

ALGORITMA DAN PEMROGRAMAN 1

Penyusun:

Hendri Ardiansyah, S.Kom., M.kom
Resti Amalia, S.Kom., M.Kom
Agus Budi Prasetyo, S.Kom., M.Kom



Gd. A; R. 212 Universitas Pamulang
Jl. Surya Kencana No. 1 Pamulang | Tangerang Selatan | Banten

ALGORITMA DAN PEMROGRAMAN 1

Penulis :

Hendri Ardiansyah, S.Kom., M.kom
Resti Amalia, S.Kom., M.Kom
Agus Budi Prasetyo, S.Kom., M.Kom

ISBN : 978-602-5867-81-1

Editor :

Saprudin

Desain Sampul:

Ubaid Al Faruq, M.Pd.

Tata Letak:

Aden

Penerbit:

Unpam Press

Redaksi:

Jl. Surya Kencana No. 1

R. 212, Gd. A Universitas Pamulang Pamulang | Tangerang Selatan | Banten

Tlp/Fax: 021. 741 2566 – 7470 9855 Ext: 1073

Email: unpampress@unpam.ac.id

Cetakan pertama, 20 Desember 2019

Hak cipta dilindungi undang-undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara apapun tanpa izin penerbit.

LEMBAR IDENTITAS ARSIP

**Data Publikasi Unpam Press
I Lembaga Pengembangan Pendidikan dan Pembelajaran Universitas Pamulang**

Gedung A. R.212 Kampus 1 Universitas Pamulang
Jalan Surya Kencana No.1, Pamulang Barat, Tangerang Selatan, Banten.
Website : www.unpam.ac.id | email : unpampress@unpam.ac.id

Algoritma dan Pemrograman 1/ Hendri Ardiansyah, S.Kom., M.Kom, Resti Amalia, S.Kom., M.Kom., Agus Budi Prasetyo, S.Kom., M.Kom..-1sted.
ISBN 978-602-5867-81-1

1. Algoritma dan Pemrograman 1 I. Hendri Ardiansyah, S.Kom., M.Kom. II. Resti Amalia, S.Kom., M.Kom. III. Agus Budi Prasetyo, S.Kom., M.Kom.
M076-20122019-01

Ketua Unpam Press : Pranoto
Koordinator Editorial dan Produksi: Ubaid Al Faruq, Ali Madinsyah
Koordinator Bidang Hak Cipta : Susanto
Koordinator Publikasi dan Dokumentasi : Aden
Desain Cover : Ubaid Al Faruq

Cetakan pertama, 20 Desember 2019

Hak cipta dilindungi undang-undang. Dilarang menggandakan dan memperbanyak sebagian atau seluruh buku ini dalam bentuk dan dengan cara apapun tanpa ijin penerbit.

MATA KULIAH
ALGORITMA DAN PEMROGRAMAN 1

IDENTITAS MATA KULIAH

Program Studi	: Teknik Informatika S-1
Mata Kuliah / Kode	: Algoritma dan Pemrograman 1 /TPL0022
Sks	: 2 Sks
Prasyarat	: --
Deskripsi Mata Kuliah	: Mata kuliah ini merupakan mata kuliah wajib Program Studi Teknik Infomatika S-1 yang membahas tentang Pengantar Algoritma, Dasar-dasar Algoritma, Tipe Data, Ekspresi, Operator, Array dan Pengambilan Keputusandan fiskal, pengantar ekonomi pembangunan
Capaian Pembelajaran	: Setelah menyelesaikan mata kuliah ini mahasiswa mampu membuat aplikasi sederhana dengan menerapkan algoritma secara runtunan, pemilihan dan pengulangan

Ketua Program Studi
Teknik Informatika S-1

Ketua Tim Penulis

Syaeful Bakhri, ST., M.Eng.Sc., Ph.D
NIDN 0421127402

Hendri Ardiansyah, S.Kom., M.Kom
NIDN 0401038601

KATA PENGANTAR

Alhamdulillahi robbil 'aalamin, puji syukur kehadirat Allah SWT yang telah memberikan taufiq, rahmat dan hidayah kepada penulis serta atas izin-Nya sehingga modul perkuliahan ini dapat diselesaikan sesuai dengan target. Dalam penulisan modul perkuliahan ini telah disesuaikan dengan RPS (Rencana Pembelajaran Semester) Algoritma dan Pemrograman 1.

Modul perkuliahan ini digunakan untuk mata kuliah algoritma dan pemrograman 1 pada jurusan teknik informatika yang terbagi menjadi 14 pertemuan dimulai dari pengenalan algoritma dan pemrograman sampai dengan mengimplementasikan algortima untuk membuat program sederhana.

Setelah mempelajari modul perkuliahan ini penulis berharap para mahasiswa mampu menguasai konsep dasar-dasar pemrograman sebagai pedoman untuk membuat aplikasi sederhana sesuai dengan konsep algoritma dan pemrograman.

Ucapan terima kasih penulis ucapkan kepada berbagai pihak yang telah membantu dalam pembuatan modul perkuliahan ini, khususnya kepada teman-teman dosen teknik informatika yang telah berkenan untuk mengoreksi kebenaran naskah ini.

Tangerang Selatan, 20 Desember 2019

Penulis

DAFTAR ISI

ALGORITMA DAN PEMROGRAMAN 1	i
LEMBAR IDENTITAS ARSIP	ii
IDENTITAS MATA KULIAH.....	iii
KATA PENGANTAR.....	iv
DAFTAR ISI	v
DAFTAR TABEL.....	x
DAFTAR GAMBAR.....	xi
PERTEMUAN 1 PENGANTAR ALGORITMA DAN PEMROGRAMAN	12
A. Tujuan Pembelajaran.....	12
B. Uraian Materi	12
1. Pengertian Algoritma	12
2. Kalimat Deskriptif	14
3. Flow Chart.....	15
4. Pseudo Code.....	18
C. Soal Latihan / Tugas	23
D. Referensi.....	24
PERTEMUAN 2 SISTEM BILANGAN BINER.....	25
A. Tujuan Pembelajaran.....	25
B. Uraian Materi	25
1. Karakter.....	25
2. Byte	25
3. Biner	27
C. Latihan tugas	35
D. Referensi.....	36
PERTEMUAN 3 TIPE DATA, VARIABEL DAN KONSTANTA	37
A. Tujuan Pembelajaran.....	37
B. Uraian Materi	37
1. Variabel	37
2. Konstanta	39
3. Tipe Data.....	39
a. Tipe Data <i>numeric integer</i> (bilangan bulat).....	40
b. Tipe Data Floating-Point (Pecahan).....	43
c. Tipe Data Karakter	44

d. Tipe Data Boolean.....	46
e. Modifikasi Tipe Data.....	46
f. Rangkuman Tipe Data	46
g. Deklarasi Variabel dengan Tipe Data	47
C. Soal Latihan / Tugas.....	50
D. Referensi.....	51
PERTEMUAN 4 ASSIGNMENT STATEMENT, ARITMETIC EXPRESSION DAN OPERATOR.....	52
A. Tujuan Pembelajaran.....	52
B. Uraian Materi	52
1. Assignment Statement.....	52
2. Arithmetic Expression dalam Assignment Statement	55
3. Program Flowchart.....	59
C. Soal/Latihan	61
D. Referensi.....	61
PERTEMUAN 5 PREPROCESSOR DAN LIBRARY FUNCTION.....	63
A. Tujuan Pembelajaran.....	63
B. Uraian Materi	63
1. Definisi Preprocessor.....	63
2. Definisi Library Function	66
3. Penggunaan Library Function.....	70
C. Soal Latihan/Tugas.....	72
D. Referensi.....	73
PERTEMUAN 6 INPUT, OUTPUT, ALGORITMA DAN PENGETAHUAN TERKAIT ...	74
A. Tujuan Pembelajaran.....	74
B. Uraian Materi	74
1. Output.....	74
a. Mencetak konstanta tanpa format.....	74
b. Mencetak konstanta dengan format.....	75
c. Manipulator.....	76
d. Tanda Format.....	77
2. Input.....	78
a. Menginput karakter melalui keyboard.....	81
b. Menginput String	83
c. Menginput nilai numerik melalui keyboard.....	84
3. Algoritma Merepresentasikan apa yang diketahui	84

C. Soal Latihan / Tugas	86
D. Referensi.....	87
PERTEMUAN 7 CONTROL STATEMENT MENGGUNAKAN IF.....	88
A. Tujuan Pembelajaran.....	88
B. Uraian Materi	88
1. Definisi.....	88
Listing:.....	90
C. Soal Latihan/Tugas.....	103
D. Referensi.....	103
PERTEMUAN 8 CONTROL STATEMENT IF (Lanjutan)	104
A. Tujuan Pembelajaran.....	104
B. Uraian Materi	104
1. Pendahuluan	104
2. Jenis-Jenis Decission If	104
3. Alur Kerja Perintah IF.....	115
C. Soal/Latihan	119
D. Referensi.....	120
PERTEMUAN 9 CONTROL STATEMENT MENGGUNAKAN NESTED IF SWITCH dan LOGICAL OPERATOR.....	121
A. Tujuan Pembelajaran.....	121
B. Uraian Materi	121
1. Nested if	121
2. Milti Condition dan Logical Operator	126
3. Konversi Multi Condition menjadi Nested If.....	129
4. Contoh Program Sederhana Menggunakan Nested If Dan Multi Condition	130
5. Switch & Case Berjenjang	131
C. Soal Latihan	134
D. Referensi.....	134
PERTEMUAN 10 PERULANGAN MENGGUNAKAN FOR, WHILE	135
A. Tujuan Pembelajaran.....	135
B. Uraian Materi	135
1. For	135
2. While.....	140
C. Soal Latihan :	144
D. Referensi.....	145

PERTEMUAN 11 PERULANGAN MENGGUNAKAN DO WHILE DAN NESTED LOOP	
146	
A. Tujuan Pembelajaran.....	146
B. Uraian Materi	146
1. Do ... While.....	146
2. Loop dalam Loop (Nested Loop)	149
3. Contoh – Contoh Penggunaan Nested Loop	152
C. Soal Latihan / Tugas.....	157
D. Referensi.....	158
PERTEMUAN 12 ARRAY SATU DIMENSI	159
A. Tujuan Pembelajaran.....	159
B. Uraian Materi	159
1. Array Satu Dimensi.....	159
2. Deklarasi Array.....	161
3. Menyiapkan Array satu dimensi pada Bahasa pemrograman C++	162
4. Alamat elemen – elemen Array satu dimensi.....	163
5. Membuat Array Numerik Dengan Nilai Awal	166
C. Soal Latihan / Tugas.....	169
D. Referensi.....	170
PERTEMUAN 13 ARRAY DUA DIMENSI	171
A. Tujuan Pembelajaran.....	171
B. Uraian Materi	171
1. Kelebihan & Kekurangan Array	171
2. Definisi.....	172
3. Alur Kerja Array 2 Dimensi.....	176
4. Flowchart.....	178
5. Pejumlahan Array.....	178
6. Perkalian Array 2 Dimensi	180
C. Soal Latihan/ Tugas.....	182
D. Referensi.....	182
PERTEMUAN 14 ALGORITMA DAN PEMROGRAMAN DALAM MATEMATIKA.....	184
A. Tujuan Pembelajaran.....	184
B. Uraian Materi	184
1. Menentukan suatu bilangan genap atau ganjil.....	184
2. Konversi bilangan desimal ke bilangan biner	185
3. Menentukan suatu bilangan merupakan bilangan prima	186

4. Mencari faktor persekutuan terbesar	189
5. Menghitung luas suatu area/bidang yang dibatasi oleh garis	190
C. Soal Latihan / Tugas	192
D. Referensi	192
GLOSARIUM	193
REFERENSI	194

DAFTAR TABEL

Tabel 1. 1 Simbol-Simbol Flowchart	15
Tabel 1. 2 Pseudo Code.....	18
Tabel 2. 1 Konversi Bilangan Biner ke desimal	30
Tabel 3. 1 Keyword Dalam ANSI Bahasa C++/C	38
Tabel 3. 2 Contoh Penamaan Variabel.....	39
Tabel 3. 3. Tipe Data Pada C++	46

DAFTAR GAMBAR

Gambar 1. 1 Flowchar Menghitung luas lingkaran	17
Gambar 1. 2 Flowchart menentukan bilangan ganjil atau genap	18
Gambar 1. 3 Flowchar Algoritma Euclidean	23
Gambar 2. 1 Satuan Memory	27
Gambar 3. 1 Kategori Tipe Data Dasar	40
Gambar 3. 2. Jangkauan Nilai Tipe data short	40
Gambar 3. 3 Jangkauan Nilai Tipe data int.....	42
Gambar 3. 4 Jangkauan Nilai Tipe data long	42
Gambar 3. 5. Jangkauan Nilai Tipe data long long	43
Gambar 3. 6. Kode ASCII.....	45

PERTEMUAN 1

PENGANTAR ALGORITMA DAN PEMROGRAMAN

A. Tujuan Pembelajaran

Setelah mengikuti materi pada pertemuan ke-1 ini mahasiswa mampu menuliskan algoritma dalam bentuk *flowchart*, *pseudo code*, dan kalimat deskriptif.

B. Uraian Materi

1. Pengertian Algoritma

Dalam pengertian Algoritma merupakan urutan atau alur dan langkah-langkah dalam menyelesaikan dan mengetahui masalah yang bersifat logis dan juga sistimatis. Dalam kehidupan sehari-hari, apabila kita amati, sebenarnya kita telah melakukan kaidah-kaidah algoritma. Sebagai contoh adalah saat kita memasak mie instan. Pada bagian belakang bungkus mie instan apabila kita lihat, disitu terdapat langkah demi langkah bagaimana cara memasak dan juga menyajikannya. Apabila langkah-langkah yang dijabarkan tidaklah logis maka kita akan mendapatkan hasil yang pasti tidak akan sesuai seperti yang diharapkan.

Dan juga algoritma tidak harus mengikuti langkah-langkah baku seperti perhitungan dimatematika. Algoritma mengajarkan bagaimana memecahkan masalah dengan berbagai solusi dan memilih mana solusi yang terbaik.

Berikut ini adalah ciri-ciri yang dimiliki dalam sebuah algoritma yaitu :

a. Kepastian

Langkah - langkah yang dijabarkan harus pasti dan tidak bermakna ganda.

b. Batasan

Batasan dipakai agar algoritma berakhir setelah menjalankan sejumlah proses dan langkah-langkah.

c. Efektif

Efektif yang dimaksud disini adalah Instruksi yang dijalankan dengan efektif.

d. Masukan

Algoritma tidak harus memiliki satu saja masukan, tapi algoritma bisa memiliki nol atau lebih masukan.

e. Keluaran

Keluaran yang dimiliki paling tidak menghasilkan satu luaran.

Algoritma memiliki sifat-sifat sebagai berikut :

- a. Symbol dan sintaks tidak harus baku dan tidak harus dari suatu bahasa pemrograman tertentu..
- b. Tidak bergantung kepada suatu bahasa pemrograman tertentu.
- c. Urutan dan notasi-notasinya dapat digunakan untuk ditranslasikan pada bahasa pemrograman manapun.
- d. Algoritma ini dapat diterapkan disemua kejadian sehari-hari dan dapat dipakai untuk mewakili suatu urutan kejadian secara logis .

Tiga struktur dasar dari algoritma :

a. Runtunan (Sequence)

Setiap instruksi dalam algoritam dijalankan secara berurutan (step by step).

b. Pemilihan (Selection)

Instruksi akan dijalankan apabila persyaratan terpenuhi atau bernilai benar (true), jika instruksi bernilai salah (false), maka instruksi ini tidak akan dijalankan.

c. Pengulangan (repetition)

Pengerjaan instruksi yang berulang sesuai dengan kondisi yang telah ditentukan.

Dalam penulisan algoritma atau yang biasa disebut notasi algoritmik, tidak ada hal baku didalam menuliskannya. Suatu notasi Algoritma bukanlah notasi pada bahasa didalam pemrograman sehingga siapapun itu orangnya dapat membuat notasi algoritma yang berbeda-beda. Akan tetapi aturan-aturan dan kaidah-kaidah harus ditaati untuk menghindari kekeliruan.

Dibawah ini adalah beberapa notasi yang sering digunakan dalam menulis dan mendeskripsikan algoritma diantaranya yaitu :

- a. Suatu kalimat deskriptif
- b. Pseudo Code
- c. Flow chart

Dalam dunia komputer, algoritma, bahasa pemrograman dan juga program sangatlah berhubungan dengan erat. Bahasa didalam Pemrograman adalah bahasa yang digunakan komputer untuk penulisan suatu program. Sedangkan program adalah beberapa atau sekumpulan instruksi yang berupa pernyataan yang kemudian ditulis dengan menggunakan bahasa computer atau bahasa pemrograman yang melibatkan pemilihan struktur data. Algoritma yang baik tanpa pemilihan struktur data yang tepat akan membuat program menjadi kurang baik, demikian juga sebaliknya.

2. Kalimat Deskriptif

Sebuah algoritma dapat dituliskan dalam berbagai cara, bahkan dengan bahasa sehari-hari. Algoritma dapat dituliskan dalam kalimat deskriptif yang menjelaskan instruksi atau langkah-langkah yang akan dilakukan dengan jelas. Penulisan algoritma dengan kalimat deskriptif adalah yang paling paling mudah dan sederhana untuk dibuat namun memiliki beberapa kekurangan. Kalimat deskriptif dapat bersifat ambigu, dan oleh karena itu, kalimat deskriptif tidak memiliki karakteristik yang pasti. Suatu algoritma harus jelas dan tidak boleh memiliki lebih dari satu makna. Kalimat deskriptif jarang sekali digunakan dalam menyelesaikan masalah-masalah yang komplek.

Teks algoritma kalimat deskriptif meliputi :

a. Head

Head atau judul memberikan nama atau apa yang akan kita buat pada algoritma; biasanya nama yang kita buat sudah dapat memberikan gambaran kepada aturan dari penyelesaian masalah dan juga masalah yang akan diselesaikan.

b. Deklarasi

Menyatakan jenis dari setiap elemen data (variabel) yang akan digunakan dalam algoritma.

c. Deskripsi

Adalah inti dari prosedur untuk penyelesaian masalah, meliputi pernyataan atau operasi, fungsi, penjelasan, dll.

Berikut contoh algoritma dengan kalimat deskriptif :

Algoritma pada Luas Lingkaran

{Program menghitung luas lingkaran dan menampilkannya ke layar dengan inputan berupa jari-jari lingkaran.}

Deklarasi :

jarling = real {jari jari lingkaran dengan tipe data bil. pecahan}

luas = real {luas lingkaran dengan tipe data bil. pecahan}

phi = 3.14

Deskripsi:

1. Baca jarling
2. Hitung luas = phi * jarling * jarling
3. Tampilkan luas ke layar
4. Selesai

3. Flow Chart

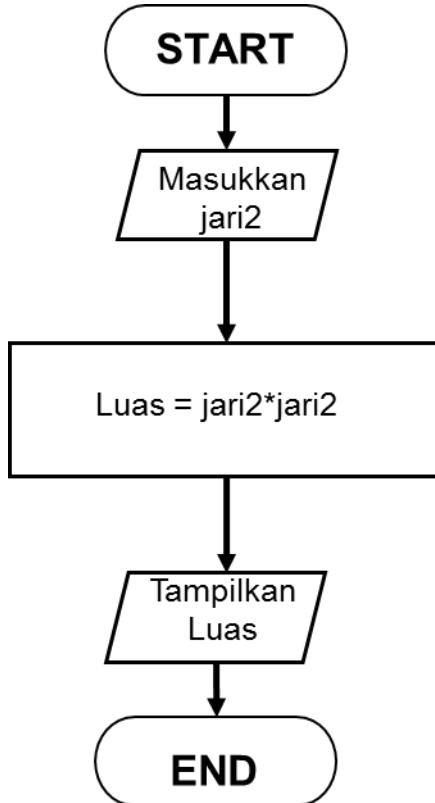
Flow chart adalah penggambaran secara grafis dari notasi algoritma. Dengan menggunakan flow chart kita dapat menggambarkan urutan atau langkah-langkah yang berisi pernyataan dalam penulisan algoritma. Flow chart berisi sekumpulan simbol-simbol yang menggambarkan proses tertentu. Suatu algoritma yang ditulis menggunakan flow chart dapat menggunakan simbol-simbol sebagai berikut :

Tabel 1. 1 Simbol-Simbol Flowchart

Simbol	Maksud
	Terminal (START, END)

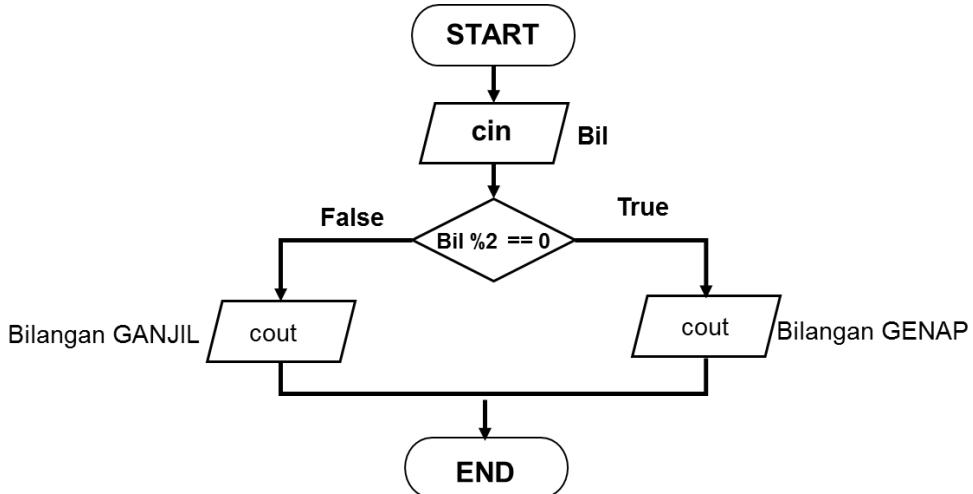
Simbol	Maksud
	Input/Output (READ, WRITE)
	Proses
	Pemilihan atau pencabangan (YES, NO)
	Menampilkan
	Alur
	Titik sambungan pada halaman yang sama
	Titik konektor yang berada pada halaman lain
	Call (Memanggil subprogram)
	Dokumen
	Stored Data (Simpanan Data)
	Preparation (Pemberian nilai awal suatu variabel)

Dibawah ini adalah suatu contoh flow chart dalam algoritma untuk menghitung luas lingkaran dimana rumus luas lingkaran adalah $\pi * \text{jari-jari}^2$. Sedangkan nilai jari-jari lingkaran akan di input melalui keyboard, dan hasil perhitungan luas akan ditampilkan di layar.



Gambar 1. 1 Flowchar Menghitung luas lingkaran

Sebagai contoh lain adalah flow chart algoritma untuk mengecek apakah sebuah bilangan yang di inputkan melalui keyboard adalah bilangan genap atau bilangan ganjil. Dimana untuk mengetahui sebuah bilangan tersebut adalah bilangan genap atau ganjil, adalah dengan cara membagi bilangan tersebut dengan angka 2, apabila bilangan tersebut habis dibagi 2 atau dengan kata lain bilangan modulus 2 sama dengan 0 maka bilangan tersebut adalah bilangan genap, jika sisa pembagiannya adalah 1 maka bilangan tersebut adalah bilangan ganjil.



Gambar 1. 2 Flowchart menentukan bilangan ganjil atau genap

4. Pseudo Code

Pseudo code adalah notasi algoritma yang menyerupai notasi bahasa pemrograman tingkat tinggi. Keuntungan menggunakan pseudocode adalah kemudahan dalam mentranslasi ke notasi bahasa pemrograman, karena terdapat kemiripan dengan notasi bahasa pemrograman. Seorang programer yang ingin menerapkan algoritma tertentu, terutama yang kompleks atau algoritma baru, biasanya akan memulainya dengan membuat deskripsi dalam bentuk pseudocode. Setelah pseudocode tersebut jadi, maka langkah selanjutnya hanya tinggal menterjemahkannya ke bahasa pemrograman tertentu.

Penulisan algoritma menggunakan pseudo code dapat menggunakan notasi-notasi sebagai berikut :

Tabel 1. 2 Pseudo Code

Pernyataan	Notasi algoritmik	Maksud
Penulisan	Cout (x)	Nilai x dicetak di piranti keluaran
	Cout (x,y)	Nilai x dan y dicetak di piranti keluaran
	Cout ("Hello")	Text Hello dicetak di piranti keluaran
Pembacaan	read(a)	Baca nilai a
	read(a,b)	Baca nilai a,b
Penugasan	bil←x	Isikan nilai variabel x kedalam variabel bil

Pernyataan	Notasi algoritmik	Maksud
Komentar	// komentar	Komentar untuk 1 (satu) baris dimulai dengan //
	/* Komentar baris 1 Komentar baris 2 --- Komentar baris n */	Komentar untuk banyak baris dimulai dengan /* dan diakhiri dengan */
Ekspresi	a > b	Ekspresi boolean yang akan memiliki nilai kembalian true jika nilai a lebih besar dari nilai b , apabila tidak maka nilai kembalinya false
	a >= b	Ekspresi boolean yang akan memiliki nilai kembalian true jika a >= b , apabila tidak maka nilai kembalinya false
	a < b	Ekspresi boolean yang akan memiliki nilai kembalian true jika nilai a < nilai b , apabila tidak maka nilai kembalinya false
	a <= b	Ekspresi boolean yang akan memiliki nilai kembalian true jika nilai a <= b , apabila tidak maka nilai kembalinya false
	a == b	Ekspresi boolean yang akan memiliki nilai kembalian true jika nilai a = nilai b , apabila tidak maka nilai kembalinya false
	a != b	Ekspresi boolean yang akan memiliki nilai kembalian true jika nilai a tidak sama dengan nilai b , apabila tidak maka nilai kembalinya false
	a AND b	Ekspresi boolean yang akan memiliki nilai kembalian true jika kedua kondisi (a dan b) bernilai true , apabila tidak maka nilai kembalinya false
	a OR b	Ekspresi boolean yang akan memiliki nilai kembalian true jika salah satu

Pernyataan	Notasi algoritmik	Maksud
		kondisi (a dan b) bernilai true , apabila tidak maka nilai kembalinya false
	Not a	Ekspresi boolean yang akan memiliki nilai kembalian true jika hasil evaluasi nilai a adalah false , apabila tidak maka nilai kembalinya false
Kondisi	If <kondisi> then <pernyataan>	Jika kondisi true / benar maka pernyataan akan dijalankan
	if <condi> then <case 1> else <case 2>	apabila kondisi true / benar maka pernyataan 1 akan dijalankan sebaliknya apabila kondisi false / salah maka pernyataan 2 yang akan dijalankan.
Pengulangan	while <cond> do { case }	Pengulangan pernyataan akan dijalankan selama kondisi true / benar apabila kondisi false / salah maka pengulangan akan dihentikan
	repeat { case } until <cond>	Pengulangan pernyataan akan dijalankan selama kondisi true / benar apabila kondisi false / salah maka pengulangan akan dihentikan
	for variable = nilai1 to nilai2 { pernyataan }	Pengulangan pernyataan akan dijalankan dari variable nilai1 sampai dengan nilai2

Berisi uraian penyelesaian masalah, meliputi pernyataan atau operasi, fungsi, penjelasan, dll.

Berikut ini adalah contoh suatu algoritma menggunakan pseudo code untuk mencetak pesan lulus apabila nilai ujian adalah lebih besar dari 60.

Algoritma Lulus

{ mencetak pesan Lulus jika nilai ujian ≥ 60 .}

Deklarasi :

nilai : integer

Deskripsi:

```
read(nilai)
if nilai >= 60 then
    print('Lulus' )
else
    print('Tidak Lulus')
endif
```

Algoritma Euclidean adalah salah satu algoritma untuk mencari FPB (Faktor Persekutuan Terbesar) dari 2 buah bilangan. Langkahnya adalah dengan membagi bilangan yang lebih besar (misalkan a) dengan bilangan yang lebih kecil (b), pembagian tersebut akan menghasilkan hasil bagi h1 dan sisa s1. Jika s1 = 0, maka FPB nya adalah b. Tetapi bila s1 tidak sama dengan 0, maka prosesnya diulang lagi dengan membagi b dengan s1. Pembagian tersebut akan menghasilkan hasil bagi h2 dan sisa s2. Jika s2 = 0, maka FPB nya adalah s1 (sisa pembagian sebelumnya), jika s2 bukan 0, maka lakukan pembagian s1 dengan s2.

