

Program 5.3 terbesar3bil_alt3.cpp

```
1  #include <stdio.h>
2  main()
3  {
4      int A, max;
5      printf("Input 3 buah bilangan\n");
6      printf("Bilangan 1: "); scanf("%i", &A);
7      max = A;
8      printf("Bilangan 2: "); scanf("%i", &A);
9      if (A > max) {
10         max = A;
11     }
12     printf("Bilangan 3: "); scanf("%i", &A);
13     if (A > max) {
14         max = A;
15     }
16
17     printf("\nBilangan terbesar: %i", max);
18
19     return 0;
20 }
```

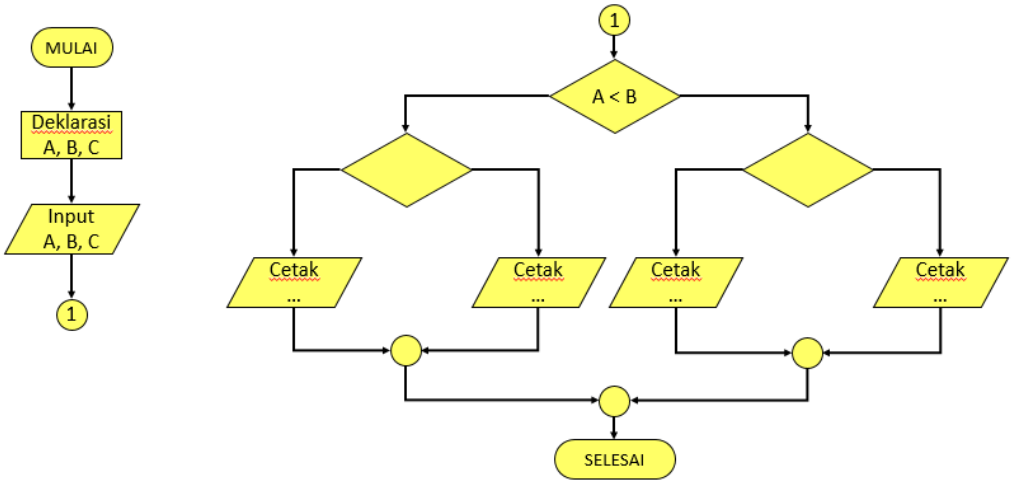
Ujicoba Program

A	B	C	KELUARAN
3	7	9	
9	3	7	
7	9	3	

A	B	C	KELUARAN
9	7	3	
3	9	7	
7	3	9	

Apakah Anda menemukan kelemahan/kekurangan dari program di atas? Tuliskan jika ada.

5. Apakah masih ada alternatif penyelesaian lain, untuk mencari nilai terbesar? Yakinlah. Selalu ada cara yang lain. Lengkapi **flowchart** di bawah ini dan buatlah **program** Bahasa C berdasarkan flowchart tersebut. **Ujilah** dengan sejumlah nilai seperti langkah sebelumnya!



Tuliskan program yang Anda buat!

6. Sekarang kita belajar Struktur SWITCH…CASE. Tuliskan dan jalankan Program 5.4 berikut ini dan inputkan beberapa kombinasi nilai **grade** dan **sks** seperti pada tabel. Catat hasilnya!

Program 5.4 angka_mutu.cpp

```
#include <stdio.h>
main()
{
    char grade;
    int sks;
    int angka_mutu = 0;

    printf("Program Hitung Angka Mutu");
    printf("\nInput Grade (A, B, C, D, E) : ");
    grade = getchar();
    printf("Input SKS : ");
    scanf("%i", &sks);

    switch(grade) {
        case 'A' : angka_mutu = 4 * sks; break;
        case 'B' : angka_mutu = 3 * sks; break;
        case 'C' : angka_mutu = 2 * sks; break;
        case 'D' : angka_mutu = 1 * sks; break;
        case 'E' : angka_mutu = 0 * sks; break;
        default  : angka_mutu = 0;
    }

    printf("\nGrade : %c", grade);
    printf("\nSKS : %i", sks);
    printf("\nAngka Mutu : %i", angka_mutu);
    return 0;
}
```