Berikut ini adalah contoh kalimat deskriptif, flow chart dan pseudo code untuk algoritma euclidean :

- Kalimat deskriptif algoritma euclidean

Algoritma Euclidean

{ Diberikan dua buah bilangan bulat positif a dan b ($a \geq b$).

Algoritma Euclidean mencari pembagi bersama terbesar dari kedua bilangan tersebut, yaitu bilangan positif terbesar yang habis membagi a dan b}

Deskripsi :

1. Jika $a = 0$ maka

b adalah jawabannya;

Stop.

Tetapi jika $b \neq 0$,

Lanjutkan ke langkah 2.

2. Bagilah a dengan b dan misalkan s adalah sisanya

3. Ganti nilai a dengan nilai b, nilai b dengan nilai s,

lalu ulang kembali ke langkah 1.

Pseudo code algoritma euclidean

```
{ Program mencari pembagi bilangan terbesar, a dan b bilangan  
bulat positif }
```

Deklarasi :

a, b : integer { bil yg akan dicari pbt-nya }

s : integer { sisa hasil bagi }

Deskripsi :

```
read(a,b) { $a \geq b$ }
```

```
while  $b \neq 0$  do
```

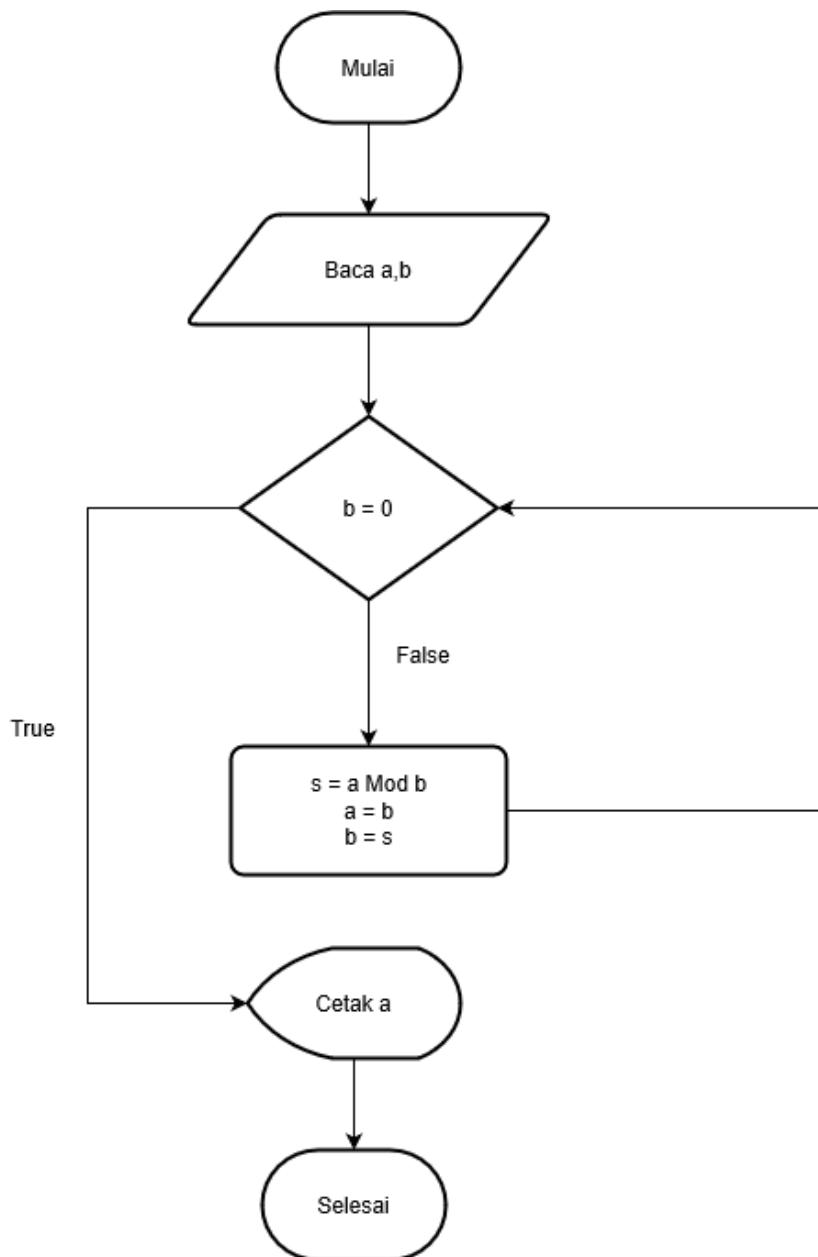
```
    s <- a Mod b {hitung sisa hasil bagi}
```

```
    a <- b
```

```
    b <- s
```

```
endwhile {kondisi selesai pengulangan:  $b=0$ , maka pbt=a}
```

```
print(a)
```

**Gambar 1. 3 Flowchar Algoritma Euclidean****C. Soal Latihan / Tugas**

1. Buatlah algoritma (dalam bentuk kalimat deskriptif, flow chart dan pseudo code) untuk menukar dua buah bilangan.
2. Buatlah algoritma untuk mencetak deret angka bilangan ganjil dari 1 sampai 20.

3. Buatlah algoritma (dalam bentuk kalimat deskriptif, flow chart dan pseudo code) untuk menginput 3 buah bilangan, kemudian tentukan bilangan terbesar, terkecil, dan rata-ratanya

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 2 SISTEM BILANGAN BINER

A. Tujuan Pembelajaran

Setelah mengikuti materi pada pertemuan ke-2 ini mahasiswa mampu:

1. Mahasiswa mengetahui tentang bilangan biner.
2. Mahasiswa mengetahui tentang konversi bilangan biner dan operasi penjumlahan bilangan biner

B. Uraian Materi

1. Karakter

Apapun yang dapat kita baca, apapun itu dan dimanapun itu seperti dilayar computer, di koran, majalah dll. Itu merupakan karakter atau bisa disebut juga gabungan dari karakter-karakter.

Contoh: **int a = 5;**

Int dan **a** : adalah karakter yang kita kenal sebagai **huruf**.

5 : adalah karakter yang kita kenal sebagai **angka**
= ; : adalah karakter-karakter khusus

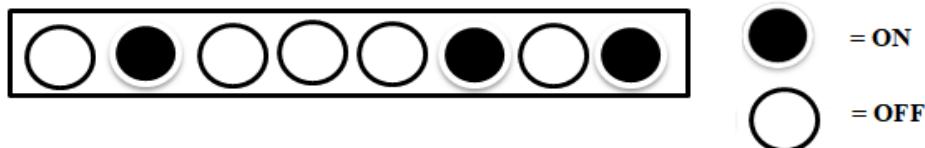
2. Byte

Byte merupakan suatu kumpulan bit yang terdiri dari 8 bit dan digabung menjadi satu. Atau bias disimpulkan, 1 byte terdiri dari 8 bit. Satuan yang biasa dipakai didalam byte untuk mengilustrasikan kapasitas sesuatu didalam suatu media penyimpanan seperti kilobyte (KB), megabyte (MB), gigabyte (GB) dan terabyte (TB). Symbol atau lambing dari bit dan byte dibedakan dengan menggunakan huruf B besar untuk. Satu byte bisa mewakili nilai yang ada antara 0 sampai 255 dalam desimal.

Satu byte dapat diilustrasikan sebagai kumpulan 8 buah bohlam lampu. pemisalan ini dilakukan karena karena bohlam lampu hanya mempunyai dua kondisi, yaitu hidup dan mati, yang mampu mewakili komponen computer terkecil memory (bit) yang disebut **ON** dan **OF**. **ON** diilustrasikan sebagai bohlam yang menyala dan **OF** diilustrasikan sebagai bohlam yang mati.

Contoh:

Karakter 'E' (huruf E) disimpan dalam memory dalam satu Byte, yang diilustrasikan dengan 8 buah bohlam lampu sebagai berikut:



Komposisi bit-bit ini adalah menurut standar **ASCII (American Standard Code for Information Interchange)** adalah suatu standar yang dibuat oleh Amerika (USA). Didalam dunia digital bit yang mengilustrasikan kondisi diatas adalah **ON** yang direpresentasikan dengan angka **1**, sedangkan **OFF** yang direpresentasikan dengan angka **0**.

Oleh karena itu karakter 'E' yang diilustrasikan dengan 8 buah lampu diatas , didalam computer disimpan dalam 1 Byte atau 8 bit memory dengan komposisi bit-bit sebagai berikut(table ASCII dapat dilihat di pertemuan 2):



Karakter 'E' nilai ASCIInya =

Ilustrasi diatas menggambarkan adanya bit-bit sebanyak 8 bit dengan nilai perbitnya dilihat dari sebelah kanan. Setiap bit yang bernilai 1 mempunyai nilai tersendiri tergantung diposisi mana bit itu berada. Dimulai dari kanan dengan angka 1 kemudian bergeser kekiri dengan kelipatan angka yang ada disebelah kanannya.

Berikut adalah satuan memory yang dapat digunakan untuk menyatakan daya tampung dimulai dari Byte sampai Yotta Byte didalam dunia digital yang selalu digunakan untuk menyatakan memory computer dan lainnya.

Byte (B)	
Kilo Byte (KB),	1 KB = 1024 B
Mega Byte (MB),	1 MB = 1024 KB
Giga Byte (GB),	1 GB = 1024 MB
Tera Byte (TB),	1 TB = 1024 GB
Peta Byte (PB),	1 PB = 1024 TB
Exa Byte (EB),	1 EB = 1024 PB
Zetta Byte (ZB),	1 ZB = 1024 EB
Yotta Byte (YB),	1 YB = 1024 ZB

Gambar 2. 1 Satuan Memory

Untuk pernyataan satuan kilogram dengan kilobyte itu berbeda dikarenakan kilogram menggunakan **Bilangan Desimal** yang digunakan sehari-hari oleh manusia, sedangkan kilobyte dipakai oleh komputer yang menggunakan Bilangan Biner.

3. Biner

Bilangan biner sering disebut juga dengan bilangan berbasis 2, untuk memahami konsep bilangan biner atau bilangan berbasis dua, sebelumnya akan kita pelajari terlebih dahulu konsep bilangan berbasis sepuluh atau bilangan desimal.

a. Bilangan decimal

Bilangan decimal sering disebut juga bilangan berbasis 10, bilangan decimal ini sering kita temui didalam kehidupan sehari-hari. Bilangan decimal disebut dengan bilangan berbasis sepuluh dikarenakan adanya sepuluh angka yang dapat digunakan mulai dari angka 0-9.

Contoh 1:

$$\begin{array}{l} \text{Angka 42} \quad \rightarrow (4 \times 10^1) + (2 \times 10^0) \\ \qquad \qquad \qquad 40 \quad + \quad 2 \quad = 42 \end{array}$$

Contoh 2:

$$\begin{array}{l} \text{Angka 2503} \quad \rightarrow (2 \times 10^3) + (5 \times 10^2) + (0 \times 10^1) + (3 \times 10^0) \\ \qquad \qquad \qquad 2000 \quad + 500 \quad + 0 \quad + 3 \quad = 2503 \end{array}$$

Contoh 3:

Angka **123456789** →

$$(1 \times 10^8) + (2 \times 10^7) + (3 \times 10^6) + (4 \times 10^5) + (5 \times 10^4) + (6 \times 10^3) + (7 \times 10^2) + (8 \times 10^1) + (9 \times 10^0)$$

$$100.000.000 + 20.000.000 + 3.000.000 + 400.000 + 50.000 + 6.000 + 700$$

$$+ 80 + 9$$

$$= \mathbf{123456789}$$

b. Bilangan biner

Bilangan biner disebut juga bilangan yang bebasis 2 karena hanya memiliki dua bilangan saja yaitu terdiri dari angka 0 dan 1.

Contoh 1:

Berapakah nilai decimal dari angka biner (0100 0110)?

Jawab:

$$0100\ 0110_2 = (0 \times 2^7) + (1 \times 2^6) + (0 \times 2^5) + (0 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)$$

$$0100\ 0110_2 = 0 + 64 + 0 + 0 + 0 + 4 + 2 + 0$$

$$0100\ 0110_2 = 72$$

Contoh 2:

Berapakah nilai decimal dari angka biner (0101 0110)?

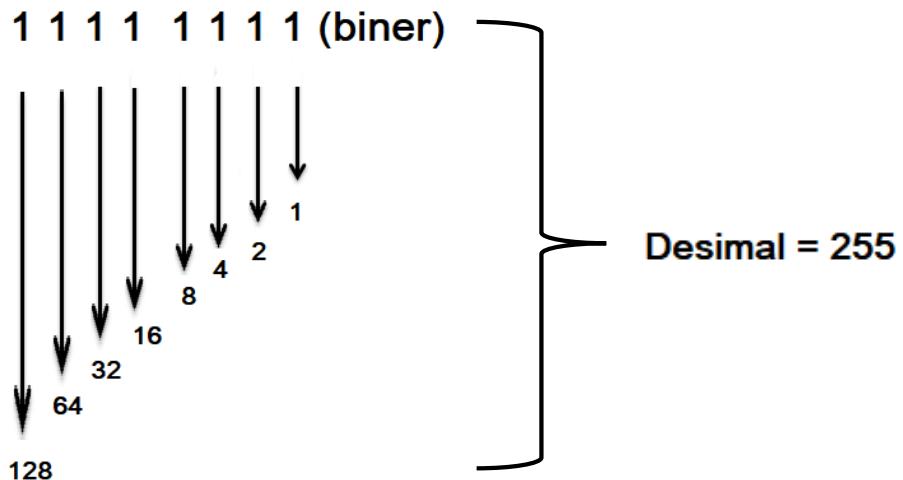
Jawab:

$$0101\ 0110_2 = (0 \times 2^7) + (1 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)$$

$$0101\ 0110_2 = 0 + 64 + 0 + 16 + 0 + 4 + 2 + 0$$

$$0101\ 0110_2 = 86$$

Ilustrasi Bilangan Biner



$$\text{Angka } 1 = 1 \times 2^0$$

$$\text{Angka } 2 = 1 \times 2^1$$

$$\text{Angka } 4 = 1 \times 2^2$$

$$\text{Angka } 8 = 1 \times 2^3$$

$$\text{Angka } 16 = 1 \times 2^4$$

$$\text{Angka } 32 = 1 \times 2^5$$

$$\text{Angka } 64 = 1 \times 2^6$$

$$\text{Angka } 128 = 1 \times 2^7$$

c. Operasi bilangan biner

1) Penjumlahan pada bilangan biner

Didalam bilangan biner, penjumlahan bilangan biner sama dengan penjumlahan biasa di dalam bilangan decimal, ketika mencapai limit bilangan hasilnya menjadi nol dengan menyisakan satu unyuk ditambahkan didepannya atau biasa disebut (*carry*). Berikut ini adalah table konversi bilangan decimal ke biner dari 1 sampai 128.

Tabel 2. 1 Konversi Bilangan Biner ke desimal

0000 0001	1	0010 0001	33	0100 0001	65	0110 0001	97
0000 0010	2	0010 0010	34	0100 0010	66	0110 0010	98
0000 0011	3	0010 0011	35	0100 0011	67	0110 0011	99
0000 0100	4	0010 0100	36	0100 0100	68	0110 0100	100
0000 0101	5	0010 0101	37	0100 0101	69	0110 0101	101
0000 0110	6	0010 0110	38	0100 0110	70	0110 0110	102
0000 0111	7	0010 0111	39	0100 0111	71	0110 0111	103
0000 1000	8	0010 1000	40	0100 1000	72	0110 1000	104
0000 1001	9	0010 1001	41	0100 1001	73	0110 1001	105
0000 1010	10	0010 1010	42	0100 1010	74	0110 1010	106
0000 1011	11	0010 1011	43	0100 1011	75	0110 1011	107
0000 1100	12	0010 1100	44	0100 1100	76	0110 1100	108
0000 1101	13	0010 1101	45	0100 1101	77	0110 1101	109
0000 1110	14	0010 1110	46	0100 1110	78	0110 1110	110
0000 1111	15	0010 1111	47	0100 1111	79	0110 1111	111
0001 0000	16	0011 0000	48	0101 0000	80	0111 0000	112
0001 0001	17	0011 0001	49	0101 0001	81	0111 0001	113
0001 0010	18	0011 0010	50	0101 0010	82	0111 0010	114
0001 0011	19	0011 0011	51	0101 0011	83	0111 0011	115
0001 0100	20	0011 0100	52	0101 0100	84	0111 0100	116
0001 0101	21	0011 0101	53	0101 0101	85	0111 0101	117
0001 0110	22	0011 0110	54	0101 0110	86	0111 0110	118
0001 0111	23	0011 0111	55	0101 0111	87	0111 0111	119
0001 1000	24	0011 1000	56	0101 1000	88	0111 1000	120
0001 1001	25	0011 1001	57	0101 1001	89	0111 1001	121
0001 1010	26	0011 1010	58	0101 1010	90	0111 1010	122
0001 1011	27	0011 1011	59	0101 1011	91	0111 1011	123
0001 1100	28	0011 1100	60	0101 1100	92	0111 1100	124
0001 1101	29	0011 1101	61	0101 1101	93	0111 1101	125
0001 1110	30	0011 1110	62	0101 1110	94	0111 1110	126
0001 1111	31	0011 1111	63	0101 1111	95	0111 1111	127
0010 0000	32	0100 0000	64	0110 0000	96	1000 000	128

Operasi penjumlahan bilangan biner ini pun mengikuti aturan yang dipakai untuk bilangan desimal, dikarenakan bilangan biner berbasis dua, maka angka-angka yang terlinat hanya ada dua yaitu 0 dan 1.

Empat kasus sederhana untuk mendapatkan aturan penambahan dalam bilangan biner yaitu:

- a) Angka nol jika ditambah dengan angka nol, Hasilnya adalah nol.
Dalam bilangan biner diwakilkan dengan $0 + 0 = 0$.
- b) Angka nol ditambah dengan angka 1 menghasilkan 1. Dalam bilangan biner diwakilkan dengan $0 + 1 = 1$.
- c) Angka 1 jika ditambah dengan angka nol, hasilnya adalah angka 1. Biner ini sama dengan $1 + 0 = 1$.
- d) Angka 1 jika ditambah dengan angka 1, Hasilnya adalah angka 2. Dalam bilangan biner diwakilkan dengan $1 + 1 = 10$.

Keempat kasus di atas dapat kita disimpulkan sebagai berikut:

Contoh 1: a. $0 + 0 = 0$

b. $0 + 1 = 1$

c. $1 + 0 = 1$

d. $1 + 1 = 10$ (angka satu akan disimpan untuk penjumlahan berikutnya).

$$\begin{array}{r} \text{a} \quad \begin{array}{r} 0 \\ 0 \\ \hline 0 \end{array} + \quad \text{b} \quad \begin{array}{r} 0 \\ 1 \\ \hline 1 \end{array} + \quad \text{c} \quad \begin{array}{r} 1 \\ 0 \\ \hline 1 \end{array} + \quad \text{d} \quad \begin{array}{r} 1 \\ 1 \\ \hline 10 \end{array} + \end{array}$$

Untuk penjumlahan bilangan biner yang lebih besar lagi, angka satu akan disimpan untuk penjumlahan didepannya seperti yang ada didalam penjumlahan bilangan desimal biasa.

Contoh 2:

1). 1011 (11)

2). 0111 (7)

1100 (12)

1011 (11)

----- +

----- +

10111 (23)

10010 (18)

Sama halnya didalam penjumlahan decimal ketika terdapat penjumlahan antara $9 + 1 = 10$, didalam bilangan biner pun sama $1+1=10$. Itulah yang membedakan antara bilangan decimal yang berbasis **sepuluh** dengan bilangan biner yang berbasis **dua**, dimana ketika penjumlaha mencapai limit bilangan, maka hasilnya akan kembali ke angka nol dengan kelebihan satu yang bias ditambahkan ke angka yang ada didepannya.

Contoh 3: $10 + 13 = 23$ (decimal)

$$\begin{array}{r}
 1010 \\
 1101 \\
 \hline
 10111
 \end{array}
 \xrightarrow{\quad + \quad}
 \begin{array}{r}
 10 \\
 13 \\
 \hline
 23
 \end{array}$$

↓ ↓ ↓ ↓ \\
 16 4 2 1

2) Komplemen 1 dan 2

Dalam komputer dengan bahasa digital terdapat didalamnya dua cara untuk menyatakan nilai negatif, yaitu dengan cara mencari komplemen satu dan juga komplemen dua.

Komplemen satu adalah suatu sistem penomoran untuk menyatakan nilai-nilai atau negatif yang ada didalam beberapa jenis komputer. Dalam kasus ini aturan yang ada bahwa nilai nol akan diwakilkan dengan dua buah nilai, yaitu -0 (negatif nol) dan +0 (positif nol).

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 = +3$$

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 0 = +2$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 = +1$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 = +0$$

$$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 = -0$$

$$1\ 1\ 1\ 1\ 1\ 1\ 1\ 0 = -1$$

$$1\ 1\ 1\ 1\ 1\ 1\ 0\ 1 = -2$$

$$1\ 1\ 1\ 1\ 1\ 1\ 0\ 0 = -3$$

Dari aturan ini, nilai +0 (nol positif) berpasangan dengan nilai -0 (nol negatif), +1 (satu positif) berpasangan dengan -1 (satu negatif), dan seterusnya. Hal ini memperlihatkan bahwa negasi dari nilai 0 adalah nilai -0, negasi dari nilai 1 adalah nilai -1, begitu seterusnya.

Ada beberapa kelemahan didalam aturan diatas, yaitu terdapat nilai yang kurang sesuai sehingga diciptakannya aturan baru yaitu aturan yang ke dua yang disebut komplemen dua.

Aturan komplemen dua tidak jauh berbeda dengan aturan komplemen satu, akan tetapi dalam prosesnya negasi semua bit akan dibalik, sehingga tidak ada lagi kebigungan didalam merepresentasikan nilai +0 (nol positif) dengan -0(nol negative), karena hanya ada satu nilai 0 (nol), seperti berikut:

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 = +3$$

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 0 = +2$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 = +1$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 = 0$$

$$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 = -1$$

$$1\ 1\ 1\ 1\ 1\ 1\ 1\ 0 = -2$$

$$1\ 1\ 1\ 1\ 1\ 1\ 0\ 1 = -3$$

$$1\ 1\ 1\ 1\ 1\ 1\ 0\ 0 = -4$$

Aturan diatas menunjukan bahwa nilai 0 (nol) akan berpasangan dengan nilai -1 (satu negative), nilai +1 (satu positif) akan berpasangan dengan -2(dua negative), dan seterusnya. Hal ini menegaskan bahwa negasi dari 0 adalah -1, negasi dari +1 adalah -2, dan begitu seterusnya.

Komplemen 1 dan 2 dalam bilangan biner penting karena dapat digunakan untuk menyatakan bilangan negatif. Aritmetika komplemen 2 pada umumnya digunakan dalam komputer untuk mendukung bilangan negatif.

Komplemen 1 dari bilangan biner dapat diperoleh dengan merubah seluruh angka 1 dengan nol dan sebaliknya.

Adapun contoh komplemen 1 adalah sebagai berikut :

- a. $1010 \rightarrow$ komplemen 1 $\rightarrow 0101$
- b. $1101 \rightarrow$ komplemen 1 $\rightarrow 0010$
- c. $0001 \rightarrow$ komplemen 1 $\rightarrow 1110$
- d. $0111 \rightarrow$ komplemen 1 $\rightarrow 1000$

$1011 \rightarrow$ Bilangan biner

$0100 \rightarrow$ Komplemen 1

Komplemen 1 adalah **negasi** dari bilangan biner asal.

Kemudian pengertian komplemen 2 adalah bilangan biner yang terjadi jika ditambahkan 1 terhadap komplemen 1, yaitu :

Komplemen 2 = (komplemen 1) + 1

Jadi komplemen 2 dari 1011011 adalah

1011011

$0100100 \rightarrow$ Komplemen 1

0100100

$\begin{array}{r} 1 \\ \hline + \end{array}$

$0100101 \rightarrow$ Komplemen 2

Contoh lain komplemen dari suatu bilangan biner :

- a) Komplemen 2 dari 1100 adalah $0011 + 1 = 0100$
- b) Komplemen 2 dari 1011 adalah $0100 + 1 = 0101$
- c) Komplemen 2 dari 0101 adalah $1010 + 1 = 1011$
- d) Komplemen 2 dari 110010 adalah $001101 + 1 = 001110$

Cara lain untuk mendapatkan komplemen 2 adalah dengan cara mencari negasi dari bilangan biner tersebut kemudian menambahkan angka 1 diakhir bilangan biner.

C. Latihan tugas

1. carilah bilangan biner dari bilangan decimal berikut!

a. 39

b. 50

c. 79

d. 100

2. diketahui bilangan biner:

$A = 1100\ 0011$

$B = 1010\ 0101$

$C = 1011\ 0100$

$D = 0101\ 1110$

Berapakah jika:

a. $A + B$

b. $B + C$

c. $C + D$

d. $A + C$

e. $A + D$

3. Carilah komplemen 2 dari data biner berikut?

a. 0000 1011

b. 0101 0101

c. 0100 0110

d. 0011 0011

e. 0011 1000

4. Carirai bilangan biner dari penjumlahan bilangan decimal berikut:

a. $120 + 10$

b. $87 + 57$

c. $115 + 125$

d. $125 + 225$

e. $256 + 123$

D. Referensi

- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 3

TIPE DATA, VARIABEL DAN KONSTANTA

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu:

1. Membedakan jenis-jenis tipe data dasar dalam pemrograman
2. Menggunakan jenis-jenis tipe data dasar dalam pemrograman
3. Memahami penggunaan variabel dan konstanta dalam pemrograman
4. Mendeklarasikan variabel menggunakan jenis-jenis tipe data dasar.

B. Uraian Materi

Sebuah variabel akan mengalokasikan tempat penyimpanan yang mempunyai nama dalam memori komputer. Setiap variabel akan memiliki tipe tertentu yang akan menentukan ukuran dan letak memori, rentang nilai yang dapat disimpan dan operasi yang dapat diterapkan ke variabel tersebut.

1. Variabel

a. Definisi

Salah satu hal yang paling dasar dalam pemrograman adalah variabel, jika diilustrasikan sebuah variabel seperti sebuah kotak kecil, dimana kita dapat menyimpan barang-barang dalam kotak tersebut untuk digunakan nanti. Konsep variabel diambil dari matematika seperti :

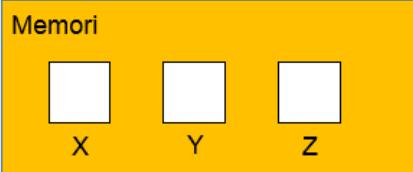
$x = 1 \rightarrow$ menyimpan nilai 1 dalam variabel x

Dalam pemrograman variabel adalah lokasi atau area atau tempat di dalam memori yang dapat menyimpan data sementara dalam suatu program, dan data tersebut dapat diubah, disimpan atau ditampilkan kapanpun dibutuhkan.

Setiap variabel harus diberi nama, dan nama variabel harus berbeda antara satu variabel dengan variabel yang lain. Masing-masing variabel memiliki alamat sendiri didalam memori komputer, kita cukup menyebutkan nama variabel dimana data disimpan, maka komputer akan dapat menemukan alamat variabel tempat data tersebut tersimpan pada memori.

Contoh :

Misal terdapat 3 buah variabel yang di beri nama **X**, **Y** dan **Z**, jika dilustraskan didalam memori komputer digambarkan sebagai berikut :



Terlihat dari ilustrasi gambar diatas, variabel **X**, **Y** dan **Z** akan menempati lokasi atau area tertentu didalam memori komputer. Dimana variabel tersebut digunakan untuk menampung nilai yang akan digunakan dalam program.

b. Pemberian Nama Variabel

Pemberian nama variabel ditentukan oleh pembuat program sendiri, namun dalam pemberian nama variabel terdapat syarat-syarat seperti berikut :

- 1) Nama variabel tidak boleh sama dengan nama *keyword* dan *function*.
- 2) Nama variabel maksimum 32 karakter.
- 3) Nama variabel harus diawali dengan huruf atau garis bawah (*underscore _*), karakter berikutnya boleh angka, huruf atau garis bawah
- 4) Nama variabel tidak boleh ada spasi.

Tabel 3. 1 Keyword Dalam ANSI Bahasa C++/C

auto	float	friend	inline
break	for	delete	new
case	go to	class	operator
char	if	asm	private
const	int	while	protected
continue	long	volatile	public
default	register	void	template
do	return	unsigned	this
double	short	union	virtual
else	signed	typedef	struct
enum	size of	switch	static
extern			

c. Contoh pemberian nama variabel

Berikut adalah beberapa contoh pemberian nama variabel yang benar, dan contoh pemberian nama variabel yang salah:

Tabel 3. 2 Contoh Penamaan Variabel

Penamaan Yang benar	Penamaan Yang salah	Keterangan
X	1X	Awalnya bukan huruf atau garis bawah
X1		
luas	luas-1	Mengandung tanda minus
LUAS	Keliling Lingkaran	(-)
KelilingLingkaran	benar/salah	Mengandung spasi
Keliling_Lingkaran	float	Mengandung spesial karakter
KL	switch	
_panjang	keliling-lingkaran	Sama dengan keyword
FLOAT		Sama dengan keyword
		Mengandung tanda minus (-)



Dalam bahasa C++/C penamaan variabel berbeda antara huruf besar dan huruf kecil (*case sensitif*), variabel **nilai** berbeda dengan **NILAI** berbeda dengan **Nilai**, **FLOAT** berbeda dengan **float** yang merupakan *keyword*.

2. Konstanta

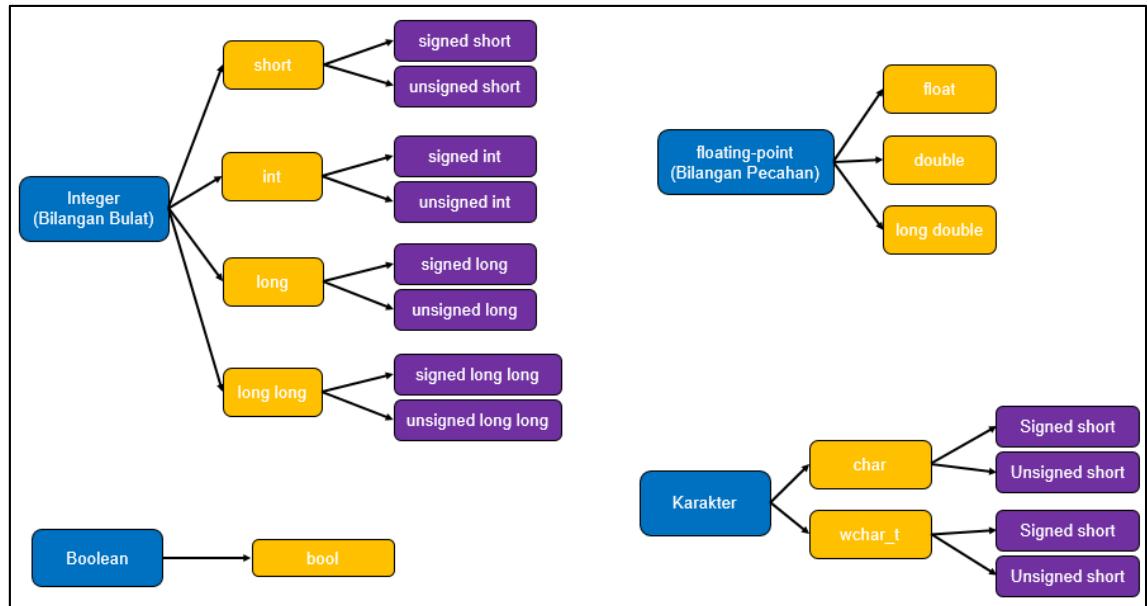
Konstanta merupakan nilai numerik/angka atau karakter yang tetap. Seperti nilai **PI** yaitu 22/7 atau 3.14159 merupakan nilai konstanta, nilai yang tidak dapat diubah atau nilainya tetap. Untuk mendeklarasikan konstanta menggunakan *keyword* **const**.

```
const double PI = 3.14159;
```

Dengan menggunakan *keyword* **const**, nilai dari varibel **PI** tidak bisa diubah setelah dideklarasikan.

3. Tipe Data

Tipe data merupakan jenis-jenis data yang dikategorikan berdasarkan sifat dan jenisnya, gambar berikut memperlihatkan kategori tipe data dasar :



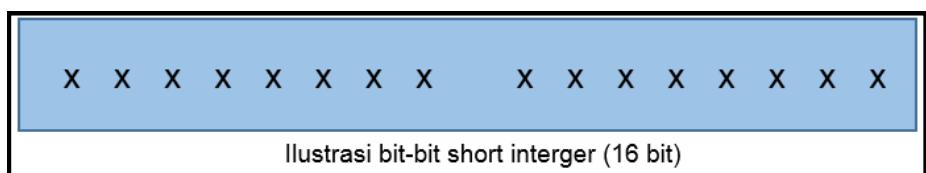
Gambar 3. 1 Kategori Tipe Data Dasar

a. Tipe Data *numeric integer* (bilangan bulat)

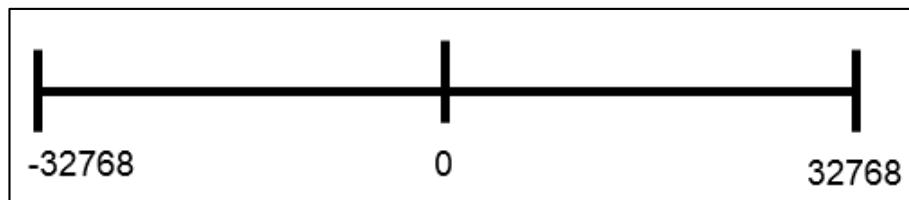
Integer adalah jenis tipe data untuk menyimpan nilai numerik/angka dalam bentuk bilangan bulat, yaitu bilangan yang tidak mengandung angka dibelakang koma. Sebagai contoh 4, -17, 50 dan 100 adalah bilangan bulat. Bilangan bulat bisa positif(+) , negatif(-) atau 0, tipe data bilangan bulat terbagi menjadi :

1) Short

Tipe data short dalam memori komputer menempati area 2 byte (16bit), dengan ilustrasi sebagai berikut :

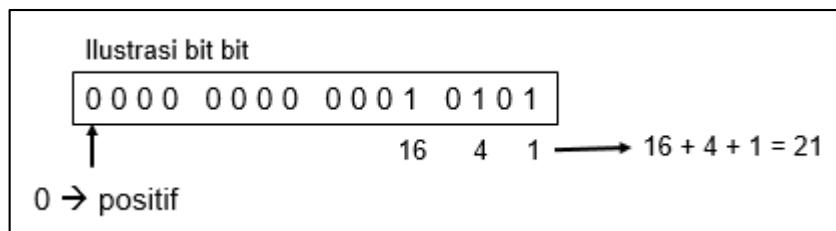


Sedangkan nilai yang dapat ditampung tipe data **short** berkisar -32768 sampai dengan 32768 dengan ilustrasi sebagai berikut :



Gambar 3. 2. Jangkauan Nilai Tipe data short

Tipe data short hanya dapat menampung nilai numerik/angka dengan maksimum nilai 32768, sehingga jika ingin menampung nilai yang lebih besar dari 32768 harus menggunakan tipe data yang lain, tipe data yang dapat menampung nilai yang lebih besar.



Beberapa contoh ilustrasi bit-bit serta nilai yang tersimpan dalam 2 byte tipe data **short**.

0000 0000 0000 0000 =
0

0000 0000 0000 0001 =
1

0000 0000 0000 0011 =
3

0000 0000 0000 1000 =
8

0111 1111 1111 1111 =
32767

1000 0000 0000 0000 = -
32768

1000 0000 0000 0001 = -
32767

1111 1111 1111 1111 = -1



Penulisan bit-bit yang dikelompokkan menjadi 4 bit dengan tujuan untuk mempermudah pembacaan bit-bit tersebut.

1 | 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 = -32768
 ↑
 1 → negatif

Nilai ini didapat dari bentuk
Two's complement (komplemen 2)
Ditulis 2's Complement

Cara memperoleh nilai 2's Complement

data	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
One's complement	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Ditambah 1	1 +
2's complement	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

One's complement diperoleh dari membalikan nilai bit-bit pada data awal, bit 0 menjadi bit 1 dan sebaliknya.

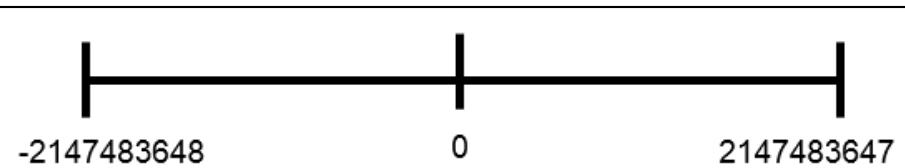
Pada 2's complement jika dihitung bit-bit yang on, nilainya adalah 32768.

2) Int

Tipe data **int** dalam memori komputer menempati area 4 byte (32bit), dengan ilustrasi sebagai berikut :

xxxx
Ilustrasi bit-bit interger (32 bit)

Nilai yang dapat ditampung dalam tipe data **int** berkisar -2147483648 sampai dengan 2147483647

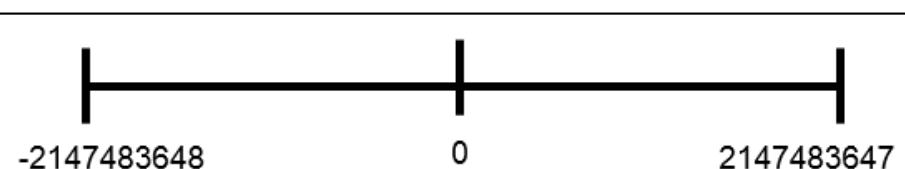


Gambar 3. 3 Jangkauan Nilai Tipe data int

3) Long

Tipe data **long** dalam memori komputer menempati area 4 byte (32bit) sama seperti tipe data **int**, jangkauan nilai yang dapat ditampung juga sama seperti tipe data **int**.

xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx
Ilustrasi bit-bit interger (32 bit)



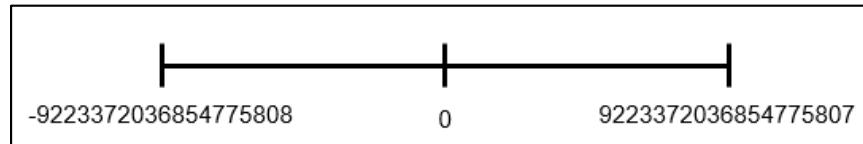
Gambar 3. 4 Jangkauan Nilai Tipe data long

4) long long

Tipe data **long long** dalam memori komputer menempati area 8 byte (64bit), dengan ilustrasi sebagai berikut :

xxxx
Ilustrasi bit-bit long long integer (64bit)

Nilai yang dapat ditampung antara -9223372036854775808 sampai dengan 9223372036854775807.



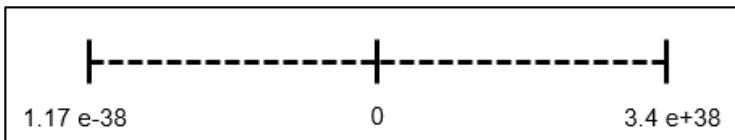
Gambar 3. 5. Jangkauan Nilai Tipe data long long

b. Tipe Data Floating-Point (Pecahan)

Banyak perhitungan yang dilakukan oleh komputer membutuhkan angka yang memiliki bagian fraksional atau angka yang memiliki nilai dibelakang koma (bilangan desimal). Misalnya, ketika menghitung luas lingkaran perlu melibatkan nilai PI dengan nilai berkisar 3.14159. bahasa pemrograman C++ mendukung angka-angka non-integer(bukan bilangan bulat) seperti nilai PI, bilangan non-integer disebut dengan *floating-point*.

1) Float (*floating-point single precision*)

kata kunci atau keyword yang digunakan untuk mendeklarasikan *floating-point single precision* adalah **float**. Tipe data **float** disimpan dalam area 4 byte (32bit) dan memiliki ketelitian sampai dengan 6 *digit* dibelakang koma, dengan jangkauan nilai ditampilkan pada gambar berikut :

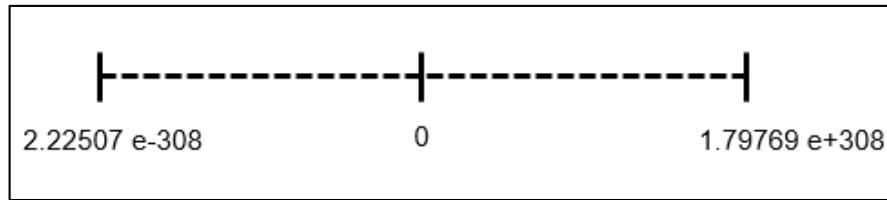


1.35 e-02 artinya $\rightarrow 1.35 \times 10^{-2} = 1.35 \times 0.01 = 0.0135$

1.35 e+02 artinya $\rightarrow 1.35 \times 10^2 = 1.35 \times 100 = 135$

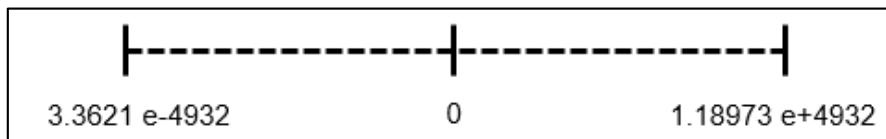
2) Double (*floating-point double precision*)

kata kunci atau *keyword* yang digunakan untuk mendeklarasikan *floating-point double precision* adalah **double**. Tipe data double disimpan dalam area 8 byte (64bit) dan memiliki ketelitian hingga 15 digit, dengan jangkauan nilai ditampilkan pada gambar berikut :



3) Long double (*floating-point double precision*)

kata kunci atau *keyword* yang digunakan untuk mendeklarasikan *floating-point double precision* adalah **long double**. Tipe data **long double** disimpan dalam area 12 byte (96bit), dengan jangkauan nilai ditampilkan pada gambar berikut :



c. Tipe Data Karakter

Jenis tipe data **karakter** digunakan untuk menyimpan kode karakter. Kode karakter merupakan suatu bilangan bulat yang terkait karakter yang diwakilkan. Misalnya, huruf **A** diwakili oleh kode 65 dalam kode **ASCII** (*American Standard Code for Information Interchange*).

ASCII table															
Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex
(nul)	0	0000	0x00	(sp)	32	0040	0x20	@	64	0100	0x40	`	96	0140	0x60
(soh)	1	0001	0x01	!	33	0041	0x21	A	65	0101	0x41	a	97	0141	0x61
(stx)	2	0002	0x02	"	34	0042	0x22	B	66	0102	0x42	b	98	0142	0x62
(etx)	3	0003	0x03	#	35	0043	0x23	C	67	0103	0x43	c	99	0143	0x63
(eot)	4	0004	0x04	\$	36	0044	0x24	D	68	0104	0x44	d	100	0144	0x64
(eng)	5	0005	0x05	%	37	0045	0x25	E	69	0105	0x45	e	101	0145	0x65
(ack)	6	0006	0x06	&	38	0046	0x26	F	70	0106	0x46	f	102	0146	0x66
(bel)	7	0007	0x07	'	39	0047	0x27	G	71	0107	0x47	g	103	0147	0x67
(bs)	8	0010	0x08	(40	0050	0x28	H	72	0110	0x48	h	104	0150	0x68
(ht)	9	0011	0x09)	41	0051	0x29	I	73	0111	0x49	i	105	0151	0x69
(nl)	10	0012	0x0a	*	42	0052	0x2a	J	74	0112	0x4a	j	106	0152	0x6a
(vt)	11	0013	0x0b	+	43	0053	0x2b	K	75	0113	0x4b	k	107	0153	0x6b
(np)	12	0014	0x0c	,	44	0054	0x2c	L	76	0114	0x4c	l	108	0154	0x6c
(cr)	13	0015	0x0d	-	45	0055	0x2d	M	77	0115	0x4d	m	109	0155	0x6d
(so)	14	0016	0x0e	.	46	0056	0x2e	N	78	0116	0x4e	n	110	0156	0x6e
(si)	15	0017	0x0f	/	47	0057	0x2f	O	79	0117	0x4f	o	111	0157	0x6f
(dle)	16	0020	0x10	0	48	0060	0x30	P	80	0120	0x50	p	112	0160	0x70
(dc1)	17	0021	0x11	1	49	0061	0x31	Q	81	0121	0x51	q	113	0161	0x71
(dc2)	18	0022	0x12	2	50	0062	0x32	R	82	0122	0x52	r	114	0162	0x72
(dc3)	19	0023	0x13	3	51	0063	0x33	S	83	0123	0x53	s	115	0163	0x73
(dc4)	20	0024	0x14	4	52	0064	0x34	T	84	0124	0x54	t	116	0164	0x74
(nak)	21	0025	0x15	5	53	0065	0x35	U	85	0125	0x55	u	117	0165	0x75
(syn)	22	0026	0x16	6	54	0066	0x36	V	86	0126	0x56	v	118	0166	0x76
(etb)	23	0027	0x17	7	55	0067	0x37	W	87	0127	0x57	w	119	0167	0x77
(can)	24	0030	0x18	8	56	0070	0x38	X	88	0130	0x58	x	120	0170	0x78
(em)	25	0031	0x19	9	57	0071	0x39	Y	89	0131	0x59	y	121	0171	0x79
(sub)	26	0032	0x1a	:	58	0072	0x3a	Z	90	0132	0x5a	z	122	0172	0x7a
(esc)	27	0033	0x1b	;	59	0073	0x3b	[91	0133	0x5b	{	123	0173	0x7b
(fs)	28	0034	0x1c	<	60	0074	0x3c	\	92	0134	0x5c		124	0174	0x7c
(gs)	29	0035	0x1d	=	61	0075	0x3d]	93	0135	0x5d	}	125	0175	0x7d
(rs)	30	0036	0x1e	>	62	0076	0x3e	^	94	0136	0x5e	~	126	0176	0x7e
(us)	31	0037	0x1f	?	63	0077	0x3f	_	95	0137	0x5f	(del)	127	0177	0x7f

Gambar 3. 6. Kode ASCII

Tipe data karakter terbagi menjadi 2, yaitu **char** dan **wchar_t (wide character type)** :

1) Char

Tipe data **char** digunakan untuk menyatakan karakter tunggal dalam 1 byte (8 bit) seperti huruf alfabet (baik huruf besar maupun huruf kecil), angka, tanda baca, dan karakter kontrol (seperti *new line* dan *tab*). Dalam bahasa pemrograman c++ **karakter** diapit oleh kutip tunggal('), seperti:

‘A’

2) wchar_t

Tipe data **wchar_t (wide character type)** terdiri dari setidaknya 2 byte (16 bit) dan dengan demikian mampu menyimpan karakter **Unicode modern**. Unicode adalah kode 16-bit yang juga digunakan dalam Windows NT dan berisi kode untuk sekitar 35.000

karakter dalam 24 bahasa. Tipe data **wchar_t** biasa digunakan untuk penggunaan karakter yang besar seperti bahasa china.

d. Tipe Data Boolean

Hasil perbandingan operator logika **AND** atau **OR** akan menghasilkan nilai **boolean** yaitu **benar** (*true*) atau **salah** (*false*). Dalam bahasa pemrograman C++ tipe **boolean** digunakan untuk merepresentasikan nilai **boolean**. Ekspresi nilai boolean akan direpresentasikan sebagai angka 1 jika bernilai **benar** (*true*), dan angka 0 jika bernilai **salah** (*false*).

```
bool benar = true → 1 (satu)
bool salah = false → 0 (nol)
```

e. Modifikasi Tipe Data

Terdapat beberapa tipe data modifikasi yang dapat dikenakan terhadap tipe data dasar, seperti **signed** dan **unsigned**. Untuk lebih jelasnya ditunjukkan pada **Tabel 3.3**.

f. Rangkuman Tipe Data

Tipe data yang digunakan dalam bahasa pemrograman C++ ditampilkan pada tabel berikut :

Tabel 3.3. Tipe Data Pada C++

Tipe data	Penulisan dengan C++	Jumlah byte	Jangkauan nilai yang dapat ditampung
Character	char atau signed char	1	-128 sampai dengan 127
	unsigned char	1	0 sampai dengan 255
	wchar_t	2	
Integer	signed short atau short	2	-32768 sampai dengan 32768
	Unsigned short	2	0 sampai dengan 65535
	int atau signed int	4	-2147483648 sampai dengan 2147483647
	unsigned int	4	0 sampai dengan 4294967295
	long atau signed long atau signed long int	4	-2147483648 sampai dengan 2147483647

Tipe data	Penulisan dengan C++	Jumlah byte	Jangkauan nilai yang dapat ditampung
	unsigned long atau unsigned long int	4	0 s.d 4294967295
	long long int	8	-9223372036854775808 s.d 9223372036854775807
	unsigned long long int	8	0 sampai dengan 1844644073709551615
Floating point single precision	Float	4	1.17e-38 sampai dengan 3.4e+38
Floating point double precision	double	8	2.2e-308 sampai dengan 1.7e+308
	long double	12	3.4e-4932 sampai dengan 1.18e+4932

g. Deklarasi Variabel dengan Tipe Data

Sebelum variabel digunakan dalam program, variabel harus dideklarasikan terlebih dahulu agar program dapat mengenali variable yang akan digunakan dalam program. Deklarasi variabel tidak terlepas dari tipe data yang digunakan, cara mendeklarasikan variabel seperti berikut:

tipedata<spasi>namavariabel
Contoh : int nilai long long int _investasi float rata2 double keliling unsigned int nilai char A

deklarasi variabel dapat dilakukan dengan beberapa variabel yang dipisahkan dengan koma, seperti berikut:

tipedata<spasi>namavariabel1, namavariabel2,..., namavariabeln

Contoh :

```

int nilai1, nilai2
long long int _investasi, _HS
float rata2, nilai_akhir
double keliling, luas
unsigned int nilai1, nilai2, nilai3
char A, B, C

```

Contoh-Contoh penggunaan Variabel dan Tipe data dengan bahasa pemrograman C++

Listing 3.1 : variabel_01.cpp

```

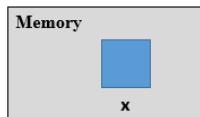
#include<iostream>
using namespace std;
main()
{
    int x; //deklarasi variabel x dengan tipe data int
    x = 15; //mengisi nilai variabel x dengan nilai 5
    cout<<x; //mencetak isi variabel x
}

```

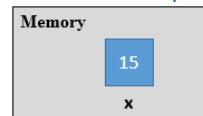
Output : 15



int x; → artinya mendeklarasikan atau menyiapkan sebuah variabel dengan nama x bertipe data integer(bilangan bulat).



x = 15; → artinya variabel yang telah dideklarasikan atau disiapkan diisi dengan nilai 15.



cout → dibaca si-out keyword yang berfungsi untuk mencetak.

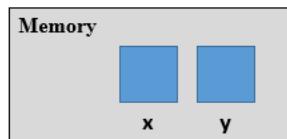
cout<<x; → artinya mencetak nilai yang tersimpan dalam variabel x yaitu 15.

Listing 3.2 : variabel02.cpp

```
#include<iostream>
using namespace std;
main()
{
    int x, y; //deklarasi variabel x dan y dengan tipe
               //data int
    x = 15; //mengisi nilai variabel x dengan nilai 15
    y = 50; //mengisi nilai variabel y dengan nilai 50
    cout<<x; //mencetak isi variabel x
    cout<<y; //mencetak isi variabel y
}
```

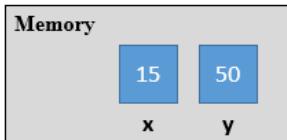
Output : 15 50

`int x, y;` → artinya mendeklarasikan atau menyiapkan dua buah variabel dengan nama **x** dan **y** bertipe data integer(bilangan bulat).



`x = 15;` → artinya variabel yang telah dideklarasikan atau disiapkan di isi dengan nilai 15

`y = 50;` → artinya variabel yang telah dideklarasikan atau disiapkan di isi dengan nilai 50



`cout<<x;` → artinya mencetak nilai yang tersimpan dalam variabel x yaitu 15.

`cout<<y;` → artinya mencetak nilai yang tersimpan dalam variabel y yaitu 50.

Listing 3.3 : variabel03.cpp

```
#include<iostream>
using namespace std;
main()
{
    const double phi = 3.14; //deklarasi variabel phi
                           //dengan tipe data int dan konstanta
    cout<<phi; //mencetak isi variabel x
}
```

Output : 3.14

`const double phi = 3.14;` → deklarasi variabel x dengan tipe data double dengan dengan keyword `const`
`const` → dengan menggunakan keyword `const` artinya nilai yang ada pada variabel phi tidak bisa di ubah
`cout<<phi;` → mencetak isi variabel phi

Listing 3.4 : variabel04.cpp

```
#include<iostream>
using namespace std;
main()
{
    int nilai_uts, nilai_uas, rata_rata; //deklarasi variabel
    nilai_uts = 80; //mengisi nilai variabel
    nilai_uts = 70; //mengisi nilai variabel
    rata_rata = (nilai_uts+nilai_uas)/2;
    cout<<rata_rata; //mencetak isi variabel x
}
Output : 75
```

Listing 3.5 : variabel05.cpp

```
#include<iostream>
using namespace std;
main()
{
    float f; //deklarasi variabel f dengan tipe data float
    f = 1.5; //mengisi nilai variabel f dengan nilai 1.5
    cout<<f; //mencetak isi variabel f
}
Output : 15
```

Listing 3.5 : variabel06.cpp

```
#include<iostream>
using namespace std;
main()
{
    float f, g, h; //deklarasi variabel f dengan tipe data float
    f = 7; //mengisi nilai variabel f dengan nilai 1.5
    f = 3; //mengisi nilai variabel f dengan nilai 1.5
    h = 7/3;
    cout<<h; //mencetak isi variabel h
}
Output : 2.33333
```

C. Soal Latihan / Tugas

1. Sebutkan jenis-jenis tipe data dasar dalam bahasa pemrograman c++ ?
2. berikan tipe data yang tepat dari nilai-nilai berikut :
 - a. 2000
 - b. 1.5
 - c. -15000000

- d. 999999999
3. Deklarasikan sebuah variabel yang beri nama **nilai akhir** kemudian isikan suatu nilai ke dalam variabel tersebut ?
4. Berapah nilai minimum dan maksimum yang dapat di tampung dalam tipe-tipe data berikut :
 - a. short
 - b. int
 - c. long double
 - d. char
5. Manakan nama-nama variabel yang benar dan nama variabel yang salah dari variabel-variabel berikut, dan berikan penjelasan.
 - a. 1Nilai
 - b. _kelas
 - c. long
 - d. CONST
 - e. \$dolar

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*.
BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*.
Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*.
Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 4

ASSIGNMENT STATEMENT, ARITMETIC EXPRESSION DAN OPERATOR

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu menuliskan dan membuat Assignment Statement, Aritmetic Expression dan Operator dalam pemrograman.

B. Uraian Materi

Dalam pertemuan ke – 3 telah jelaskan tentang memberikan nama terhadap suatu variabel, pertanyaan selanjutnya adalah: bagaimana memasukkan data ke dalam variabel-variabel tersebut?

Di C++ untuk memasukkan data ke variabel terdapat dua cara:

1. Menggunakan Assignment Statement
2. Menggunakan input (read) statement

1. Assignment Statement

Assignment statement atau biasa di terjemahkan menjadi **pernyataan penugasan**, yang di maksud di sini adalah mengisi sebuah variabel dengan suatu nilai. Format penulisan atau penggunaan assignment statement adalah sebagai berikut:

variabel = ekspresi;

Dalam **assignment statement / pernyataan penugasan** nilai ekspresi harus cocok atau sesuai dengan tipe data variabel. Ekspresi di sisi kanan akan dievaluasi dan nilainya akan diberikan ke dalam variabel yang ada pada sisi kiri.