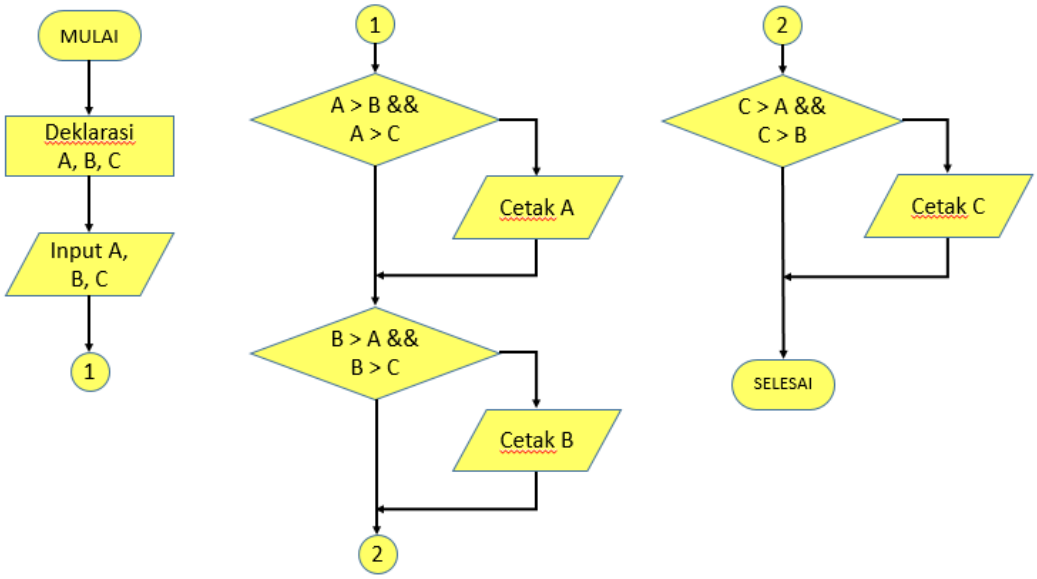
Ujicoba Program

Grade	SKS	KELUARAN
A	3	
B	2	
C	3	

Grade	SKS	KELUARAN
b	2	
-	3	
X	2	

5.4 LATIHAN

1. Buatlah program bahasa C berdasarkan flowchart / algoritma berikut ini!



Tuliskan Program dari Flowchart di atas!

5.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Tulis program untuk menentukan lama bekerja seorang pegawai, jika jam masuk dan jam pulang diinput. Catatan: jam berupa angka 1-12, dan seorang pegawai bekerja kurang dari 12 jam.
   Contoh keluaran :

Jam masuk	Jam keluar	Keluaran/tampilan
10	11	Lama bekerja 1 jam
10	2	Lama bekerja 4 jam
10	7	Lama bekerja 9 jam

2. Buatlah program dalam bahasa C untuk menyelesaikan masalah berikut :
   Program akan menerima masukan berupa kode, jenis dan harga, dengan jenis adalah "A", "B", dan "C". Untuk setiap jenis, masing-masing akan diberikan diskon sebesar 10% untuk A, 15% untuk B, dan 20% untuk C. Program akan menghitung berapa harga setelah didiskon.
   Contoh masukan :

```
Kode    = 10
Jenis   = B
Harga   = 10000
```

   Contoh keluaran :

```
Jenis barang B mendapat diskon = 15%, Harga setelah didiskon
= 8500
```

3. Tulis program untuk menentukan biaya parkir yang dihitung berdasarkan lama parkir. Lama parkir dihitung dari selisih jam masuk dan jam keluar diinput. Biaya parkir 2 jam pertama 2000, perjam berikutnya 500.
   Contoh keluaran

Jam masuk	Jam keluar	Lama	keluaran/tampilan
10	11	1	Biaya = 2000
10	2	4	Biaya = 3000

PRAKTIKUM 6
STRUKTUR PERULANGAN FOR

6.1 TUJUAN PRAKTIKUM

Tujuan Umum

Mahasiswa dapat memahami:

1. Bentuk umum struktur perulangan FOR.
2. Penggunaan struktur perulangan FOR pada sebuah program.

Tujuan Khusus

Mahasiswa dapat :

1. Menuliskan program yang menggunakan struktur perulangan FOR.
2. Menulis program untuk alur yang dinyatakan dalam bentuk Flowchart atau pseudocode.