Dalam bahasa C++ → **=** disebut dengan **operator penugasan**.

Contoh 1:

**int X;
A = 8;**

X = 8 → artinya mengisi variabel **X** dengan nilai 8 (**X** diisi 8)
Atau nilai **X** dibuat sama dengan 8.

Ilustrasi digambarkan sebagai berikut:



Sebelum intruksi **X=8**; dilaksanakan, variabel **X** sebenarnya sudah ada isinya, tetapi kita tidak mengetahui isinya.

X

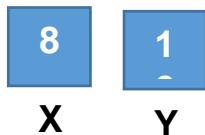


Setelah intruksi **X = 8**; dilaksanakan maka isi dari variabel **X** menjadi 8.

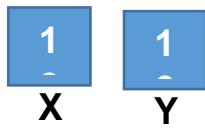
X

Contoh 2:

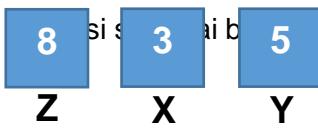
X = Y	Mengisi variabel X dengan isi dari variabel Y . Dimana isi variabel Y tidak berubah.
int X = 8, Y = 10; X = Y; cout<< X; cout<< Y;	Tercetak: 10 10
 Ilustrasi program di atas: sebelum intruksi X = Y dilaksanakan, misal X = 8, Y = 10 .	setelah intruksi X = Y dijalankan, maka isi dari variabel X menjadi 10 , dan isi dari variabel Y tetap 10 (tidak berubah).



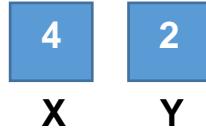
setelah intruksi **X = Y** dijalankan, maka isi dari variabel **X** menjadi **10**, dan isi dari variabel **Y** tetap **10** (tidak berubah).



Contoh 3:

$Z = X + Y;$	Mengisi variabel Z dengan hasil penjumlahan dari variabel X dan Y .
<pre>#include<iostream> main(){ int Z=6, X=3, Y=5; Z = X + Y; }</pre>	<p>Misal nilai variabel Z, X dan Y sebelum intruksi dilaksanakan digambarkan sebagai berikut:</p>  <p>Setelah intruksi Z = X + Y dilaksanakan, nilai Z berubah dengan</p> 
Tercetak: 8	
	<p>Proses pelaksanaan intruksi Z = X+Y dalam komputer:</p> <p>Pertama nilai X dan Y di bawa ke processor (CPU) kemudian dilakukan penambahan nilai X dan Y di dalam CPU. Hasil dari penjumlahan tersebut di isikan ke dalam variabel Z.</p> <p>Isi variabel Z menjadi 8,</p> <p>Sedangkan isi variabel X dan Y tidak berubah.</p>

Contoh 4:

X = 4; Y = 2; X = X + Y;	Mengisi variabel X dengan hasil penjumlahan nilai X dan Y . (X di isi dengan nilai X + Y) Sebelum intruksi X = X + Y di laksanakan, nilai X dan Y digambarkan sebagai berikut: 
Tercetak: 7	 <p>Setelah intruksi X = X + Y dilaksanakan, nilai dari variabel X berubah menjadi 6, dan ini dari variabel Y tetap 2 (tidak berubah).</p>

Contoh 5:

X = 6;	Mengisi variabel X dengan hasil penjumlahan nilai A dengan konstanta 1.
X = X + 1;	(variabel X diisi dengan nilai A+1;)

Contoh – Contoh Assignment Statement dasar menggunakan bahasa C++:

1)

```
#include<iostream>
using namespace std;
main()
{ int X = 7;
cout<<X+2;
cout<<"\n"<<X;
}
```

Tercetak : **9**
7

2)

```
#include<iostream>
using namespace std;
main()
{ int X = 7;
cout<<int(X =
X+2);
cout<<X;
}
```

Tercetak: **9**
9

3)

```
#include<iostream>
using namespace std;
main()
{ int X = 7, Y = 5;
X = Y;
cout<<X;
cout<<"\n"<<Y;
}
```

Tercetak: **5**
5

4)

```
#include<iostream>
using namespace std;
main()
{ int X = 7, Y = 5;
X = Y;
Y = X;
cout<<X;
cout<<"\n"<<Y;
}
```

Tercetak: **5**
5

2. Arithmetic Expression dalam Assignment Statement

Perhatikan assignment statement berikut ini:

$$X = A + B * (C - D) ;$$

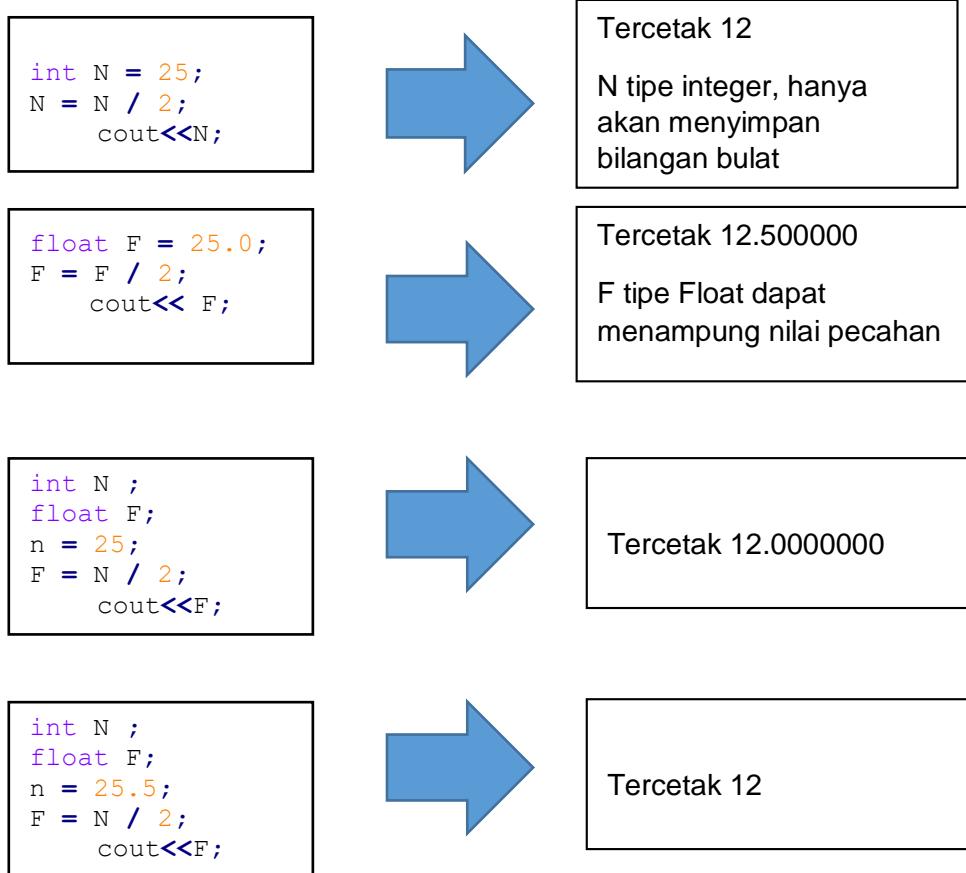
Disebut aritmetic expression atau pernyataan aritmatik dimana A,B,C dan D disebut operand (yang dioprasikan). Sedngkan +, *, -, dan () disebut aritmetic operator(operator aritmatika).

a. Operator Aritmatika

Dalam bahasa C++ dan Java, operator aritmatika yang digunakan adalah sebagai berikut:

Operator	Tingkat Hierarki atau Power
• / %	Kali, Bagi dan Modulus sama kuatnya Kali, bagi dan modulus Lebih didahulukan dari Tambah dan Kurang
+ -	Tambah dan kurang sama derajatnya
()	Yang ada di dalam () adalah satu kesatuan nilai

a. Pembagian.



b. Modulus(Sisa Pembagian Bilangan Biner)

```
int N = 25;
int K;
K = N% 2;

cout<<K;
```

Tercetak 1
Karena $25 \div 2 = 12$
sisa 1

Operator % biasa disebut modulus hanya berlaku untuk bilangan integer, sehingga % disebut sisa pembagian bilangan integer.

```
int A = 25;
cout<<A;
```

ERROR
Tipe float tidak dapat dikenali oleh operator %(
harus menggunakan fungsi)

Contoh hasil pembagian modulus:

10 % 2 = 0
10 % 3 = 1
10 % 4 = 2
10 % 5 = 0
8 % 6 = 2
0 % 2 = 0

c. Aritmatika dalam instruksi Cout().

```
int A = 25;
cout<<A;
```

Akan tercetak : 25
Nilai variable A tetap dan tidak berubah

```
int A = 25;
cout<<A+1;
```

Akan tercetak: 25
Yaitu $25 + 1$. Sedangkan nilai A tetap tidak berubah

```
int A = 25;
cout<<A = A+1;
```

Isi variable A ditambah 1 dahulu menjadi 26 kemudian dicetak.

Untuk semua instruksi cout berikut dengan nilai awal A=25

Int A = 25;	Tercetak	Keterangan
Cout<< A+1;	26	Nilai A tetap = 25
Cout<< A+=1;	26	Nilai A ditambah 1 dulu, menjadi 26 kemudian dicetak
Cout<< A++;	25	Nilai A dicetak dulu, kemudian ditambah 1, sekarang nilai A=26
Cout<< ++A;	26	Nilai A ditambah 1 dulu, menjadi 26 kemudian dicetak
Cout<< A--;	25	Nilai A dicetak dulu, kemudian dikurangi 1, sekarang nilai A=24
Cout<< --A;	24	Nilai A dikurangi 1 dulu, menjadi 24 kemudian dicetak
Cout<< ++A++		Error

d. Contoh Penulisan Aritmatika Statement

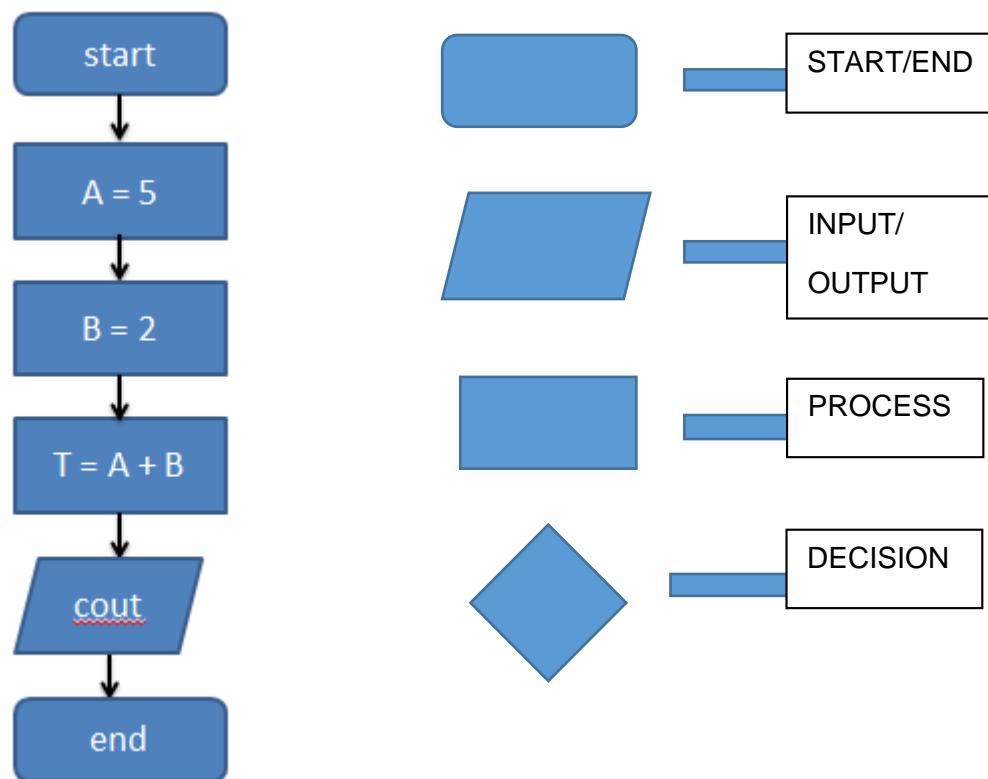
Penulisan dalam amtematika	Penulisan dalam program
$X = A + B / C$	$X = A + B / C;$
$X = (A + B) / C$	$X = (A + B) / C;$
$\frac{A + B}{C}$	$X = (A + B) / C$
$X = \frac{A + B}{C}$	
$\frac{A + B}{2C}$	$X = (A + B) / (2 * C)$
$X = \frac{A^2 - \sqrt{2B}}{4AC}$	$X = (A^2 - \sqrt{2B}) / (4 * A * C)$

3. Program Flowchart

Contoh:

```
#include <iostream>
using namespace std;
int main()
{
    int A, B, T;
    A= 5;
    B= 2;
    T = A + B;
    cout<<T;
}
```

Algoritma atau program ini jika digambarkan dalam bentuk bagan, yang dinamakan flowchart, akan digambarkan sebagai berikut:



a. Beberapa cara penulisan program menambah $5+2$

```
#include <iostream>
using namespace std;
int main()
{
    int A, B, T;
    A= 5;
    B= 2;
    T = A + B;
    cout<<T;
}
```

Tercetak : 7

```
#include <iostream>
using namespace std;
int main()
{
    int A, B, T;
    A= 5; B= 2; T = A + B;
    cout<<T;
}
```

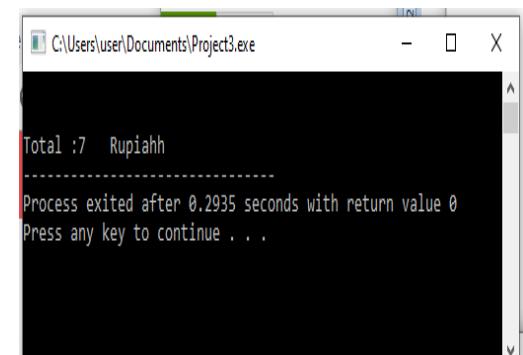
Tercetak : 7

(dalam satu baris
dapat ditulis lebih dari
satu instruksi)

```
#include <iostream>
using namespace std;
int main()

{ int A, B, T; A= 5; B= 2; T = A + B; cout<<T;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int A, B, T;
    A= 5;
    B= 2;
    T = A + B;
    cout<<"\n\n Total :"<<T<<""
Rupiah";
}
```



Bisa dicetak menggunakan keterangan “rupiah”

C. Soal/Latihan

1. Apa yang akan tercetak jika program berikut dijalankan:

a. <pre>#include <iostream> using namespace std; main() { int A = 5; cout<<A+2<<endl; cout<<A; }</pre>	b. <pre>#include <iostream> using namespace std; main() { int A, B; A = 5; B = 2; A = B; cout<<A <<endl; cout<<B; }</pre>
c. <pre>#include <iostream> using namespace std; main() { int A, B, C; A = 7; B = (A / 2) * 2; C = A - B; cout<<C; }</pre>	d. <pre>#include <iostream> using namespace std; main() { int A = 5, B = 2, X; X = A; A = B; B = X; cout<<A <<endl; cout<<B; }</pre>

2. Tulis pernyataan matematika berikut dalam bentuk pemrograman:

a. $A^2 + B^2 + C^3$

b. $\frac{A}{2} + C^2$

c. $\sqrt{\frac{x}{y} + x^2}$

D. Referensi

A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.

Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.

Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.

Kristanto, A. (2003). *Algoritma & Pemograman Dengan C++*. Yogyakarta: Graha Ilmu.

Lestari, F. D. (2017). Analisa Algoritma Faktor Persekutuan Terbesar (FPB) Menggunakan Bahasa Pemrograman C++. *Jurnal Evolusi*, 63-68.

- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- S, R. A. (2018). *Logika Algoritma dan Pemrograman Dasar*. Bandung: Modula.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 5

PREPROCESSOR DAN LIBRARY FUNCTION

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu mampu memahami dan mengerti dari penggunaan dari setiap fungsi pada preprocessor dan library function. Serta mahasiswa mampu menerapkan penggunaannya di dalam pemrograman.

B. Uraian Materi

1. Definisi Preprocessor

Preprocessor include merupakan salah satu jenis pengarah preprocessor directive (#include) yang terdapat pada C++. Atau dengan kata lain dapat dikatakan bahwa preprocessor merupakan program yang sangat berperan penting dalam mengontrol jalannya alur program. Dimana preprocessor akan selalu dijalankan lebih dulu ketika suatu proses kompilasi sudah terjadi. Pada suatu compiler C++ proses penggeraan preprocessor akan dilakukan dengan sendirinya sehingga dalam proses penggunaannya, seorang programmer tidak perlu menjalankan suatu proses preprocessor terlebih dahulu.

Dimana bentuk umum dari preprocessor adalah sebagai berikut:

```
# include <nama_file>
```

Bentuk awal dari #include <nama_file> merupakan bahwa pencarian suatu file akan dilakukan pada direktori khusus yaitu yang merupakan direktori file include. Sedangkan bentuk yang kedua seperti #include "namafile" merupakan bahwa pencarian suatu file dilakukan pertama kali pada direktori aktif dimana tempat program sumber dan apabila tidak ditemukan maka pencarian akan dilanjutkan kepada direktori lainnya yang sesuai dengan perintah pada suatu sistem operasi.

Bentuk umum preprocessor tidak diakhiri dengan tanda semicolon, hal ini dikarenakan bentuk tersebut bukan merupakan suatu pernyataan, tetapi merupakan suatu preprocessor directive. Dimana baris tersebut akan memerintahkan kepada suatu compiler yang akan menyisipkan suatu file lain. Dalam hal ini adalah file yang berakhiran .h atau disebut sebagai file header.

#include <iostream.h>	Digunakan pada program yang akan melibatkan objek dengan cout
#include <conio.h>	Akan digunakan apabila sudah melibatkan clrscr(), yang merupakan suatu instruksi yang berfungsi membersihkan layar
#include <iomanip.h>	Akan digunakan apabila telah melibatkan instruksi setw() yang berfungsi mengatur lebar dari suatu tampilan data pada program.
#include <math.h>	Akan digunakan oleh suatu program dimana menggunakan perintah sqrt () yang berfungsi untuk sebuah operasi pada fungsi matematika kuadrat.

Adapun cara kerja suatu preprocessor adalah membaca perintah dari pengarah suatu preprocessor yang akan dilakukan dalam suatu program yang akan diawali dengan bentuk # atau pound.

Ada 3 jenis dari preprocessor sebagai berikut:

a. Include file

Pada include file merupakan suatu preprocessor berfungsi untuk melibatkan pustaka yang berupa suatu file header yang terdapat pada sebuah program yang akan dikerjakan. Dimana dengan preprocessor ini maka kita dapat memasukkan suatu kode instruksi yang berada pada sebuah file header yang sedang dibuat. Dimana format penulisannya adalah sebagai berikut:

```
#include <nama_file_header.h>
```

#include "nama_file_header" dimana tanda kutip disini berfungsi untuk memasukkan suatu alternatif path dari suatu file dan akan disertakan pada suatu program.

b. Definisi makro

Pada definisi makro suatu preprocessor ini akan berfungsi sebagai penentu definisi dari suatu identifier tertentu yang akan ditulis dalam suatu program C++. Dimana preprocessor ini juga dapat digunakan untuk

menentukan rumus dari suatu makro fungsi. Preprocessor ini akan dijalankan apabila dengan menyertakan suatu preprocessor directif #define, #undef dalam suatu kode program yang sering kita kerjakan. Adapun format penulisan directive #define adalah sebagai berikut:

```
#define nama_makro nilai_makro
```

c. Pengarah kondisional kompilasi

Pada proses ini, suatu preprocessor berfungsi mengarahkan akan kerja dari suatu program yang kita buat dengan beberapa pengarah dari preprocessor yang digunakan untuk memberikan dan mengatur akan sebuah solusi yang dapat dijalankan dalam program C++. Dimana preprocessor ini dijalankan dengan melibatkan preprocessor directive #if, #else, #elif, #ifdef, #ifndef ke dalam kode program yangs sedang kita buat.

Berikut adalah penjelasan dari beberapa instruksi – instruksi dari preprocessor yaitu sebagai berikut:

a. #define

Pada nstruksi #define berfungsi untuk melaksanakan proses substitusi makro dari suatu lembar teks yang satu ke lembar teks yang lain. Dimana proses ini akan melalui suatu file pada teks tersebut yang akan digunakan. Seperti instruksi #define nama karakter-sequence. Dimana pada instruksi ini tidak terdapat tanda titik koma. Sehingga apabila diawal karakter telah berhasil maka dia akan berhenti di akhir baris pada suatu program tersebut.

Misal:

“true” untuk angka 1

“false” untuk angka 0

Maka dapat dituliskan:

```
#define true 1
```

```
#define false 0
```

Dimana compiler akan mengganti 1 dan 0 setiap kali menemui true dan false.

b. #error

Pada perintah #error dalam suatu program disini berfungsi untuk memaksa suatu compiler dalam menghentikan suatu proses kompilasi

pada suatu program. Dimana pada awalnya #error ini berfungsi di proses “debugging”. Perintah yang sering digunakan adalah #error message. Dimana apabila menemui #error akan muncul suatu notif yang berupa angka baris pada suatu program yang ingin dijalankan.

c. #if, #ifdef, #ifndef, #else, #elif, #endif

Perintah – perintah berupa #if, #ifdef, #ifndef, #else, #elif, #endif processor disini akan digunakan pada jenis versi perintah program yang dilakukan secara tersukur. Sehingga apabila ekspresi #if, #ifdef atau #ifndef bernilai benar maka kodennya tepat berada diantara salah satu dari instruksi tersebut dan #endif akan tersusun pada suatu program. Tetapi apabila yang terjadi adalah sebaliknya, maka akan terlewati atau tidak tersusun sama sekali. #endif disini berfungsi untuk menandai akhir dari suatu blok. Dan instruksi #else akan digunakan kepada salah satu yaitu instruksi – instruksi tersebut dengan cara yang sama seperti “else” pada perintah program C

d. #include

Instruksi #include berfungsi untuk memerintah suatu compiler membaca dan menyusun suatu file pada sumber yang lain.

e. Pragma

#pragma merupakan suatu perintah yang sudah diketahui perancangannya yang terdapat suatu jenis perintah yang harus ditujukan pada suatu perintah dalam sebuah program. Suatu perintah akan terdapat sebuah pilihan untuk membantu dalam pembuatan dari trace suatu program. Dimana proses penggandaan (trace) ini akan ditentukan oleh instruksi dari suatu #pragma.

f. #undef

Pada instruksi #undef disini berfungsi untuk memindahkan suatu definisi yang telah ditentukan sebelumnya dari suatu nama makro yang mengikutinya. Dimana prinsi penggunaan suatu #undef adalah untuk memungkinkan dimana nama makro ditempatkan hanya berapa pada bagian code yang memerlukannya saja.

2. Definisi Library Function

Fungsi merupakan kumpulan dari suatu instruksi yang terdapat pada program yang akan dikelompokkan menjadi fungsi sendiri atau letaknya terpisah dari suatu program yang akan menggunakan fungsi tersebut.

Sedangkan standar library function merupakan suatu fungsi standart yang sudah disediakan oleh program Bahasa C dimana untuk menggunakannya, maka kita harus mencantumkan suatu header file dari fungsi tersebut yaitu dengan menggunakan perintah #include dalam suatu program.

Misal:

```
#include <stdio.h>
void main()
{
    int C, D, R;
    C = 7;
    D = 8;
    R = C + D;
    printf ("%i", T);
}
```

Berdasarkan contoh diatas, dimana instruksi printf () tersebut bukanlah merupakan suatu instruksi yang secara langsung melakukan proses pekerjaan untuk mencetak, tetapi hanya memanggil sebuah fungsi tersebut yang bernama printf(). Dimana fungsi printf ("%i", T) adalah untuk memerintahkan computer supaya menjalankan semua perintah yang ada dalam sebuah fungsi yang dinamakan dengan printf(). Sehingga untuk memanggil suatu fungsi adalah hanya dengan menggunakan nama pada fungsi tersebut.

Contoh penggunaan library function yaitu:

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

Adapun beberapa fungsi yang ada pada stdio.h adalah sebagai berikut:

- a. Printf ()
- b. Scanf()
- c. Getchar()
- d. Gets()
- e. Puts()

Sehingga dalam merancang suatu function ada beberapa hal yang harus diperhatikan, seperti::

- a. Input adalah data yang akan menjadi masukka suatu sistem
- b. Proses, yaitu bagaimana proses algorima dari program yang akan digunakan dalam suatu fungsi tersebut.

- c. Output, merupakan suatu informasi yang akan dikembalikan oleh fungsi kepada si pemanggil instruksi tersebut.

Penulisan fungsi dibagi menjadi 2 macam, yaitu:

- Function
- Prototype

Fungsi prototype merupakan suatu pendeklarasian dari fungsi yang disebut sebagai kepala atau judul fungsi atau dikenal sebagai pengenal fungsi dari suatu program.

- Function Definition

Function definition merupakan suatu penulisan fungsi yang dilakukan secara lengkap dalam suatu program.

Contoh function dalam C++ yaitu:

```
#include <iostream>
using namespace std;
int addition (int x, int y) //called or
calling function berisi formal parameter
{
    int s;
    s=x+y;
    return s;
}
int main ()
{
    int z;
    z = addition (4,1); // caller function
berisi actual parameter
    cout << "The result is " << z;
    return 0;
}
```

Jawab:

```
int addition (int x, int y)
z = addition( 4, 1)
```

Dimana nilai dari 4 dan 1 merupakan nilai yang dikirim oleh suatu fungsi pemanggil atau yang disebut sebagai caller function ke suatu fungsi called function melalui suatu parameter yang ada. Dimana argument pada caller function disini merupakan actual parameter dan suatu argument pada calling function merupakan formal parameter

Terdapat 3 macam passing pada suatu function dalam C++ yaitu:

a. Passing by value

Contoh:

```
//passing by value
#include <iostream>
using namespace std;
void foo(int val)
{   val = 7;
}
int main()
{   int value = 6;
cout << "value = " << value << '\n';
foo(value);
    cout << "value = " << value << '\n';
    system("pause");
    return 0;
}
```

b. Passing by reference

Contoh:

```
//passing by reference
#include <iostream>
using namespace std;
void foo(int &val)
{   val = 7;
}
int main()
{   int value = 6;
cout << "value = " << value << '\n';
foo(value);
cout << "value = " << value << '\n';
system("pause");
return 0;
}
```

c. Passing by address

Contoh:

```
//passing by address
#include <iostream>
using namespace std;
void foo(int *ptr)
{   *ptr = 7;
}
int main()
{   int value = 6;
```

```

cout << "value = " << value << '\n';
foo(&value);
cout << "value = " << value << '\n';
system("pause");
return 0;
}

```

3. Penggunaan Library Function

Adapun beberapa penggunaan library function adalah sebagai berikut:

- Library function yang berhubungan dengan penanganan tanggal dan waktu

Dalam penanganan tanggal dan waktu pada Bahasa C mempunyai suatu pustaka sendiri, yaitu time.h. dimana fungsi tersebut memiliki beberapa kumpulan fungsi sebagai berikut:

Fungsi	Deskripsi
Difftime	Untuk menghitung perbedaan waktu yang terjadi dari dua nilai time_t
Time	Untuk mengembalikan suatu nilai waktu yang ada saat ini
Clock	Untuk mengembalikan suatu nilai perhitungan clock dari sebuah prosesor
Ctime	Untuk melakukan konversi nilai time_t ke representasi tekstual
Strftime	Digunakan untuk melakukan konversi suatu objek struct tm pada representasi tekstual
Wcsftime	Digunakan untuk melakukan konversi objek struct tm ke suatu representasi custom wide string
Gmtime	Digunakan untuk melakukan suatu konversi nilai time_t ke suatu ekspresi kalender menggunakan koordinat universal GMT

- Library function yang berhubungan dengan fungsi matematika

Library function matematika digunakan untuk operasi matematika. Library function yang digunakan dalam matematika adalah math.h. sehingga dengan penggunaan library ini, seorang programmer harus

menuliskan suatu instruksi yaitu #include <math.h>. adapun beberapa fungsi yang digunakan dalam fungsi matematika ini adalah sebagai berikut:

Fungsi	Deskripsi
Fabs()	Digunakan untuk memberikan nilai kembalian yang berupa suatu nilai yang absolut
Ceil()	Digunakan untuk memperoleh suatu bilangan bulat yang paling kecil dimana nilai yang muncul tidak boleh kurang dari nilai argumennya
Floor()	Digunakan untuk memperoleh suatu bilangan bulat yang muncul tidak lebih besar dari nilai argumennya
Round()	Digunakan untuk memperoleh suatu nilai bilangan bulat yang terdekat dari nilai argumennya
Trunc()	Digunakan untuk memperoleh suatu nilai bilangan bulat dari argumennya
Pow()	Digunakan untuk menghitung perpangkatan
Pow10()	Digunakan untuk menghitung perpangkatan 10
Sqrt()	Digunakan untuk menghitung akar
Hypot()	Digunakan untuk menghitung sisi miring dari segitiga siku - siku
Log()	Digunakan untuk menghitung nilai algoritma

Contoh:

- 1) Fungsi akar kuadrat sqrt()

```
#include <stdio.h>
#include <math.h>
void main()
{
    Int X, Y;
    X = 16
    Y = sqrt(X);
    Printf ("%i", Y);
}
```

Penjelasan:

$X = 16$

$Y = 4$ (akar dari 16)

$Y = \text{sqrt}(X)$. Pada perintah ini bukan untuk menghitung akar kuadrat X tetapi hanyalah sebuah perintah yang memanggil fungsi `sqrt()` supaya semua fungsi yang ada di dalam `sqrt()` dapat dilaksanakan kemudian. Dan hasil perhitungan dari `sqrt` disimpan oleh Y.

2) Fungsi pangkat (pow)

```
#include <stdio.h>
#include <math.h>
void main()
{int X, Y, Z;
X = 7;
Y = 2;
Z = pow (X, Y);
Printf("%i", Z);
}
```

Penjelasan:

$X = 7$

$Y = 2$ (merupakan pangkat)

Maka $Z = 7^2 = 49$

C. Soal Latihan/Tugas

1. Buatlah program sederhana dengan menggunakan bahasa C dengan menggunakan salah satu library function pada matematika?
2. Jelaskanlah perbedaan dari function prototype dengan fungsi definition dan sebutkan contohnya?
3. Apakah function hanya bisa dituliskan dibawah `main()`. Jelaskan?
4. Apakah bisa dilakukan pemanggilan function oleh function lain? Lalu bagaimana pengaturan letaknya? Jelaskan?
5. Apakah bisa menuliskan function tanpa menuliskan prototype? Jelaskan bagaimana caranya?

D. Referensi

- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 6

INPUT, OUTPUT, ALGORITMA DAN PENGETAHUAN TERKAIT

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu:

1. Membedakan operasi input dan output
2. Membuat program dengan operasi input dan output
3. Memahami penggunaan varibel dan konstanta dalam operasi input dan output
4. Memahami persoalan dan memahami penyelesaiannya.

B. Uraian Materi

Menginput data melalui keyboard dapat diartikan sebagai kegiatan memasukkan data dengan mengetik melalui keyboard. Dalam prosesnya, permintaan pengetikan data dapat berlangsung saat ada program yang berjalan, sehingga diperlukan instruksi atau perintah input. Sedangkan untuk menampilkan dilayar bisa dilakukan melalui perintah output.

Penerapan algoritma dengan ilmu terkait adalah merepresentasikan pengetahuan yang diketahui untuk menyusun algoritma kemudian melanjutkanya ke bahasa pemrograman.

1. Output

cout

Dengan **cout** (dibaca C out) pemrogram dapat menempatkan suatu informasi ke standar output (umumnya berupa layar). Instruksi cout membutuhkan file iostream untuk di include **#include <iostream>**. include library iostream berlaku juga untuk fungsi cin.

a. Mencetak konstanta tanpa format

```
#include<iostream>

using namespace std;
int main()
{
    cout<<"Universitas Pamulang";
}

Output:
Universitas Pamulang
```

Program di atas, ada satu buah instruksi mencetak sebuah konstanta string yaitu cout<<"Universitas Pamulang"; instruksi tersebut akan mencetak kalimat **Universitas Pamulang** di layar.

Beberapa contoh pencetakan nilai konstanta tanpa format pada C++

Instruksi cetak	Yang tercetak di layar
cout<<"Universitas Pamulang";	Universitas Pamulang
cout<<"U";	U
cout<<'U';	U
cout<<123;	123
cout<<-32768	-32768
cout<<123.456	123.456
cout<<123.4567	123.457 (lebih dari 6 digit dibulatkan ke atas)
cout<<1234567.890	1.23457e+006
cout<<123E2	12300
cout<<123.45E3	123450

b. Mencetak konstanta dengan format

Perhatikan contoh instruksi pada program c++ berikut:

Code Program	Output
<pre>#include<iostream> #include<iomanip> using namespace std; int main() { int x; x=1234; cout<<123456789<<"\n"; cout<<setw(9)<<x; }</pre>	<pre>123456789 1234</pre>

Untuk dapat menggunakan instruksi setw() harus menambahkan #include <iomanip>. Instruksi setw(9) berfungsi untuk menentukan tata letak hasil cetakan elemen berikutnya yaitu variabel x, sehingga tercetak dengan lebar 9 lokasi dengan hasil cetakan rata kanan (right justified).

Instruksi setw(), hanya berlaku mengatur format cetakan satu elemen cetakan yang mengikutinya. Elemen" selanjutnya tidak dipengaruhi lagi oleh setw(). Karena setw hanya berlaku satu elemen cetakan berikutnya biasanya pemrogram lebih memilih menulis setw() digabung satu baris dengan elemen yang akan dicetak.

Code Program	Output
<pre>#include<iostream> #include<iomanip> using namespace std; int main() { int x,y,z; x=1234; y=5678; z=6789; cout<<123456789<<endl; cout<<setw(9)<<x; cout<<endl; cout<<setw(9)<<y<<endl; cout<<z; }</pre>	123456789 1234 5678 6789

Instruksi `endl` mempunyai fungsi yang sama dengan “`\n`” dimana kursor akan pindah ke posisi awal pada baris berikutnya. `setw()` dan `endl` biasa disebut manipulator.

Beberapa contoh penggunaan manipulator `endl`

Code Program	Output
<code>cout<<"ABC"; cout<<endl<<"DEF";</code>	ABC DEF
<code>cout<<"ABC"; cout<<endl; cout<<"DEF"<<endl;</code>	ABC DEF
<code>cout<<"ABC"<<endl; cout<<"DEF"</code>	ABC DEF
<code>Cout<<"ABC"<<endl<<"DEF";</code>	ABC DEF

Manipulator yang digunakan dalam bahasa C++, selengkapnya di sub bab 1.2.2

c. Manipulator

Umumnya Manipulator digunakan untuk menata tampilan dari data. misalnya untuk menyisipkan karakter *newline* digunakan `endl`.

Manipulator	Fungsi
<code>endl</code>	End line, akan membuat kursor berpindah ke posisi awal pada baris berikutnya sama dengan “ <code>\n</code> ”
<code>ends</code>	Meyisipkan karakter null
<code>flush</code>	Mencetak isi buffer

Manipulator	Fungsi
Dec hex oct	konversi ke bilangan decimal konversi ke bilangan hexadecimal konversi ke bilangan octal
setbase(n) setw(n) setfill(c)	konversi ke bilangan basis n Menata lebar cetakan sebesar n Mengisi leading fields dengan karakter c
setprecision(n) setiosflags(lf) resetiosflags(lf)	Membuat lebar decimal point = n Menset atau mangatur format yang diatur dengan tanda format ::ios atau ios Me-reset format yang diset atau diatur oleh setiosflags()

Setiap manipulator setxxxxx() atau resetxxxxx() digunakan harus menambahkan / menggunakan **#include<iomanip>**

Contoh Penggunaan Manipulator

Code program	Output
<pre>#include<iostream> #include<iomanip> using namespace std; int main() { int x; x=64; cout<<123456789<<endl; cout<<setw(9)<<123<<endl; cout<<123<<endl; cout<<hex<<x<<endl; cout<<oct<<x<<endl; cout<<dec<<x<<endl; cout<<setfill('.');cout<<setw(10)<<x<<endl; cout<<x<<endl; }</pre>	123456789 123 123 40 100 6464 64

d. Tanda Format

mengatur format/bentuk cetakan diperlukan tanda format yang harus diset dengan menggunakan sebuah manipulator setiosflags() dan direset dengan menggunakan sebuah manipulator resetiosflags().

Tanda Format	Fungsi
ios::left ios::right	dicetak rata kiri dicetak rata kanan
ios::scientific ios::fixed	dicetak dalam bentuk scientific dicetak dalam bentuk fixed point

Tanda Format	Fungsi
ios::dec ios::hex ios::oct	dicetak dalam bentuk desimal dicetak dalam bentuk hexadesimal dicetak dalam bentuk oktal
ios::uppercase	dicetak dalam bentuk hexadesimal dicetak dengan huruf besar
ios::showbase	Menambahkan 0x diawal hasil cetakan yang berbentuk hexadesimal atau 0 (nol) pada cetakan yang berbentuk oktal
ios::showpoint	Menampilkan desimal point pada hasil cetakan yang mengandung pecahan
ios::showpos	Menambahkan tanda + pada hasil cetakan yang bernilai positip

Contoh Penggunaan tanda format

Code program	Output
#include<iostream> #include<iomanip> using namespace std; int main() { int x=255; double y=123.44; cout<<setiosflags(ios::showbase); cout<<setiosflags(ios::left)<<setw(10)<<x<<endl; cout<<resetiosflags(ios::left); cout<<setiosflags(ios::right)<<setw(10)<<x<<endl; cout<<setiosflags(ios::hex)<<x<<endl; cout<<resetiosflags(ios::hex); cout<<setiosflags(ios::oct)<<x<<endl; cout<<resetiosflags(ios::oct); cout<<setiosflags(ios::dec)<<x<<endl; cout<<setiosflags(ios::fixed)<<setprecision(5)<<y<<endl; cout<<resetiosflags(ios::fixed); cout<<setiosflags(ios::scientific)<<setprecision(5)<<y<< endl; }	255 oxff 255 0377 255 123.44000 1.23440e+00 2

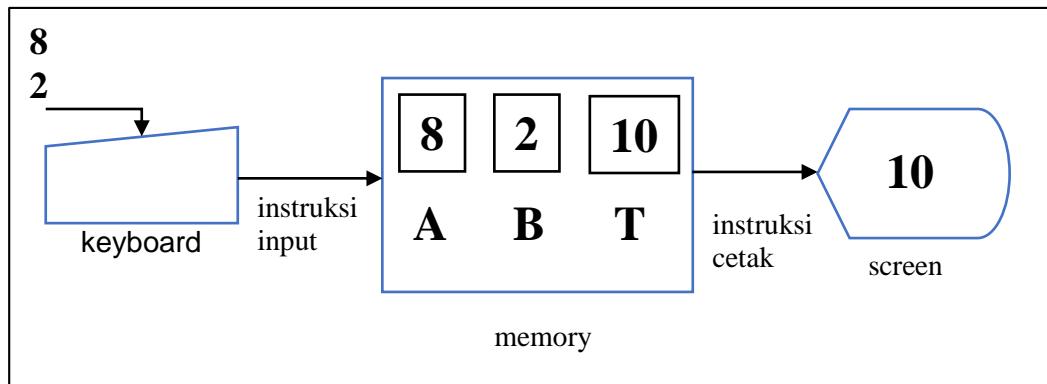
2. Input

Perintah input pada program C++ berbeda dengan bahasa pada program C yaitu tidak menggunakan format. Pada program C++, tipe data yang dininput tergantung tipe data penerimanya. Bila tipe data atau varibel penerimanya bertipe int, maka data yang diinput, apapun bentuknya dan besarnya akan diterima oleh variabel tersebut sebagai tipe int.

cin >> variable

cin (dibaca C in) disebut objek yang digunakan untuk menerima data dari standar input (keyboard)

Berikut ini ilustrasi menginput data melalui keyboard:



Data yang diketik melalui keyboard harus diterima oleh suatu area dalam memori baik baik dalam sebuah atau sekumpulan variabel.

Misal yang di input nilai integer 8 dan 2, dalam memory disiapkan 3 buah variabel (**A**, **B** dan **T**). **A** untuk menampung nilai **8** dan **B** untuk menampung nilai **2**. Kemudian **T** untuk menampung hasil penjumlahan **A** dan **B**. kemudian mencetak nilai **T** yaitu **10**.

Program C++

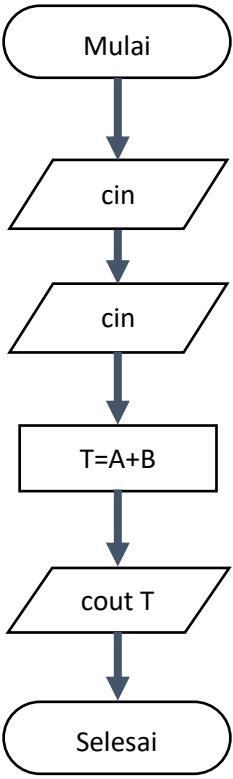
```
#include<iostream>
using namespace std;
int main()
{
    int A, B, T;
    cin>>A;           //cin = instruksi input
    cin>>B;

    T=A+B;

    cout<<"T = "<<T;//cout = intstruksi output
}
```

Output :

T = 10

Flowchart	Proses
 <pre> graph TD Mulai([Mulai]) --> Cin1[/cin/] Cin1 --> Cin2[/cin/] Cin2 --> T[A = A + B] T --> Cout[/cout T/] Cout --> Selesai([Selesai]) </pre>	<p>Nilai A, B dan T belum ditetapkan, isinya ada tapi tidak diketahui. Sewaktu melaksanakan instruksi input ke A, kursor kedip-kedip di layar</p> <p>Pemakai mengetik 8 kemudian menekan tombol <enter>. Angka 8 akan muncul di layar dan masuk ke variabel A sebagai nilai numeric, dan kursor turun ke baris berikutnya.</p> <p>Pemakai mengetik 2 <enter>, 2 akan muncul di layar dan masuk ke variabel B sebagai nilai numeric, dan kursor turun ke baris berikutnya karena menekan tombol <enter>.</p> <p>Proses $T=A+B$ tidak terlihat di layar. yang terlihat nilai 10 hasil instruksi mencetak T</p> <p>tanda ↴ hanya untuk menyatakan ditekan tombol enter. Tanda ↴ tidak muncul di layar.</p>

Untuk mempermudah pengguna lain dalam menjalankan program, sebaiknya kita menggunakan program berikut:

Program C++

```
#include<iostream>
using namespace std;
int main()
{
    int A, B, T;
    cout<<"Program Menjumlahkan dua buah
bilangan"<<endl;
    cout<<"Masukkan angka pertama : ";
    cin>>A;
    cout<<"Masukkan angka kedua : ";
    cin>>B;

    T=A+B;

    cout<<"Total dari penjumlahan "<<A<<"+ "<<B<< " =
    "<<T;
}
```

a. Menginput karakter melalui keyboard

- 1) Menggunakan `cin >> var;`

Contoh	Output
<pre>#include<iostream> using namespace std; int main() { char A; cout<<"Masukkan sebuah karakter : "; cin >> A; cout<<A; }</pre>	<p>Masukkan sebuah karakter : U ↴ U</p> <p>Bila karakter yang diinput adalah U.</p>

Setelah menginput karakter, harus diakhiri dengan menekan tombol enter (↵)

Bila diinput	Output
A	A
75	7
BCA	B
'UNPAM'	'
"UNPAM"	"

hanya karakter pertama saja yang akan tercetak di layar.

- 2) Menggunakan `var=getch()` dan `var=getche()`

Jika menggunakan `cin`, setelah karakter di input maka harus menekan tombol enter (↵), sedangkan fungsi `getch()` dan `getche()` berguna untuk membaca karakter tanpa harus menekan enter. Selain

itu, fungsi ini juga dipakai untuk membaca karakter seperti spasi, tabulasi ataupun enter. Untuk dapat menggunakan fungsi getch() dan getche() harus menambahkan library **<conio.h>**

Perbedaan dari kedua fungsi ini adalah:

- a) **getch()** : tidak menampilkan karakter dari tombol yang ditekan.
- b) **getche()** : menampilkan karakter dari tombol yang ditekan.

Contoh getch()	Output
<pre>#include<iostream> #include<conio.h> using namespace std; int main() { char A; cout<<"Masukkan sebuah karakter : "; A=getch(); cout<<A; }</pre>	<p>Masukkan sebuah karakter : U</p> <p>Bila karakter yang diinput adalah U.</p> <p>Karakter U yang tampil di layar bukanlah yang diketik, tetapi hasil instruksi cetak cout<<A. apa yang diketik di keyboard tidak terlihat di layar.</p>
Contoh getche()	Output
<pre>#include<iostream> #include<conio.h> using namespace std; int main() { char A; cout<<"Masukkan sebuah karakter : "; A=getche(); cout<<A; }</pre>	<p>Masukkan sebuah karakter : UU</p> <p>Bila karakter yang diinput adalah U.</p> <p>Ada dua huruf U yang tampil di layar. pertama adalah hasil ketikan A=getche(), kedua adalah instruksi cetak cout<<A.</p>

getch() dapat dimanfaatkan untuk menunggu sembarang tombol di keyboard ditekan. Pada kondisi seperti ini tidak diperlukan variabel.

Contoh getch()	Output
<pre>#include<iostream> #include<conio.h> using namespace std; int main() { cout<<"Silakan tekan tombol apapun untuk mengakiri program ini"<<endl; getch(); }</pre>	Silakan tekan tombol apapun untuk mengakhiri program ini.

b. Menginput String

String pada bahasa C++ pengertiannya sama dengan pengertian pada umumnya, yaitu deretan atau kombinasi dari sejumlah karakter.

1) Menggunakan cin>>var

```
#include<iostream>

using namespace std;
int main()
{
    char U[5]; //array U
    cin>>U;
    cout<<U;
}
```

U	0	1	2	3	4	5
	U	N	P	A	M	

← Yang tersimpan di dalam array
U bila diketik UNPAM

Hanya bagian ini yang bisa diisi

Bila di input	Output
UNPAM	UNPAM
UN PAM	UN
85	85
“UNP”	“UNP”
‘UNP’	‘UNP’

Bila dalam string yang diinput mengandung spasi seperti: **UN PAM**, maka hanya **UN** yang tersimpan di array U.

2) Menggunakan **cin.getline(var,sizeof(var));**

Penggunaan `cin.getline(var,sizeof(var))` dapat menyimpan karakter spasi di array.

```
#include<iostream>

using namespace std;
int main()
{
    char U[7]; //array U
    cin.getline(U,7);
    cout<<U;
}
```

Output:
UN PAM

Bila yang diketikan UN PAM

Agar tidak salah mengetik, maka sebaiknya instruksi: **cin.getline(U,7);** diganti dengan **cin.getline(U,(sizeof(U));**

c. Menginput nilai numerik melalui keyboard

```
#include<iostream>

using namespace std;
int main()
{
    int U;
    cin>>U;
    cout<<U;
}
```

Output:
30 ↴
30
Bila yang diinput adalah angka 30

Untuk tipe data lain long int, float atau double. Digunakan instruksi yang sama perbedaanya hanya pada tipe data dan jangkauan nilai yang tersimpan dalam variabel tersebut.

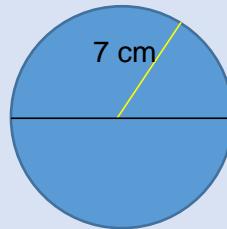
3. Algoritma Merepresentasikan apa yang diketahui

Pada pertemuan ini akan diperlihatkan, bahwa algoritma adalah “sekedar” menuliskan atau merepresentasikan apa yang kita ketahui. “Tidak ada yang diketahui, tidak ada yang ditulis, tidak ada yang ditulis, tidak ada

algoritma". Memahami persoalan dan memahami penyelesaiannya lebih sulit dan memakan waktu lebih lama dibandingkan menulis algoritmanya.

Contoh :

Susun Algoritma untuk mencetak luas sebuah lingkaran bila diketahui jari-jari = 7 cm.



Agar dapat menyusun algoritma untuk menjawab soal di atas, ada tiga hal yang perlu dikuasai:



- Mengerti dan memahami maksud dari persoalan yang akan diselesaikan. Jika tidak mengerti maksud soal tersebut, maka tidak ada yang dapat dikerjakan. Untuk contoh soal di atas, Anda harus mengerti apa yang dimaksud dengan luas lingkaran. Kemudian mengetahui cara atau rumus untuk menghitung luas Lingkaran jika jari-jari sudah diketahui.

$$\text{Luas Lingkaran} = \phi * r^2$$

$$\text{Luas Lingkaran} = \phi * r * r$$

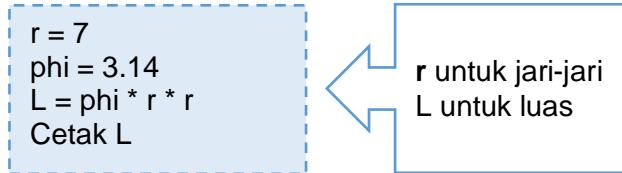
$$\phi = 3.14 = 22/7$$

$$r = \text{jari-jari}$$

Pengetahuan mengenai luas lingkaran, bukan didapat dari mata kuliah algoritma, tapi merupakan pengetahuan umum yang didapatkan dari jenjang pendidikan sebelumnya.

Tidak paham rumus atau cara menghitung luas lingkaran maka tidak dapat membuat algoritma.

- b. Setelah paham menghitung luas lingkaran dari soal yang dicontohkan:
Maka mulailah menuangkan jalan pikiran tersebut dalam bentuk langkah-langkah yang terurai secara: **rinci, lengkap dan tersusun secara logis.**



- c. Membuat Program komputer

Setelah jalan pikiran tertuang dalam bentuk **algoritma**, maka algoritma tersebut dapat diberikan ke komputer sebagai langkah-langkah penyelesaian pekerjaan. Untuk itu algoritma tersebut harus dituangkan ke dalam suatu **bahasa pemrograman**, misal bahasa C++. Algoritma yang ditulis dalam suatu bahasa pemrograman, disebut **program**.

```

#include<iostream>
using namespace std;
int main()
{
    float phi = 3.14, r, L;
    r=7;
    L = phi*r*r;
    cout<<"Luas Lingkaran = "<<L;
}
  
```

Output :

Luas Lingkaran = 153.86

C. Soal Latihan / Tugas

1. Susunlah program untuk mencetak empat buah bilangan bulat kemudian mencetak total keempat buah bilangan tersebut!
2. Jika diketahui nilai X=7, Y=5, Z=3. Berapa isi X, Y, dan Z jika dikenai instruksi sebagai berikut:
a. Z = Y b. Z=Z+X c. Z=Z-X*Y d. Z=X%Y
3. Jika diketahui A=5, B=2, berapa isi A dan B dan T jika dikenai instruksi sebagai berikut:

T=A

A=B

B=T

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR.* BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua.* Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C.* Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mitra Wacana Media.

PERTEMUAN 7

CONTROL STATEMENT MENGGUNAKAN IF

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu:

1. Memahami fungsi statement IF
2. Dapat memahami logika dengan fungsi IF
3. Dapat membuat program sederhana menggunakan fungsi IF

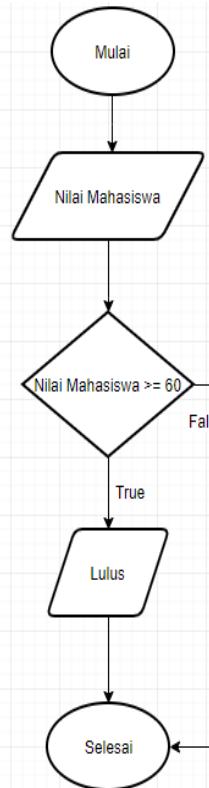
B. Uraian Materi

1. Definisi

Control statement IF adalah statement pembuat keputusan paling simple . merupakan blok kode program dimana program akan melakukan sebuah proses jika suatu kondisi pada statement IF terpenuhi. Jika kondisi dari IF tersebut tidak terpenuhi maka kode program tidak akan dijalankan. IF merupakan bagian dari statement yang mengatur tentang jalannya program.

If dapat diartikan sebagai pemilihan keputusan atau juga percabangan pada sebuah program. Contoh seorang dosen memberikan tugas kepada mahasiswa jika seorang mahasiswa mendapatkan nilai lebih dari 60 maka mahasiswa akan dinyatakan lulus. Dari contoh tersebut dapat dibuat flow chart sebagai berikut :

Syntax
If(Kondisi)
{
 //Kode yang akan dieksekusi jika benar
}



Dari flowchart diatas dapat dilihat bahwa nilai lulus akan muncul jika nilai mahasiswa lebih dari sama dengan 60, jika tidak maka akan langsung selesai. Syntax dasar dari if adalah sebagai berikut :

Kondisi pada fungsi if, setelah berjalan maka akan berupa true atau false. If menerima nilai Boolean jika bernilai true maka kode akan tereksekusi. Dari syntax diatas dapat dilihat bahwa kode yang akan dieksekusi oleh if adalah yang berada pada kurung kurawal "{}", jika tidak menggunakan kurung kurawal maka statement kode berikutnya akan menjadi kode yang berada di blok eksekusi.

Contoh penggunaan if:

Syntax:

If(kondisi)

Statement1;//statement tereksekusi

Statement2;//statement tidak tereksekusi

Listing:

```
#include<iostream>
using namespace std;
int main(){
    int nilai_mahasiswa;
    cout<<"Masukan nilai mahasiswa";cin>>nilai_mahasiswa;
    if(nilai_mahasiswa >= 60)
        cout<<"Lulus"<<endl;
    return 0;
}
```



int nilai_mahasiswa; → Artinya mendeklarasikan variable bernama nilai_mahasiswa dengan tipe data integer.
 cin>>nilai_mahasiswa; → artinya nilai dari variable nilai_mahasiswa akan diinputkan oleh user.
 If(nilai_mahasiswa >= 60) → Artinya menggunakan fungsi IF untuk menentukan jika variable nilai_mahasiswa lebih dari sama dengan 60 maka nilai akan tercetak lulus jika tidak maka tidak akan tertulis apa-apa.

Listing:

```
Include<iostream>
using namespace std;
Int main()
{
```

```

int total_belanja;
cout<<"Masukan total belanja = ";cin>>total_belanja;
if(total_belanja >= 50000){
    cout<<"Selamat anda mendapatkan diskon
Rp.5000"<<endl;
    total_belanja = total_belanja - 5000;
}
cout<<"total belanja anda adalah = "<<total_belanja<<endl;
cout<<"Terima kasih telah berbelanja di toko kami"<<endl;
return 0;
}

```



int total_belanja; → Artinya mendeklarasikan variable bernama total_belanja dengan tipe data integer.
 cin>> total_belanja; → artinya nilai dari variable nilai_mahasiswa akan diinputkan oleh user.
 If(total_belanja >= 5000) → Artinya menggunakan fungsi IF untuk menentukan jika variable total_belanja lebih dari sama dengan 5000 maka nilai akan tercetak anda mendapatkan diskon Rp. 5000 dan akan melakukan pengurangan sebanyak 5000 pada total belanja jika tidak maka hanya akan tercetak total_belanja apa-apa.