6.2 TEORI SINGKAT

Struktur Perulangan digunakan untuk menyelesaikan persoalan yang melibatkan suatu proses yang dikerjakan beberapa kali sesuai pola tertentu. Dengan kata lain, melalui struktur perulangan memungkinkan pemrogram untuk menjalankan satu atau beberapa perintah yang ada di dalam blok perulangan secara berulang sesuai dengan nilai yang ditentukan atau sampai mencapai sebuah batas tertentu.

Sebagai contoh, jika diminta membuat program untuk menginput 3 buah nilai dan mencetak total dari ketiga buah nilai tersebut, tentunya dengan mudah kita cukup mendeklarasikan 3 buah variabel untuk menampung masing-masing nilai yang diinput untuk selanjutnya dihitung totalnya. Bagaimana jika yang diminta adalah 100 atau 1000 buah nilai? Apakah kita harus mendeklarasikan variabel dan menuliskan 100 atau 1000 perintah untuk menginput nilai tersebut? Proses tersebut dapat dilakukan dengan lebih mudah menggunakan perulangan.

6.2.1 Struktur Perulangan FOR

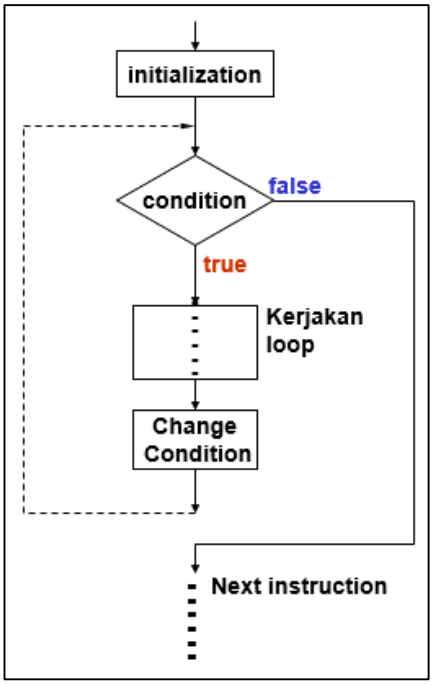
Struktur perulangan FOR biasanya digunakan untuk mengulang suatu proses yang telah diketahui jumlah perulangannya. Dari segi penulisannya, struktur perulangan FOR memiliki bentuk yang sederhana.

Bentuk umum struktur perulangan FOR adalah :

```
for (inisialisasi; kondisi; perubahan-kondisi) {
    blok-perintah-for;
}
```

Catatan:

- **inisialisasi**: Instruksi pemberian suatu nilai yang mempengaruhi nilai kondisi. Pada proses yang normal, pemberian nilai awal ini akan menyebabkan kondisi bernilai TRUE. Instruksi ini hanya pernah satu kali dilaksanakan, yaitu hanya pada saat awal struktur FOR dijalankan.
- **kondisi**: Suatu kondisi yang bernilai TRUE atau FALSE, dan akan membatasi proses perulangan. Blok perintah pada struktur perulangan akan dijalankan selama **kondisi** masih bernilai TRUE.
- **perubahan-kondisi**: Suatu instruksi yang dapat mempengaruhi nilai kondisi. Pada proses yang normal, perubahan nilai disini suatu saat akan membuat **kondisi** bernilai FALSE.



6.3 PELAKSANAAN PRAKTIKUM

1. Tuliskan Program 6.1 berikut ini pada editor Dev-C++ (program ini merupakan program untuk mencari nilai terbesar dari 3 buah bilangan yang diinput).