Dapat dilihat dari contoh syntax diatas kondisi pada IF menggunakan operator penghubung atau relational operator. Relational operator yang digunakan oleh C++ adalah sebagai berikut:

OPERATOR	MAKNA	CONTOH
==	Sama Dengan	X == Y
!=	Tidak sama dengan	X != Y
>	Lebih besar dari	X > Y
>=	Lebih besar dari atau sama dengan	X >= Y
<	Lebih kecil dari	X < Y
<=	Lebih kecil dari atau sama dengan	X <= Y

Kondisi pada if dapat kita lihat seperti berikut, if(**nilai** operator hubungan **nilai**). Nilai pada kondisi tadi dapat berupa variable ataupun nilai lainnya seperti angka atau text biasa. Contoh dari kondisi seperti berikut :

- a. Misalkan kita memiliki 2 variable **X** dan **Y**. X dan Y memiliki nilai yang sama yaitu 5, maka berikut kondisi yang akan terjadi

Kondisi	Nilai
if($X == Y$)	True
if($X != Y$)	False
if($X > Y$)	False
if($X >= Y$)	True
if($X < Y$)	False
if($X <= Y$)	True

- b. Misalkan kita memiliki 2 variable **X** dan **Y**. X memiliki nilai 7 dan Y memiliki nilai 5, maka berikut kondisi yang akan terjadi

Kondisi	Nilai
if($X == Y$)	False
if($X != Y$)	True
if($X > Y$)	True
if($X >= Y$)	True
if($X < Y$)	False
if($X <= Y$)	False

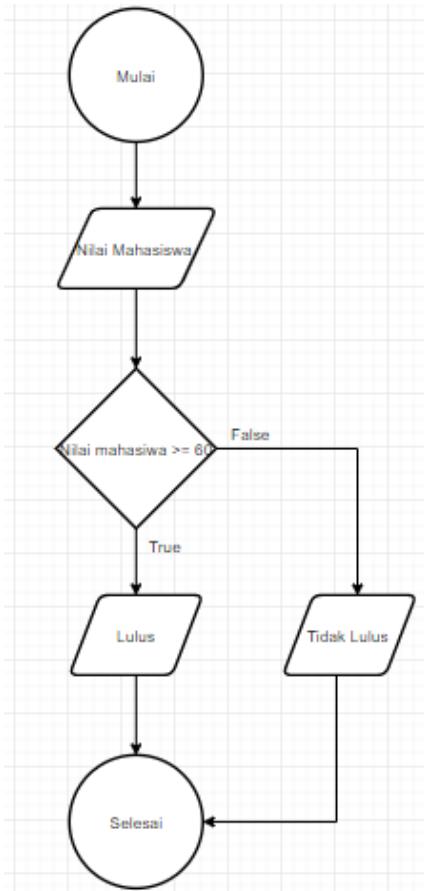
- c. Misalkan kita memiliki 2 variable **X** dan **Y**. X memiliki nilai 5 dan Y memiliki nilai 7, maka berikut kondisi yang akan terjadi

Kondisi	Nilai
if($X == Y$)	False
if($X != Y$)	True
if($X > Y$)	False
if($X >= Y$)	False
if($X < Y$)	True
if($X <= Y$)	True

Selain operator penghubung atau relational operator, If juga dapat menggunakan logical operator jika memiliki 2 atau lebih kondisi yang harus dijalankan. Berikut penjelasan tentang logical operator :

OPERATOR	MAKNA	CONTOH
&&	And/Dan	X == Y && A == B
	Or/Atau	X == Y A == B
!	Not/Tidak	!X

- d. Operator And(&&) adalah operator yang mengharuskan kedua kondisi bernilai benar atau true sehingga akan dianggap bernilai benar jika salah satu kondisi bernilai salah maka nilainya akan bernilai salah.
- e. Operator Or(||) adalah operator yang akan bernilai true atau benar jika salah satu kondisi benilai benar.
- f. Operator Not(!) adalah operator yang membalikan nilai, contoh jika kita punya variable X yang bernilai false maka jika !X akan menjadi nilai true. Sebelumnya telah dijelaskan bahwa if hanya menjalankan nilai yang bersifat benar atau true dan tidak akan mengeksekusi kode apapun jika salah. Lalu bagaimana jika bernilai salah?. Seperti yang telah disebutkan diawal if merupakan statement percabangan dan dapat digunakan untuk pemilihan. Maka jika ada nilai salah if dapat mengeksekusi statement kode lain. If-else merupakan kode sepenuhnya dari control statement if. Dimana else akan tereksekusi jika kondisi pada if bernilai salah. Maka dapat kita ilustrasikan dalam flowchart bentuk if-else :



Dari flowchart diatas dapat dilihat bahwa nilai lulus akan muncul jika nilai mahasiswa lebih dari sama dengan 60, jika nilai mahasiswa kurang dari 60 maka akan muncul tidak lulus dan program selesai. Syntax dasar dari if adalah sebagai berikut :

Listing:
If(Kondisi)
 {
 //Kode yang akan dieksekusi jika benar
 }
else
 {
 //kode yang akan dieksekusi jika salah
 }

Dari syntax dasar diatas maka dapat dibuat contoh syntax seperti ini :

Listing:

```
#include<iostream>
using namespace std;
int main(){
    int nilai_mahasiswa;
    cout<<"Masukan nilai
mahasiswa";cin>>nilai_mahasiswa;
    if(nilai_mahasiswa >= 60)
        cout<<"Lulus"<<endl;
else
    cout<<"Tidak Lulus"<<endl;
return 0;
}
```



int nilai_mahasiswa; → Artinya mendeklarasikan variable bernama nilai_mahasiswa dengan tipe data integer.
 cin>> nilai_mahasiswa; → artinya nilai dari variable nilai_mahasiswa akan diinputkan oleh user.
 If(nilai_mahasiswa >= 60) → Artinya menggunakan fungsi IF untuk menentukan jika variable nilai_mahasiswa lebih dari sama dengan 60 maka nilai akan tercetak lulus
 else → lanjutan dari fungsi if dimana menunjukkan hasil jika tidak maka akan tercetak Tidak Lulus apa-apa.

Selanjutnya berikut beberapa contoh penggunaan statement if :

Seorang siswa dijanjikan orang tuanya jika nilai ulangan siswa tersebut mendapat nilai 80 maka dia akan diajak liburan ke bandung. Jika nilainya dibawah 80 maka siswa akan diberikan tambahan les.

Listing:

```
#include<iostream>
using namespace std;
int main()
{
    int n_ulangan;
    cout<<"Masukan Nilai Ulangan : ";cin>>n_ulangan;
    if(n_ulangan >= 80)
    {
        cout<<"Nilai ulangan kamu =
"<<n_ulangan<<endl;
        cout<<"Liburan ke bandung"<<endl;
    }
}
```

```

    }
else
{
    cout<<"Nilai ulangan kamu =
"<<n_ulangan<<endl;
    cout<<"Belajar lebih rajin ya..."<<endl;
}
return 0;
}

```



int n_ulangan; → Artinya mendeklarasikan variable bernama n_ulangan dengan tipe data integer.
 cin>> n_ulangan; → artinya nilai dari variable n_ulangan akan diinputkan oleh user.
 If(n_ulangan >= 80) → Artinya menggunakan fungsi IF untuk menentukan jika variable n_ulangan lebih dari sama dengan 80 maka nilai akan tercetak Nilai ulangan kamu = n_ulangan dan akan mencetak liburan ke bandung
 else → jika tidak maka akan tercetak "Nilai ulangan kamu" dan "Belajar lebih rajin ya...".

Contoh program dengan fungsi IF:

Listing:

```

#include<iostream>
using namespace std;
int main()
{
    int n;
    cout<<" masukan bilangan : ";cin>>n;
    if(n % 2 == 0)
    {
        cout<<n<<" adalah Bilangan ganjil " <<endl;
    }
    else
    {
        cout<<n<<" adalah Bilangan genap " <<endl;
    }
    return 0;
}

```

	<p>int n; → Artinya mendeklarasikan variable bernama n dengan tipe data integer.</p> <p>cin>> n; → artinya nilai dari variable nilai_mahasiswa akan diinputkan oleh user.</p> <p>If(n % 2 == 0) → Artinya menggunakan fungsi IF untuk menentukan jika variable n % 2 sama dengan 0 maka nilai akan tercetak “adalah bilangan ganjil” jika tidak maka akan tercetak “adalah bilangan genap”.</p>
-----------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Listing:

```
#include<iostream>
using namespace std;
int main()
{
    char username[15];
    char password[15];
    cout<<"Masukan username : ";cin>>username;
    cout<<"Masukan Password : ";cin>>password;
    if(username == "admin" && password == "admin")
        cout<<"Anda adalah admin"<

```

	<p>char username[15]; → Artinya mendeklarasikan variable bernama username dengan tipe data char dan maksimal 15 kata.</p> <p>char password[15]; → Artinya mendeklarasikan variable bernama password dengan tipe data char dan maksimal 15 kata.</p> <p>cin>> username; → artinya nilai dari variable username akan diinputkan oleh user.</p> <p>cin>> password; → artinya nilai dari variable password akan diinputkan oleh user.</p> <p>If(username == "admin" && password == "admin") → Artinya menggunakan fungsi IF untuk menentukan jika variable username sama dengan “admin” dan password sama dengan “admin” maka nilai akan tercetak “anda adalah admin” jika salah satu tidak sesuai maka akan tercetak “anda bukan admin” fungsi logika && mengharuskan ke 2 nilai harus dalam kondisi benar untuk mendapatkan nilai true pada fungsi IF.</p>
-------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Listing:

```
#include<iostream>
#include<string.h>
using namespace std;
main()
{
    char A[15];
    char B[15];
    cout<<"Masukan kata 1 : ";cin>>A;
    cout<<"Masukan kata 2 : ";cin>>B;
    if(strcmp(A,B) == 0){
        cout<<"Kata 1 dan Kata 2 sama"<<endl;
    }else{
        cout<<"Kata 1 dan Kata 2 tidak sama"<<endl;
    }
    return 0;
}
```



char B[15]; → Artinya mendeklarasikan variable bernama A dengan tipe data char dan maksimal 15 kata.

char B[15]; → Artinya mendeklarasikan variable bernama B dengan tipe data char dan maksimal 15 kata.

cin>> A[15]; → artinya nilai dari variable A akan diinputkan oleh user.

cin>> B[15]; → artinya nilai dari variable B akan diinputkan oleh user.

If(strcmp(A,B) == 0) → Artinya menggunakan fungsi strcmp untuk membandingkan variable A dengan B apakah memiliki jumlah kata yang sama atau tidak jika sama maka hasil dari strcmp(A,B) adalah 0. Lalu dengan fungsi IF akan dibandingkan apakah strcmp(A,B) sama dengan 0 jika benar sama maka akan tercetak "Kata 1 dan 2 sama" jika tidak maka akan muncul "Kata 1 dan kata 2 tidak sama".

Listing:

```
#include<iostream>
using namespace std;
int main()
{
    int umur;
    cout<<"Masukan umur anda : ";
    cin>>age;
    if(umur >=13 && umur < 18)
    {
        cout<<"anda adalah remaja"<<endl;
    }
    else
    {
        cout<<"anda bukan remaja"<<endl;
    }
    if(umur >= 18)
    {
        cout<<"anda memenuhi syarat untuk memilih"<<endl;
    }
    else
    {
        cout<<" anda tidak memenuhi syarat untuk
memilih"<<endl;
    }
    return 0;
}
```



int umur; → Artinya mendeklarasikan variable bernama umur dengan tipe data integer.
 cin>> umur; → artinya nilai dari variable umur akan diinputkan oleh user.
 If(umur >= 13 && umur < 18) -> Artinya menggunakan fungsi IF untuk menentukan jika variable umur lebih dari sama dengan 13 dan umur kurang dari 18 jika benar maka nilai akan tercetak "anda adalah remaja" jika tidak maka akan tercetak "anda bukan remaja" dimana disini variable umur akan dilakukan pengecekan 2x. pertama variable umur akan dilihat apakah umur lebih dari sama dengan 13? Lalu akan dicek kembali apakah variable umur kurang dari 18? Dengan logika penghubung && (dan) maka akan dicek apakah keduanya benilai benar.
 If(umur >= 18) → Artinya menggunakan fungsi IF untuk menentukan jika variable umur lebih dari sama dengan 18 jika benar maka nilai akan tercetak "anda memenuhi syarat untuk memilih" jika tidak maka akan tercetak "anda tidak memenuhi syarat untuk memilih"

Listing:

```
#include<iostream>
using namespace std;
int main()
{
    int bilangan;
    cout<<"Masukan bilangan : ";cin>>bilangan;
    if(((bilangan % 5) == 0) && ((bilangan % 2) == 0))
    {
        cout<<"Kelipatan 5 dan 2"<<endl;
    }
    else
    {
        cout<<"bukan Kelipatan 5 dan 2"<<endl;
    }
    return 0;
}
```



int bilangan; → Artinya mendeklarasikan variable bernama bilangan dengan tipe data integer.
 cin>> bilangan; → artinya nilai dari variable bilangan akan diinputkan oleh user.
 ((bilangan % 5) == 0) → artinya variable bilangan akan menghitung sisa hasil bagi 5 lalu akan melakukan penyamanaan apakah hasil akhirnya 0 jika hasilnya sama dengan 0 maka dia akan bernilai benar.
 ((bilangan % 2) == 0) → artinya variable bilangan akan menghitung sisa hasil bagi 2 lalu akan melakukan penyamanaan apakah hasil akhirnya 0 jika hasilnya sama dengan 0 maka dia akan bernilai benar.
 ((bilangan % 5) == 0) && ((bilangan % 2) == 0) → artinya ke 2 hasil akan dilakukan pembandingan dengan operator logika and,
 ((bilangan % 5) == 0) && ((bilangan % 2) == 0) || ((bilangan % 2) == 0) → artinya hasil and tadi akan dilakukan pembandingan dengan operator logika or,
 If(((bilangan % 5) == 0) && ((bilangan % 2) == 0) || ((bilangan % 2) == 0)) → artinya jika nilai if ini bersifat benar maka akan muncul kelipatan 5 dan 2 atau 4 jika tidak maka akan muncul bukan kelipatan 5 dan 2 atau 4.

Listing:

```
#include<iostream>
using namespace std;
int main()
{
    int bilangan;
    cout<<"Masukan bilangan : ";cin>>bilangan;
    if(((bilangan % 5) == 0) && ((bilangan % 2) == 0)
    || ((bilangan % 2) == 0))
    {
        cout<<"Kelipatan 5 dan 2 atau 4"<<endl;
    }
    else
    {
        cout<<"bukan Kelipatan 5 dan 2 atau 4"<<endl;
    }
    return 0;
}
```



int bilangan; → Artinya mendeklarasikan variable bernama bilangan dengan tipe data integer.
 cin>> bilangan; → artinya nilai dari variable bilangan akan diinputkan oleh user.
 $((bilangan \% 5) == 0)$ → artinya variable bilangan akan menghitung sisa hasil bagi 5 lalu akan melakukan penyamanaan apakah hasil akhirnya 0 jika hasilnya sama dengan 0 maka dia akan bernilai benar.
 $((bilangan \% 2) == 0)$ → artinya variable bilangan akan menghitung sisa hasil bagi 2 lalu akan melakukan penyamanaan apakah hasil akhirnya 0 jika hasilnya sama dengan 0 maka dia akan bernilai benar.
 $((bilangan \% 5) == 0) \&\& ((bilangan \% 2) == 0)$ → artinya ke 2 hasil akan dilakukan pembandingan dengan operator logika and,
 $((bilangan \% 5) == 0) \&\& ((bilangan \% 2) == 0) \|\| ((bilangan \% 2) == 0)$ → artinya hasil and tadi akan dilakukan pembandingan dengan operator logika or,
 If $((((bilangan \% 5) == 0) \&\& ((bilangan \% 2) == 0)) \|\| ((bilangan \% 2) == 0))$ → artinya jika nilai if ini bersifat benar maka akan muncul kelipatan 5 dan 2 atau 4 jika tidak maka akan muncul bukan kelipatan 5 dan 2 atau 4.

Listing:

```
#include<iostream>
#include<string.h>
using namespace std;
int main()
{
    string A;//deklarasi variable bertipe string
    int budget;//deklarasi variable bertipe string
    cout<<"Silahkan masukan budget anda :
";cin>>budget;
    cout<<"Silahkan pilih : <<endl;
    cout<<"a. Smartphone"<<endl;
    cout<<"b. Laptop"<<endl;
    cout<<"Silahkan masukan pilihan anda (Masukan a
atau b) : ";cin>>A;
    if(A == "a" && budget >= 2000000)
    {
        cout<<"anda dapat membeli Smartphone"<<endl;
    }
    if(A == "b" && budget >= 5000000)
    {
        cout<<"anda dapat membeli Laptop"<<endl;
    }
    return 0;
}
```



String A; → Artinya mendeklarasikan variable bernama A dengan tipe data string.

int budget; → Artinya mendeklarasikan variable bernama budget dengan tipe data int.

cin>>budget; → artinya nilai dari variable budget akan diinputkan oleh user.

cin>>A; → artinya nilai dari variable A akan diinputkan oleh user.

If(A == “A” && budget >= 2000000) -> Artinya menggunakan fungsi IF untuk menentukan jika variable A sama dengan “a” dan variable budget lebih dari sama dengan 2000000 maka nilai akan tercetak anda dapat membeli smartphone jika tidak maka tidak akan tertulis apa-apa.

If(A == “A” && budget >= 2000000) → Artinya menggunakan fungsi IF untuk menentukan jika variable A sama dengan “a” dan variable budget lebih dari sama dengan 2000000 maka nilai akan tercetak

	anda dapat membeli smartphone jika tidak maka tidak akan tertulis apa-apa.
--	----------------------------------------------------------------------------

C. Soal Latihan/Tugas

1. Buatlah flowchart program untuk mencetak Bilangan Positif jika di inputkan angka lebih besar dari 0, mencetak bilangan negative jika di inputkan bilangan lebih kecil dari 0
2. Buatlah program untuk menghitung total pembelian yang harus dibayarkan oleh seorang pelanggan apabila diketahui kondisi berikut.
Jika jumlah barang > 100 maka diskon = 15% Jika tidak Diskon = 5%
Dimana :
Input : harga barang, jumlah barang
Output : beli, diskon dan total bayar
3. Buatlah program dan flowchart dengan C++ untuk menampilkan bilangan genap dan ganjil.

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*.
BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*.
Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*.
Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 8

CONTROL STATEMENT IF (Lanjutan)

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu mengimplementasikan algoritma menggunakan control statement IF dalam pemecahan suatu masalah dengan berbagai alternatif jawaban yang tersedia dengan mengambil keputusan untuk memilih jawaban yang tepat.

B. Uraian Materi

1. Pendahuluan

Dalam seleksi, program mengeksekusi statemen-statement tertentu tergantung dari kondisi spesifik. Perhatikan tiga statemen berikut:

- a. if (skor lebih besar atau sama dengan 90)
 nilai adalah A
- b. if (jam kerja kurang dari atau sama dengan 40)
 gaji = gajiPerJam * jam Kerja
 else
 gaji = (gajiPerJam * 40) + (1.5 * (gajiPerJam * jamKerja – 40))
- c. if (suhu lebih dari 70 derajat dan tidak hujan)
 mainGolf

Ketiga statemen di atas merupakan contoh kondisional yang akan dieksekusi hanya jika kondisi terpenuhi. Kondisi menjadi terpenuhi jika ia dievaluasi menjadi **true**.

2. Jenis-Jenis Decission If

a. Perintah if

Bentuk umum perintah if:

if (ekspresi)

statemen

Ekspresi : Merupakan sebuah kondisi logikal yang digunakan sebagai pembuat keputusan untuk memutuskan apakah mengeksekusi statemen atau tidak.

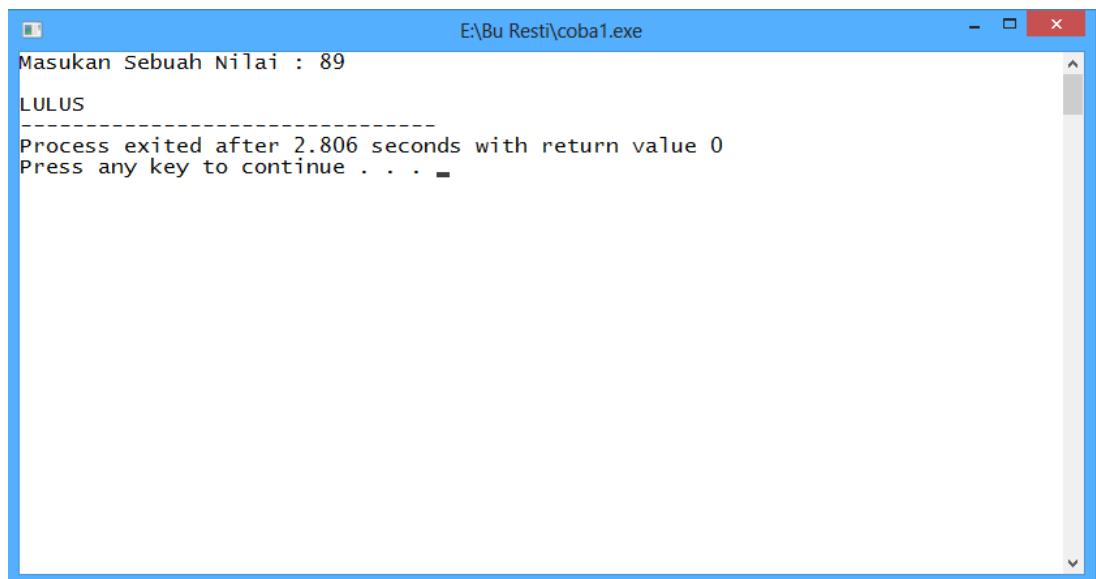
Jika nilai dari **ekspresi** adalah **true**, maka statemen akan dieksekusi dan jika nilai dari **ekspresi** adalah **false** maka statemen tidak akan dijalankan.

Contoh Program 1 :

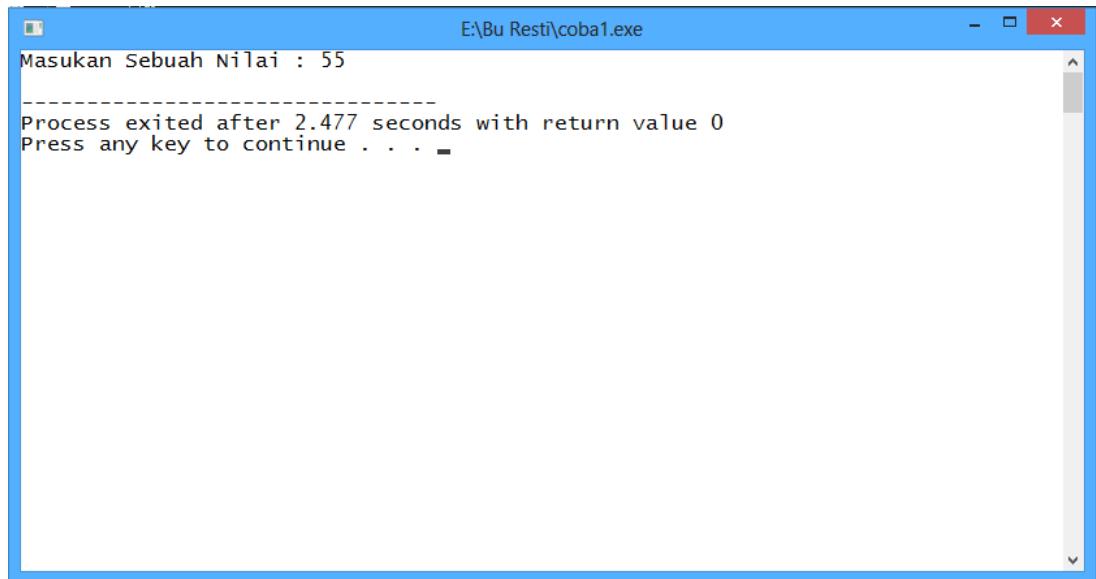
```
#include<iostream>
using namespace std;
main(){
    int nilai;
    cout<<"Masukan Sebuah Nilai : ";
    cin>>nilai;

    if(nilai>69)
        cout<<"\nLULUS";
}
```

Output :

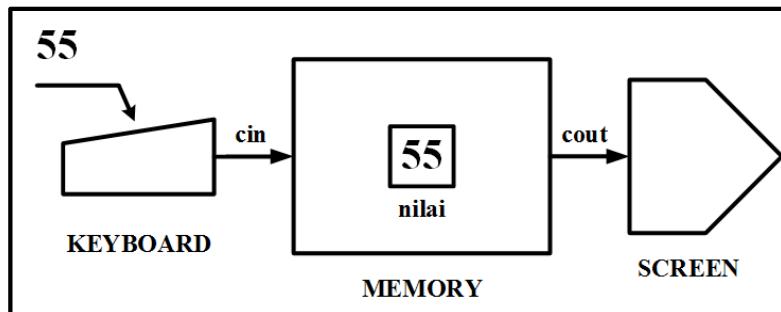
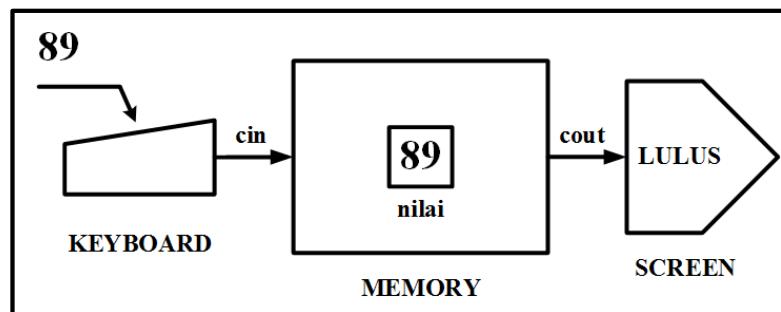


```
Masukan Sebuah Nilai : 89
LULUS
-----
Process exited after 2.806 seconds with return value 0
Press any key to continue . . .
```



```
Masukan Sebuah Nilai : 55
-----
Process exited after 2.477 seconds with return value 0
Press any key to continue . . .
```

Penjelasan :



Source code dan *output* di atas merupakan contoh penggunaan perintah if, dimana ketika nilai yang diinputkan sesuai dengan ekspresi (>69) maka compiler akan memproses statemen pertama dan mencetak kata LULUS pada output-an. Tetapi jika nilai tidak sesuai dengan ekspresi (<69) tidak ada kata yang tercetak pada output dikarenakan statemen yang terdapat pada program hanya satu (tidak memiliki statemen bernilai false).

b. Perintah if ... else

Bentuk umum perintah if ... else:

```
if (ekspresi)
    statemen1
else
```

Ekspresi : Merupakan sebuah ekspresi logikal yang digunakan sebagai pembuat keputusan untuk memutuskan apakah mengeksekusi statemen atau tidak.

Ada banyak situasi pemrograman dimana di dalamnya harus memilih salah satu dari dua alternatif. Untuk memilih salah satu dari dua alternatif, C++ menyediakan statemen if ... else yang diawali dengan kata kunci **if**, diikuti dengan sebuah ekspresi logikal yang diapit oleh sepasang kurung, diikuti dengan sebuah statemen, diikuti dengan katakunci **else**, dan diikuti dengan statemen kedua. Jika nilai dari **ekspresi** adalah **true**, maka **statemen1** akan dieksekusi. Jika nilai dari **ekspresi** adalah **false**, maka **statemen2** yang dieksekusi.

Contoh Program 2 :

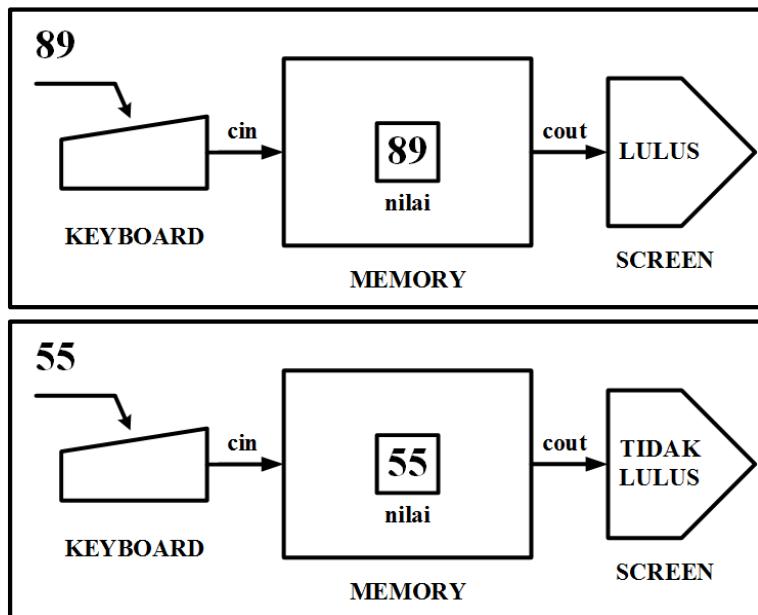
```
#include<iostream>
using namespace std;
main(){
    int nilai;
    cout<<"Masukan Sebuah Nilai : ";
    cin>>nilai;

    if(nilai>69)
        cout<<"\nLULUS";
    else
        cout<<"\nTIDAK LULUS"
}
```

Output:

```
E:\Bu Resti\coba1.exe
Masukan Sebuah Nilai : 89
LULUS
-----
Process exited after 0.9294 seconds with return value 0
Press any key to continue . . .
```

```
E:\Bu Resti\coba1.exe
Masukan Sebuah Nilai : 55
TIDAK LULUS
-----
Process exited after 2.48 seconds with return value 0
Press any key to continue . . .
```

Penjelasan:

Source code dan *output* di atas merupakan contoh penggunaan perintah **if ... else**, dimana ketika nilai yang diinputkan sesuai dengan ekspresi (>69) maka *compiler* akan memproses statemen pertama dan mencetak kata **LULUS** pada output-an. Jika nilai tidak sesuai dengan ekspresi (<69), *compiler* akan memproses statemen kedua dan mencetak kata **TIDAK LULUS** (karena pada program terdapat statemen bernilai false).

c. Perintah *if* bersarang (*Nested-If*)

Bentuk umum *if* bersarang:

```

if (ekspresi1)
    if (ekspresi2)
        statemen1
    else
        statemen2
  
```

Merupakan sebuah ekspresi logikal yang digunakan
Ekspresi1,2 : sebagai pembuat keputusan untuk memutuskan
apakah mengeksekusi statemen atau tidak.

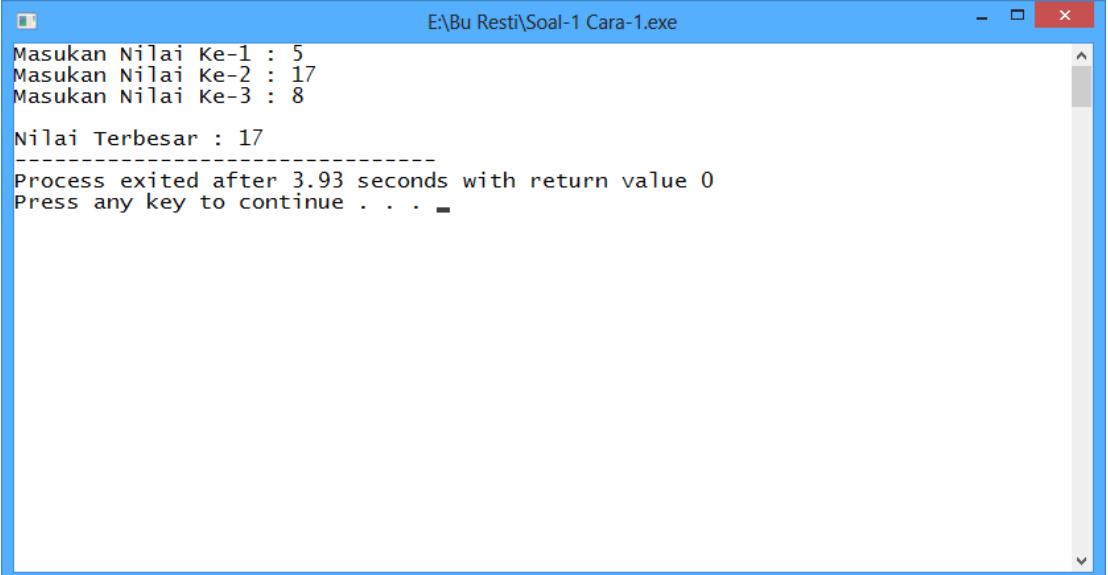
Jika suatu statemen kendali berada di dalam statemen kendali yang lain, statemen kendali tersebut dikatakan bersarang di dalam statemen kendali yang lain. Jika nilai dari **ekspresi1** adalah **true**, maka proses pengecekan akan dilanjutkan ke **ekspresi2**. Jika nilai **ekspresi2** adalah **true**, maka **statemen1** akan dieksekusi. Jika nilai **ekspresi2** adalah **false**, maka **statemen2** akan dieksekusi. Jika nilai **ekspresi1** adalah **false**, maka **statemen3** akan dieksekusi.

Contoh Program 3:

Susun program dalam bahasa C++ untuk menginput tiga buah bilangan bulat (integer), dimana ketiga buah bilangan tersebut dianggap bernilai tidak sama, kemudian mencetak salah satu bilangan yang nilainya terbesar.

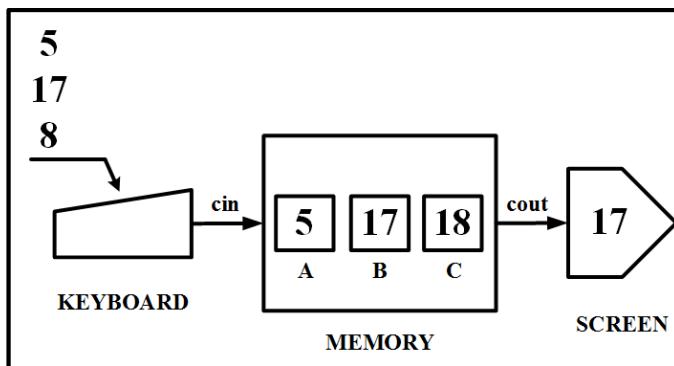
```
#include<iostream>
using namespace std;
main() {
    int a, b, c;
    cout<<"Masukan Nilai Ke-1 : "; cin>>a;
    cout<<"Masukan Nilai Ke-2 : "; cin>>b;
    cout<<"Masukan Nilai Ke-3 : "; cin>>c;

    if(a>b)
        if(a>c)
            cout<<"\nNilai Terbesar : "<<a;
        else
            cout<<"\nNilai Terbesar : "<<c;
    else
        if(b>c)
            cout<<"\nNilai Terbesar : "<<b;
        else
            cout<<"\nNilai Terbesar : "<<c;
}
```

Ouput:

```
Masukan Nilai Ke-1 : 5
Masukan Nilai Ke-2 : 17
Masukan Nilai Ke-3 : 8

Nilai Terbesar : 17
-----
Process exited after 3.93 seconds with return value 0
Press any key to continue . . .
```

Penjelasan:

Dari *source code* dan *output* diatas, untuk nilai integer yang diinputkan adalah 5, 17, 8. Berdasarkan program yang telah dibuat untuk menentukan nilai terbesar dari tiga nilai yang diinputkan, *compiler* akan melakukan pengecekan dari ketiga nilai tersebut mana yang nilai terbesar. Setelah proses pengecekan program akan mencetak hasilnya yakni 17, karena 17 merupakan nilai terbesar jika dibandingkan dengan 5 dan 8.

Contoh Program 4:

Susun program untuk menginput tiga buah bilangan bulat (misal a, b dan c dimana a< >b< >c< > a), kemudian mencetak ketiga nilai tersebut urut nilai dari terkecil ke besar.

```
#include<iostream>
using namespace std;
main(){
    int a, b, c;
    cout<<"Masukan Nilai ke-1 : "; cin>>a;
    cout<<"Masukan Nilai ke-2 : "; cin>>b;
    cout<<"Masukan Nilai ke-3 : "; cin>>c;
    if(a<b)
        if(b<c)
            cout<<"\nUrutan Nilai Dari yang Terkecil :
" <<a<< " <<b<< " <<c<< " ;
        else
            if(a<c)
                cout<<"\nUrutan Nilai Dari yang Terkecil :
" <<a<< " <<c<<
                                " <<b<< " ;
            else
                cout<<"\nUrutan Nilai Dari yang Terkecil
: " <<c<< " <<a<<
                                " <<b<< " ;
        else
            if(a<c)
                cout<<"\nUrutan Nilai Dari yang Terkecil :
" <<b<< " <<a<< " <<c<< " ;
            else
                if(b<c)
                    cout<<"\nUrutan Nilai Dari yang Terkecil
: " <<b<< " <<c<<
                                " <<a<< " ;
                else
                    cout<<"\nUrutan Nilai Dari yang Terkecil :
" <<c<< " <<b<<
                                " <<a<< " ;
    }
}
```

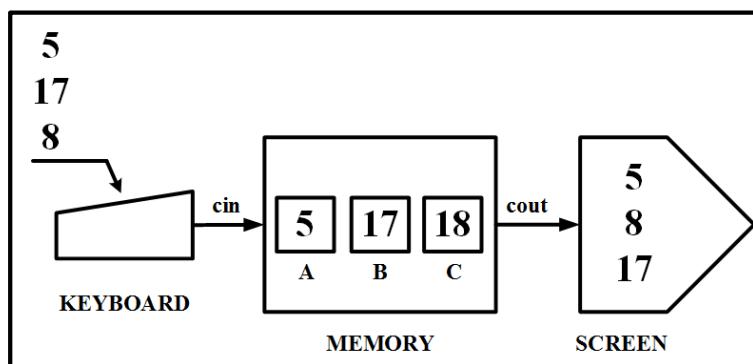
Output

```

E:\Bu Resti\Soal-3 Cara-1.exe
Masukan Nilai ke-1 : 5
Masukan Nilai ke-2 : 17
Masukan Nilai ke-3 : 8
Urutan Nilai Dari yang Terkecil : 5 8 17
Process exited after 5.876 seconds with return value 0
Press any key to continue . . .

```

Penjelasan

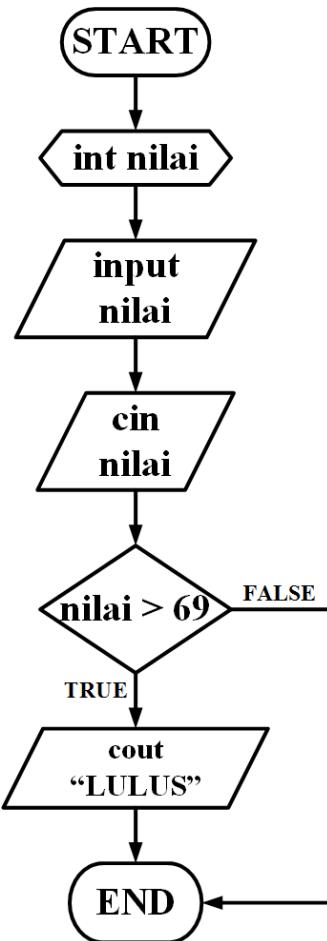


Berdasarkan *source code* dan *output* di atas, *nested-if* atau *if bersarang* dapat digunakan dalam pembuatan program untuk mengurutkan nilai dari yang terkecil hingga terbesar. Setelah nilai diinputkan secara acak (sebagai contoh, nilai yang diinputkan adalah 5, 17 dan 8), *compiler* akan melakukan proses pengurutan nilai-nilai tersebut. **Statemen pertama** akan dieksekusi apabila nilai a lebih kecil dari nilai b dan nilai b lebih kecil dari nilai c. **Statemen kedua** akan dieksekusi apabila nilai a lebih kecil dari nilai b, nilai c lebih kecil dari nilai b dan nilai a lebih kecil dari nilai c. **Statemen ketiga** akan dieksekusi apabila nilai a lebih kecil dari nilai b, nilai c lebih kecil dari nilai b dan nilai c lebih kecil dari nilai a. **Statemen keempat** akan dieksekusi apabila nilai b lebih kecil dari nilai a dan nilai a lebih kecil dari nilai c. **Statemen kelima**

akan dieksekusi apabila nilai b lebih kecil dari nilai a, nilai c lebih kecil dari nilai a dan nilai b lebih kecil dari nilai c. **Statemen keenam** akan dieksekusi apabila nilai b lebih kecil dari nilai a, nilai c lebih kecil dari nilai a dan nilai c lebih kecil dari nilai b.

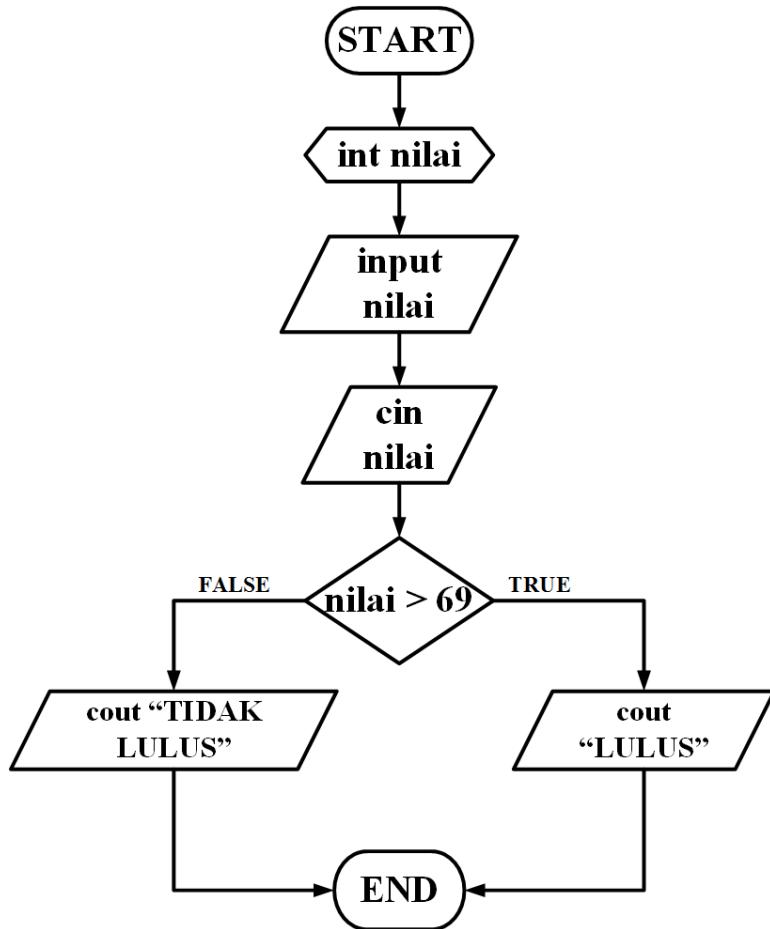
3. Alur Kerja Perintah IF

a. Perintah *if*



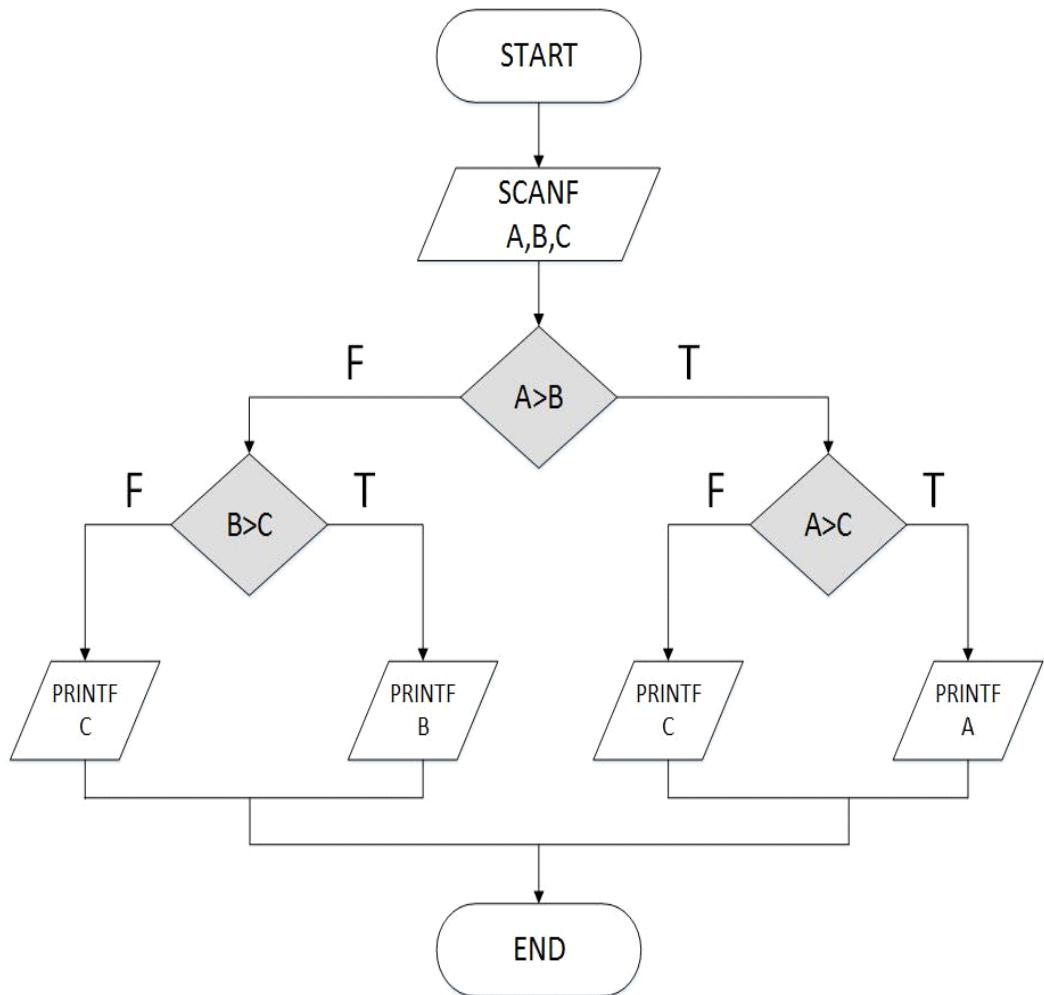
- 1) Mulai
- 2) Deklarasi varibel nilai
- 3) Input nilai
- 4) Program membaca inputan nilai
- 5) Cek nilai yang diinput
 - a) Jika nilai lebih besar dari 69, cetak kata “LULUS”
 - b) Jika nilai lebih kecil dari 69, program berakhir
- 6) Selesai.

b. Perintah *if ... else*

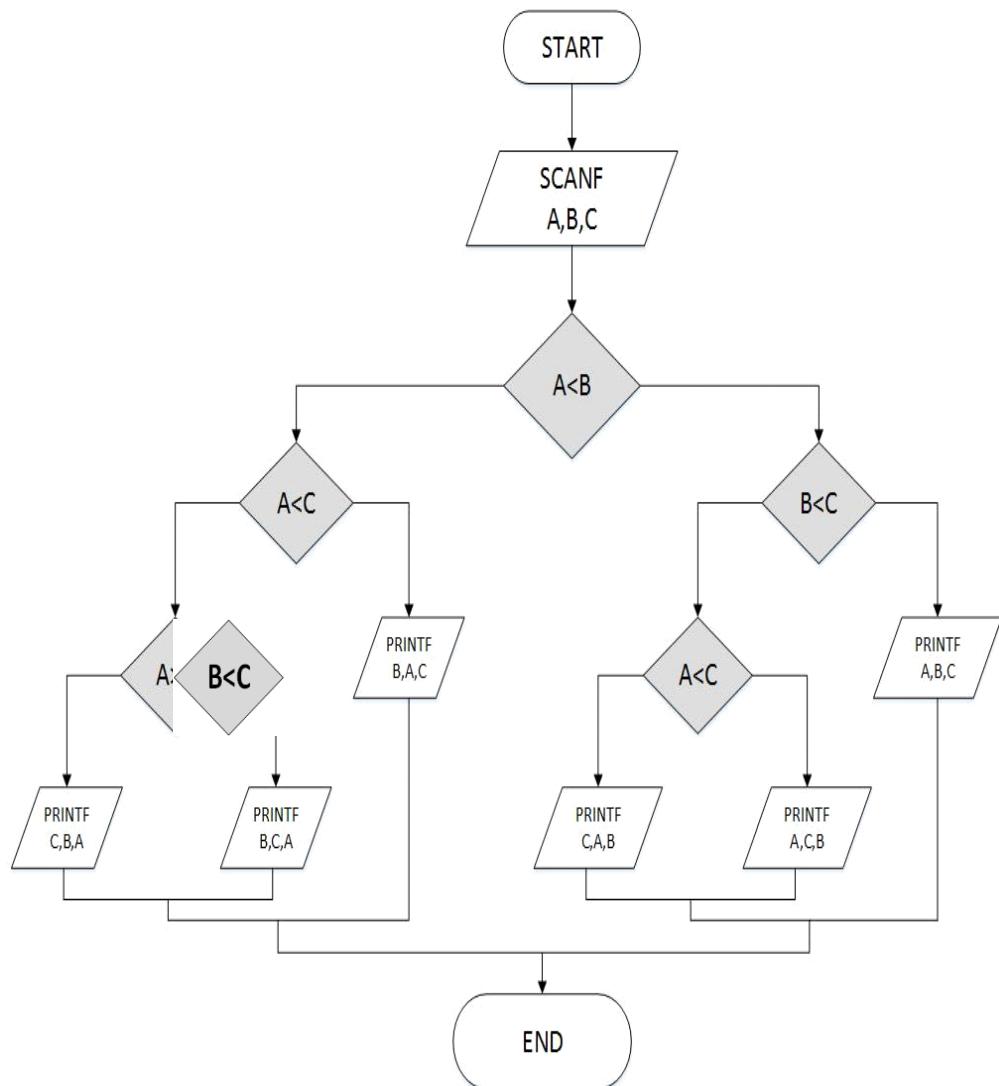


- 1) Mulai
- 2) Deklarasi variabel nilai
- 3) Input nilai
- 4) Program membaca inputan nilai
- 5) Cek nilai yang diinput
 - a) Jika nilai lebih besar dari 69, cetak kata "LULUS"
 - b) Jika nilai lebih kecil dari 69, cetak kata "TIDAK LULUS"
- 6) Selesai

c. Perintah *if* bersarang (*nested-if*)



- 1) Mulai
- 2) Program membaca inputan (nilai A, B, C)
- 3) Cek nilai yang diinput
 - a) Jika A lebih besar dari B dan A lebih besar dari C, cetak A
 - b) Jika A lebih besar dari B dan A lebih kecil dari C, cetak C
 - c) Jika A lebih kecil dari B dan B lebih besar dari C, cetak B
 - d) Jika A lebih kecil dari B dan B lebih kecil dari C, Cetak C
- 4) Selesai.



- 1) Mulai
- 2) Program membaca inputan (nilai A, B, C)
- 3) Cek nilai yang diinput
 - a) Jika A lebih kecil dari B dan B lebih kecil dari C, cetak A, B, C
 - b) Jika A lebih kecil dari B, B lebih besar dari C, dan A lebih besar dari C, cetak A, C, B
 - c) Jika A lebih kecil dari B, B lebih besar dari C, dan A lebih besar dari C, cetak C, A, B
 - d) Jika A lebih besar dari B dan B lebih kecil dari C, cetak B, A, C
 - e) Jika A lebih besar dari B, B lebih besar dari C, dan C lebih kecil dari B, cetak B, C, A
 - f) Jika A lebih besar dari B, B lebih besar dari C, dan C lebih besar dari B, cetak C, B, A
- 4) Selesai

C. Soal/Latihan

1. Gambarkan flowchar dari penggalan program berikut :

a.

```
int T=0, A=5, B=10
if(A>B){
    T = T+A;
    T = T+B;
    cout<<"T : "<<T;
}else{
    T = T - A;
    T = T - A;
}
```

b.

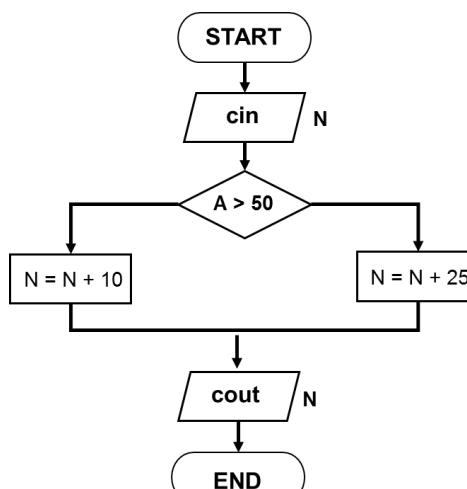
```
int T = 50, A = 10, B = 5;
if (A<B){
    cout<<T;
    A *= T;
    cout<<A;
}else{
    T /= B;
    cout<<T;
}
```

2. Susun algoritma dan program atau penggalan program untuk menginput 3 buah bilangan yang masing-masing menyatakan panjang sebuah garis.

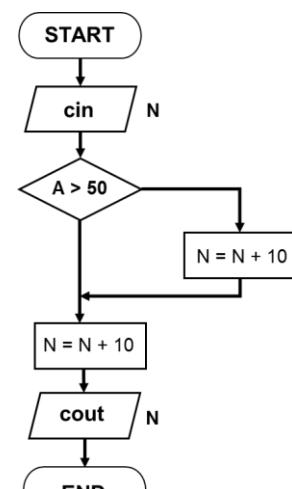
Kemudian periksa ketiga buah garis tersebut apakah dapat membentuk sebuah segitiga atau tidak. Bila ketiga buah garis tersebut dapat membentuk sebuah segitiga, maka cetak perkataan “SEGI TIGA”, sebaliknya bila ketiga garis tersebut tidak mungkin membentuk sebuah segitiga, makacetak perkataan: “BUKAN SEGITIGA”

3. Buat / tulis program dengan bahasa C++ untuk menyatakan algoritma yang di gambarkan flowchar berikut:

a.



b.



4. Untuk program dengan algortima pada soal no 3, Apa yang akan tercetak jika nilai N yang di masukkan:
- 25
 - 50
 - 75

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Kristanto, A. (2003). *Algoritma & Pemograman Dengan C++*. Yogyakarta: Graha Ilmu.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal danC*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mitra Wacana Media.