Program 6.1 terbesar3bil.cpp

```
 1  #include <stdio.h>
 2  main()
 3  {
 4      int A, max;
 5      printf("Input 3 buah bilangan\n");
 6      printf("Bilangan 1: "); scanf("%i", &A);
 7      max = A;
 8      printf("Bilangan 2: "); scanf("%i", &A);
 9      if (A > max) {
10          max = A;
11      }
12      printf("Bilangan 3: "); scanf("%i", &A);
13      if (A > max) {
14          max = A;
15      }
16
17      printf("\nBilangan terbesar: %i", max);
18
19      return 0;
20  }
```

2. Program 6.1 di atas digunakan untuk mencari nilai terbesar dari 3 buah bilangan yang diinput. Bagaimana jika bilangan yang diinput ada 5? Kita cukup menduplikasi baris 8 sampai 11 atau 12 sampe 15, sedemikian hingga akan menginput 5 buah bilangan. Ubahlah program 6.1 menjadi Program 6.2 berikut ini.

Program 6.2 terbesar5bil.cpp

```
 1  #include <stdio.h>
 2  main()
 3  {
 4      int A, max;
 5      printf("Input 5 buah bilangan\n");
 6      printf("Bilangan 1: "); scanf("%i", &A);
 7      max = A;
 8      printf("Bilangan 2: "); scanf("%i", &A);
 9      if (A > max) {
10          max = A;
11      }
12      printf("Bilangan 3: "); scanf("%i", &A);
13      if (A > max) {
14          max = A;
15      }
16      printf("Bilangan 4: "); scanf("%i", &A);
17      if (A > max) {
18          max = A;
19      }
20      printf("Bilangan 5: "); scanf("%i", &A);
21      if (A > max) {
22          max = A;
23      }
24
25      printf("\nBilangan terbesar: %i", max);
26
27      return 0;
28  }
```

3. Jalankan dan ujilah program 6.2 di atas dengan beberapa data. Tuliskan pada tabel di bawah ini.

Bil1	Bil2	Bil3	Bil4	Bil5	KELUARAN

4. Dengan tujuan yang sama (mencari nilai terbesar), bagaimana jika yang diinput 10 bilangan, 100 bilangan atau lebih? Dengan cara sebelumnya tentunya akan sangat repot, dan program menjadi sangat panjang. Perhatikan Program 6.3 berikut ini, tuliskan pada editor dan jalankan!

Program 6.3 terbesar_n_bil.cpp

```
#include <stdio.h>
main()
{
    int A, max, i;
    printf("Input 10 buah bilangan\n");
    printf("Bilangan pertama: ");
    scanf("%i", &A);
    max = A;
    for(i=1; i<=9; i++) {
        printf("Bilangan ke-%i: ", (i+1));
        scanf("%i", &A);
        if (A > max) {
            max = A;
        }
    }

    printf("\nBilangan terbesar: %i", max);

    return 0;
}
```

5. Ujicoba program 6.3 dengan data masukan berikut ini dan tuliskan perubahan nilai masing-masing variabel / kondisi pada tabel.
   Data masukan: 10 7 15 12 9 5 7 18 3 11

i	i<=9	Input A	A > max	max	Keterangan
-	-	10	-	10	Inisialisasi
1	TRUE	7	FALSE	10	Perulangan ke-1
2	TRUE	15	TRUE	15	Perulangan ke-2
3	TRUE				
4					
5					
6					
7					
8					
9					
10	FALSE				Keluar dari perulangan

6. Berdasarkan Program 6.3 di atas, jawablah beberapa pertanyaan berikut ini!

NO	PERTANYAAN	JAWABAN
1	Pada struktur FOR, tuliskan perintah bagian **inisialisasi**!	
2	Pada struktur FOR, tuliskan perintah yang menunjukkan **kondisi** akhir perulangan!	
3	Berapa kali perintah-perintah dalam blok FOR dijalankan?	
4	Jika diinginkan bilangan yang diinput menjadi 20 bilangan, bagian perintah mana yang harus diganti?	
5	Berdasarkan program 6.3 dan data inputan pada langkah ke-5, berapa nilai variabel i setelah keluar dari perulangan?	

7. Tuliskan dan jalankan program 6.4 berikut ini! Lalu tuliskan hasilnya pada tempat yang sudah tersedia.

Program 6.4 deret_ganjil_cara1.cpp

```
1  #include <stdio.h>
2  main()
3  {
4      int i;
5      printf("10 Bilangan Ganjil Pertama\n");
6      for(i=1; i<=19; i=i+2) {
7          printf("%4i", i);
8      }
9
10     return 0;
11 }
```

Tuliskan hasilnya

8. Program 6.4. merupakan program untuk menampilkan 10 bilangan ganjil yang dimulai dari 1. Program 6.5 merupakan program cara kedua untuk tujuan yang sama. Walaupun keduanya menghasilkan keluaran yang sama, namun cara kedua lebih mudah dipahami dan lebih disarankan.
   Tuliskan dan jalankan program 6.5 berikut ini. Catat hasilnya!

Program 6.5 deret_ganjil_cara2.cpp

```
1  #include <stdio.h>
2  main()
3  {
4      int i, N=1;
5      printf("10 Bilangan Ganjil Pertama\n");
6      for(i=1; i<=10; i++) {
7          printf("%4i", N);
8          N = N + 2;
9      }
10
11     return 0;
12 }
```

Tuliskan hasilnya

9. Jika program 6.5 diubah menjadi program 6.6 berikut ini, apa yang terjadi? Tuliskan dan jalankan programnya, lalu catat apa yang terjadi. Mengapa?

Program 6.6 deret_ganjil_cara2.cpp

```
1  #include <stdio.h>
2  main()
3  {
4      int i, N=1;
5      printf("10 Bilangan Ganjil Pertama\n");
6      for(i=1; i<=10; i--) {
7          printf("%4i", N);
8          N = N + 2;
9      }
10
11     return 0;
12 }
```

Tuliskan hasil program dan komentar Anda!

6.4 LATIHAN

1. Tuliskan dan jalankan program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

Program 6.7 deret_genap.cpp

```
 1  #include <stdio.h>
 2  main()
 3  {
 4      int i, N=10;
 5      printf("Deret Bilangan Genap\n");
 6      for(i=1; i<=10; i++) {
 7          printf("%4i", N);
 8          N = N - 2;
 9      }
10
11      return 0;
12  }
```

Hasil Program 6.8

Gambarkan Flowchart dari Program 6.8

Program 6.8 cari_bilangan.cpp

```
 1  #include "stdio.h"
 2  main()
 3  {
 4      int A[10] = {10,5,7,15,12,20,9,7,11,14};
 5      int N, I, flag = 0;
 6      printf("Bilangan\n");
 7      for (I=0; I<10; I++)
 8          printf("%3i", A[I]);
 9      printf("\n");
10
11      printf("Input bilangan yang dicari: ");
12      scanf("%i", &N);
13
14      for (I=0; I<10; I++) {
15          if (A[I] == N) {
16              flag = 1;
17          }
18      }
19      if (flag == 1) {
20          printf("Bilangan %i ADA ditemukan.", N);
21      } else {
22          printf("Bilangan %i TIDAK ditemukan.", N);
23      }
24
25      return 0;
26  }
```

Hasil Program 6.9

6.5 TUGAS MANDIRI

Kerjakan soal-soal berikut ini:

1. Buatlah program Bahasa C untuk mencetak deret berikut ini:
   10 20 30 40 50 60 70 80 90 100

2. Buatlah program Bahasa C untuk mencetak deret berikut ini:
   100 95 90 85 80 75 70 65 60 55

3. Buatlah program Bahasa C untuk mencetak deret berikut ini:
   1 2 4 8 16 32 64 128 256 512 1024

4. Seseorang mengendarai sepeda dengan kecepatan tetap 2 meter/detik. Susun program untuk mencetak berapa meter yang dia tempuh setelah bersepeda selama 100 detik.

5. Seseorang menyimpan uang Rp. 1.000.000 di bank dengan bunga ber-bunga 2% perbulan. Jadi setelah satu bulan uangnya menjadi Rp. 1.020.000. Satu bulan berikutnya uang Rp. 1.020.000 ini mendapat bunga lagi 2%, yaitu Rp.20.400 sehingga setelah 2 bulan uangnya menjadi Rp. 1.020.000 + Rp. 20.400 = Rp. 1.040.400. Demikian seterusnya (bunga bulan ini ditambahkan ke saldo uangnya dan mendapatkan bunga lagi pada bulan berikutnya) . Susun program untuk menghitung dan mencetak jumlah uangnya setelah 10 bulan.