PERTEMUAN 9
CONTROL STATEMENT MENGGUNAKAN NESTED IF SWITCH dan
LOGICAL OPERATOR

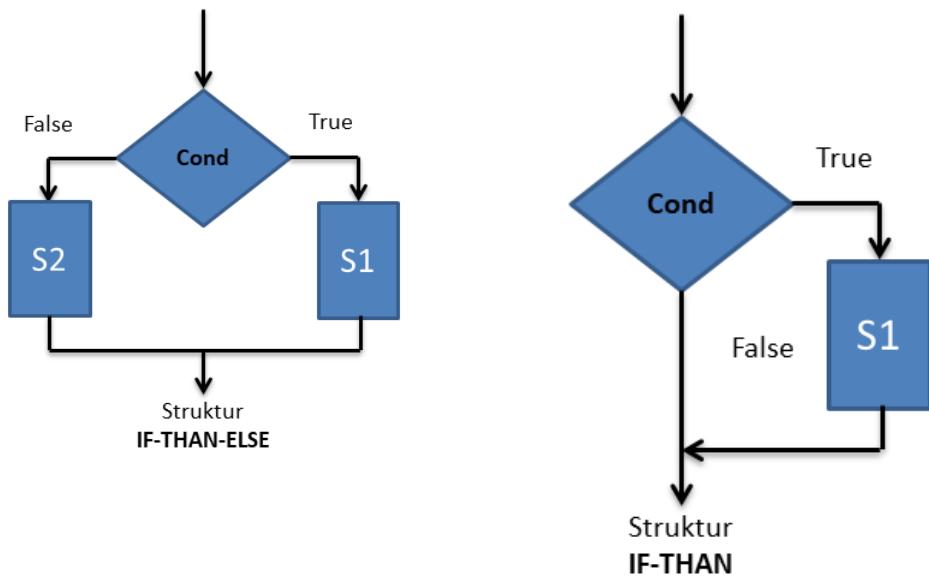
A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu mahasiswa mengerti tentang penggunaan control statement menggunakan NESTED if dan switch dan Logical Operator dapat menggunakannya dalam pemrograman

B. Uraian Materi

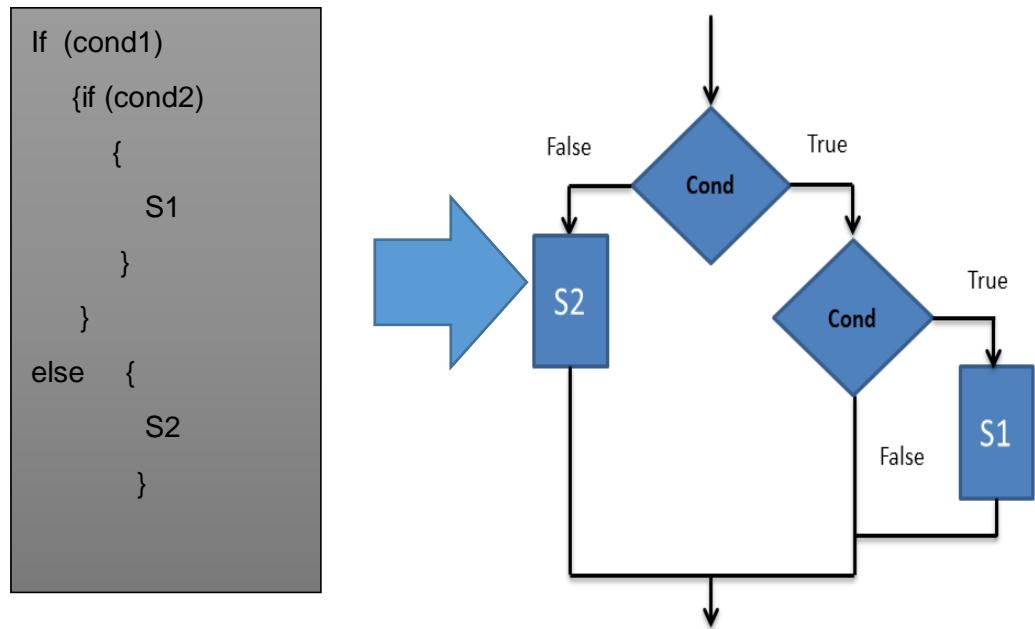
1. Nested if

Perhatikan kembali struktur **IF-THEN-ELSE** dan **IF-THEN** statement seperti yang sudah diterangkan melalui flowchart sebagai berikut:

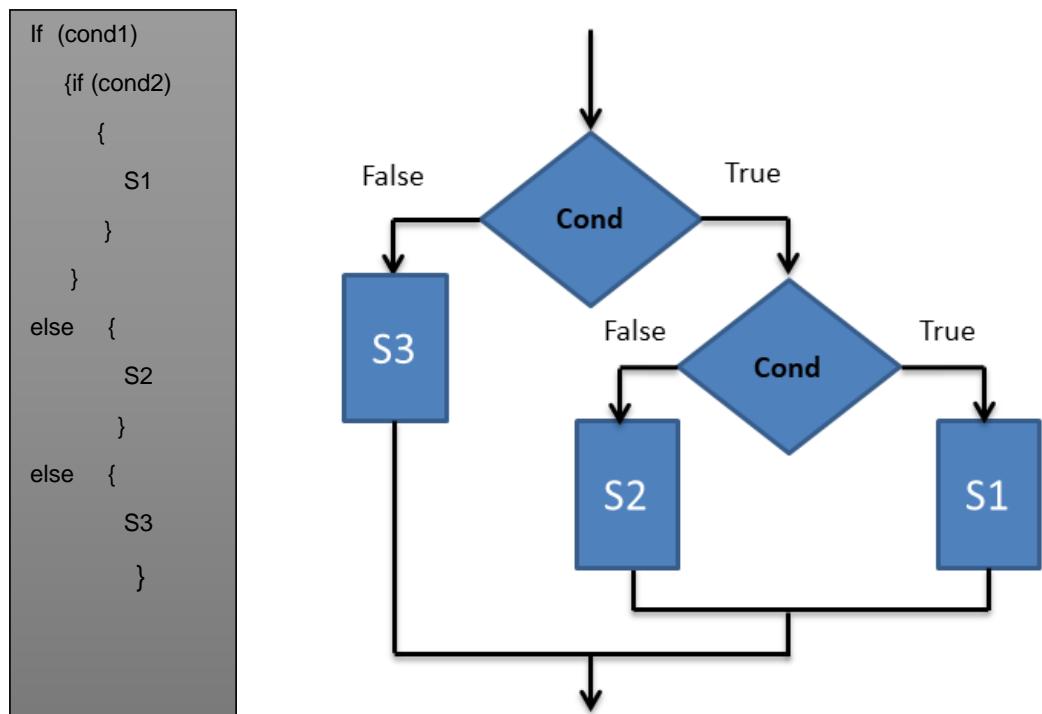


Dari ilustrasi gambar, terlihat **S** adalah satu atau sekelompok statement. Didalam kelompok **S** mungkin terdapat statement **IF** sehingga terjadi IF secara berjenjang atau secara tersarang yang biasa disebut **Nested If** (nest=sarang).

Contoh -1

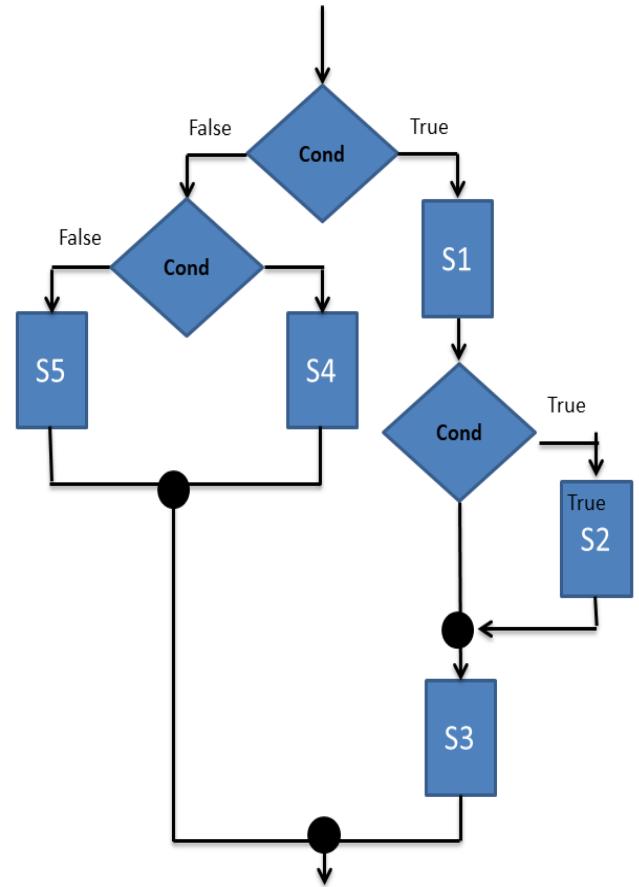


Contoh -2

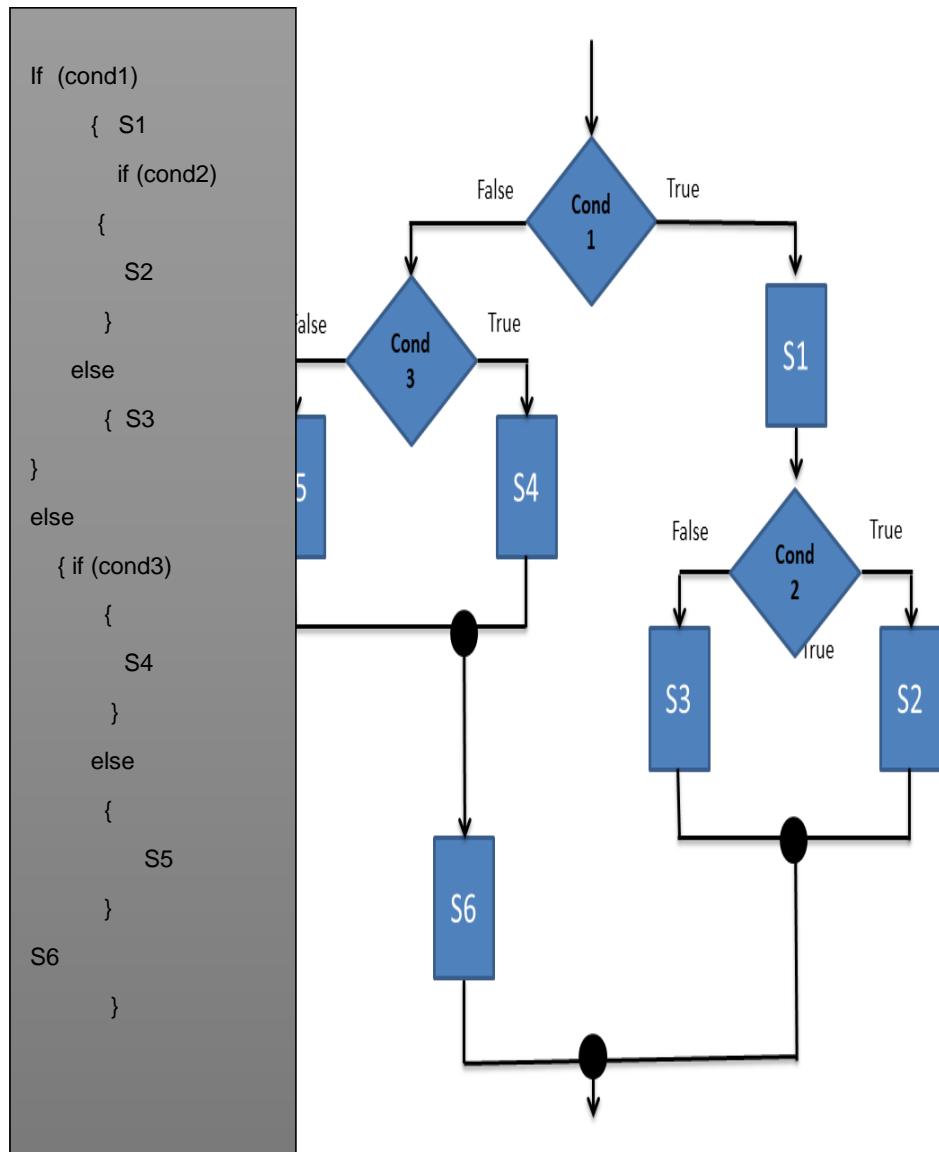


Contoh -3

```
If (cond1)
  {
    S1
    if (cond2)
    {
      S2
    }
    S3
  }else
  {
    if (cond3)
    {
      S4
    }
    else {
      S5
    }
  }
```

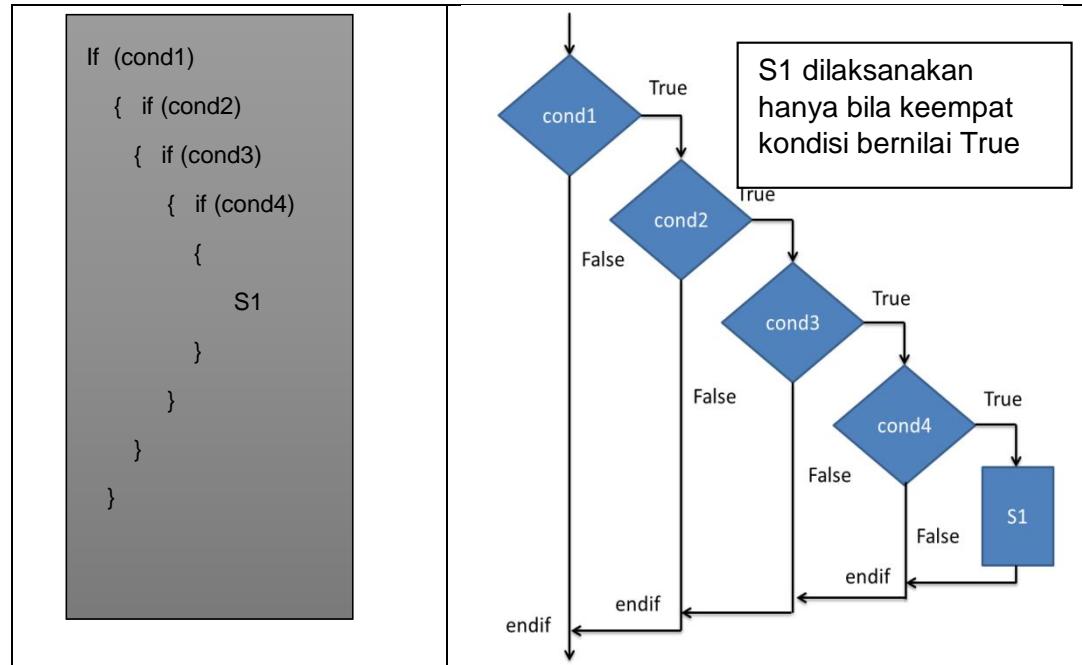


Contoh -4

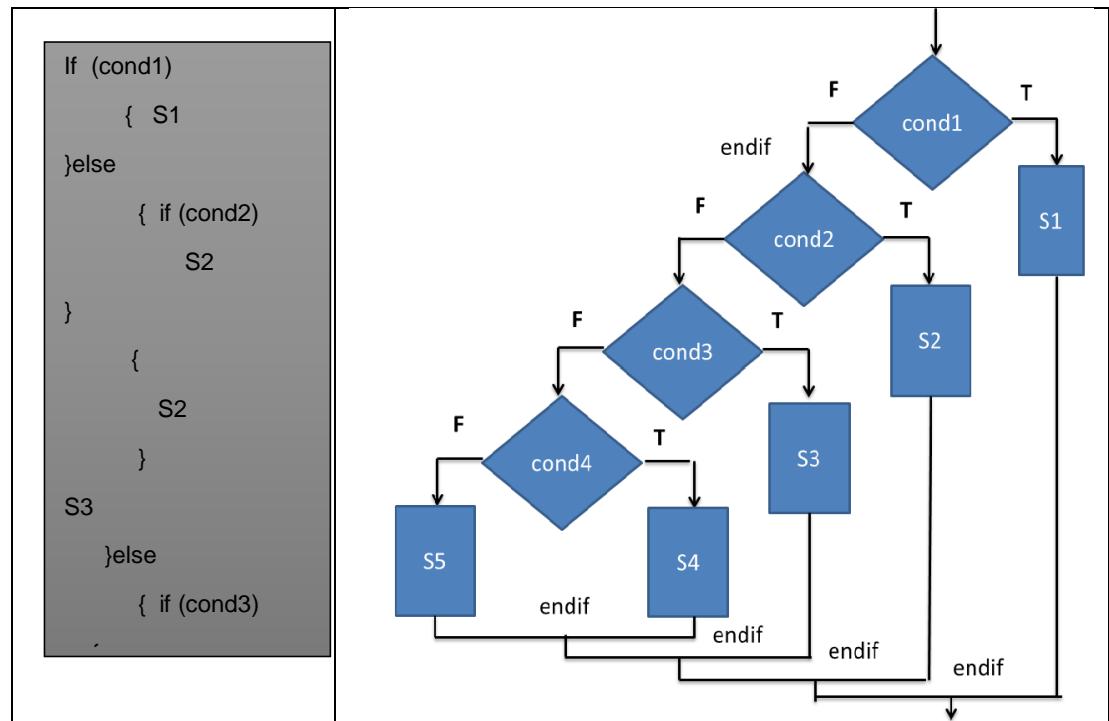


Perhatikan posisi letak Endif dalam flowchart. Posisi ini penting untuk menganalisa aliran terutama untuk nested if yang komplek atau untuk proses pengulangan yang bersifat rekursif.

Contoh -5



Contoh -6



Struktur pada contoh 5 dan 6 terlihat nested if berjenjang seperti anak tangga sehingga sering juga disebut struktur if-else-if ladder.

2. Milti Condition dan Logical Operator

Terkadang satu kondisi saja tidak cukup untuk menentukan suatu syarat sehingga diperlukan dua atau lebih kondisi. Untuk menggabung kondisi-kondisi tersebut digunakan operator yang disebut logical operator.

Ada 3 macam logical operator yaitu:

- a. NOT → !
- b. AND → &&
- c. OR → ||

Penjelasannya sebagai berikut.

a. Operator Not

Operator not walaupun sifatnya logical tapi beberapa buku literatur tidak memasukkannya kedalam logical operator. Operator **NOT** bukan untuk menggabung 2 buah kondisi, tapi bekerja sebagai pembalik nilai logika **TRUE** menjadi **FALSE** dan sebaliknya sehingga sering disebut **Unary Operator**.

Contoh:

Misal A=6 dan B= 3

Condition	Nilai
A == B	False
A > B	True
A < B	False
A >= B	True
A <= B	False
A != B	True

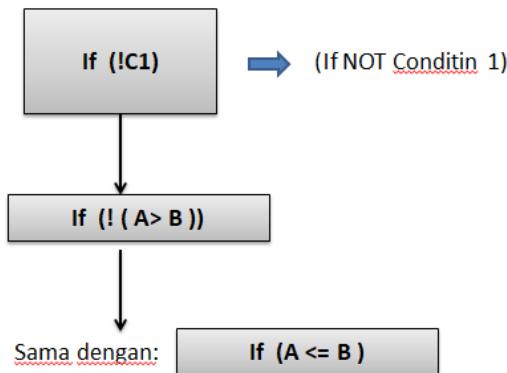


Not(Condition)	Nilai
!(A == B)	True
!(A > B)	False
!(A < B)	True
!(A >= B)	False
!(A <= B)	True
!(A != B)	False

Sehingga

Penulisan	Sama maksudnya
!(A == B)	A != B
!(A > B)	A <= B
!(A < B)	A >= B
!(A >= B)	A < B
!(A <= B)	A > B
!(A != B)	A == B

Catatan:



b. Operator AND

Operator AND menggabung dua buah kondisi menjadi satu nilai sedemikian rupa akan bernilai TRUE hanya bila kedu kondisi digabungkan bernilai TRUE, Atau dengan kata lain, kedua syarat harus terpenuhi.

Pemakaian dalam program C/C++ atau Java

If (cond1 && cond2)

Atau

If (cond1) && (cond2)

Contoh penulisan dalam bahasa C/C++ ata Java:

- 1) If (KodeSex == 1 && Umur <=25)
- 2) If (Nilai >= 60 && Nilai < 70)

Perhatikan tabel yang memperlihatkan nilai gabungan dua buah kondisi (cond1 dan cond2) yang digabung dengan operator AND. Tabel seperti ini biasa disebut Truth Table atau Tabel Kebenaran.

Cond1	Cond2	Cond1 AND Cond2
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

Hanya jika kondisi 1 dan kondisi 2 bernilai TRUE maka hasilnya akan TRUE

Contoh:

Bila nilai A=6, B = 3, C= 7 dan D= 4 maka:

If (A > B && C > D)	Bernilai TRUE
true true	
If (A > B && B > D)	Bernilai FALSE
true false	
If (A > C && B > D)	Bernilai FALSE
false false	
If (A > B && B > D)	Bernilai FALSE
true false	

- c. operator OR menggabungkan dua buah kondisi menjadi satu nilai sedemikian rupa akan bernilai TRUE dengan cukup apabila salah satu kondisi bernilai TRUE. Dan akan bernilai FALSE hanya jika kedua kondisi bernilai FALSE.

Pemakaian dalam program C/C++ atau Java

```
If ( cond1 || cond2)
```

Atau

```
If ( cond1) || (cond2)
```

Contoh penulisan dalam bahasa C/C++ ata Java:

3) If (Status == 1 || Umur >=17)
4) If (Nilai1 >= 60 || Nilai2 >= 70)

Syarat sebagai pemilih: harus sudah menikah (status=1) atau umur tidak kurang dari 17 tahun(17 tahun keatas)

Truth Table

Cond1	Cond2	Cond1 OR Cond2
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

Contoh:

Bila nilai A=6, B = 3, C= 7 dan D= 4 maka:

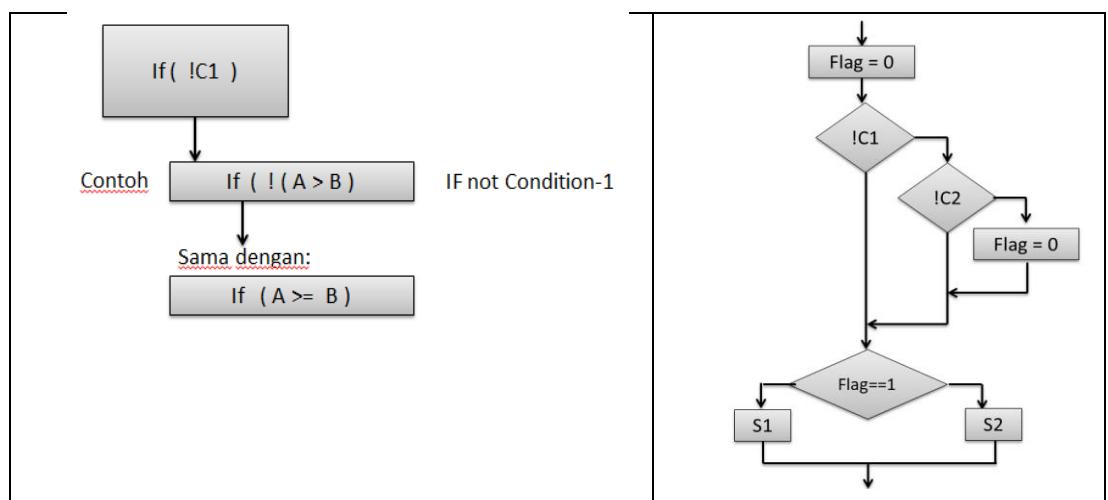
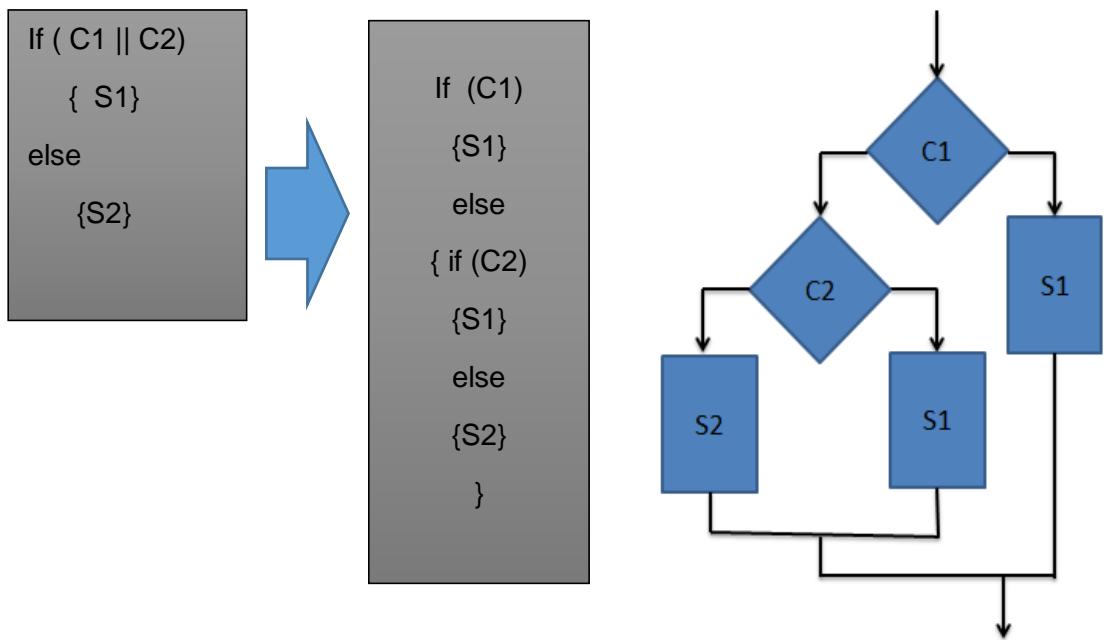
If (A > B C > D)	Bernilai TRUE
true true	
If (A > B B > D)	Bernilai TRUE
true false	
If (A > C B > D)	Bernilai FALSE

false If (A > B B > D) true	false true	Bernilai TRUE
----------------------------------------	---------------	---------------

3. Konversi Multi Condition menjadi Nested If

Berikut ini contoh cara mengkonversi statement if multi condition kedalam bentuk nested if.

Contoh:



4. Contoh Program Sederhana Menggunakan Nested If Dan Multi Condition

Berikut contoh soal dengan tiga kondisi:

Susunlah suatu program menggunakan bahasa pemrograman yaitu bahsa C++ untuk menginput tiga bilangan bulat (integer). Dimana ketiga buah bilangan tersebut bernilai tidak sama kemudian mencetak salah satu bilangan yang dinyatakan terbesar!

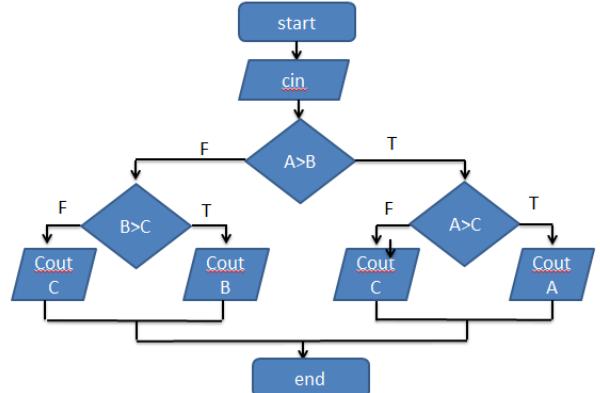
```
#include <iostream>
using namespace std;
int main()
{
    int pertama, kedua, ketiga;
    cout<<"Input bilangan pertama : ";
    cin>>pertama;
    cout<<"input bilangan kedua : ";
    cin>>kedua;
    cout<<"input bilangan ketiga : ";
    cin>>ketiga;
    if (pertama >kedua)
    {
        if(pertama > ketiga)
        cout <<pertama;

        else
        cout<<ketiga;
    }

    if(kedua > ketiga)
    cout<<kedua;
    else
    cout<<ketiga;

    cout<<endl;
    return 0;
}
```

```
C:\Users\user\Documents\Project2.exe
Input bilangan pertama : 11
input bilangan kedua : 12
input bilangan ketiga : 7
12
-----
Process exited after 8.892 seconds with return value 0
Press any key to continue . . .
```

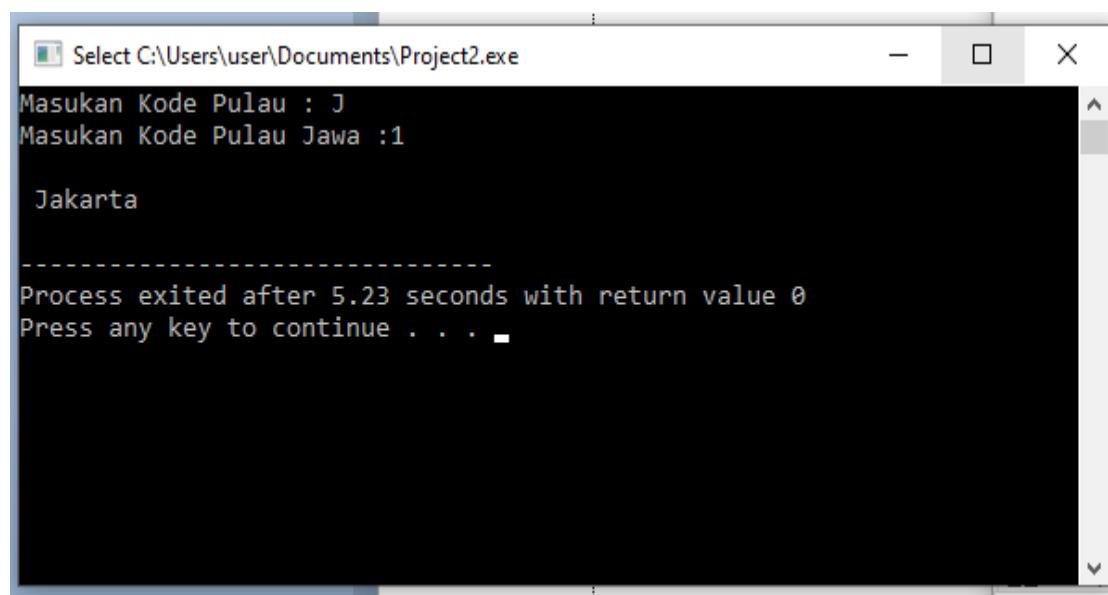


5. Switch & Case Berjenjang

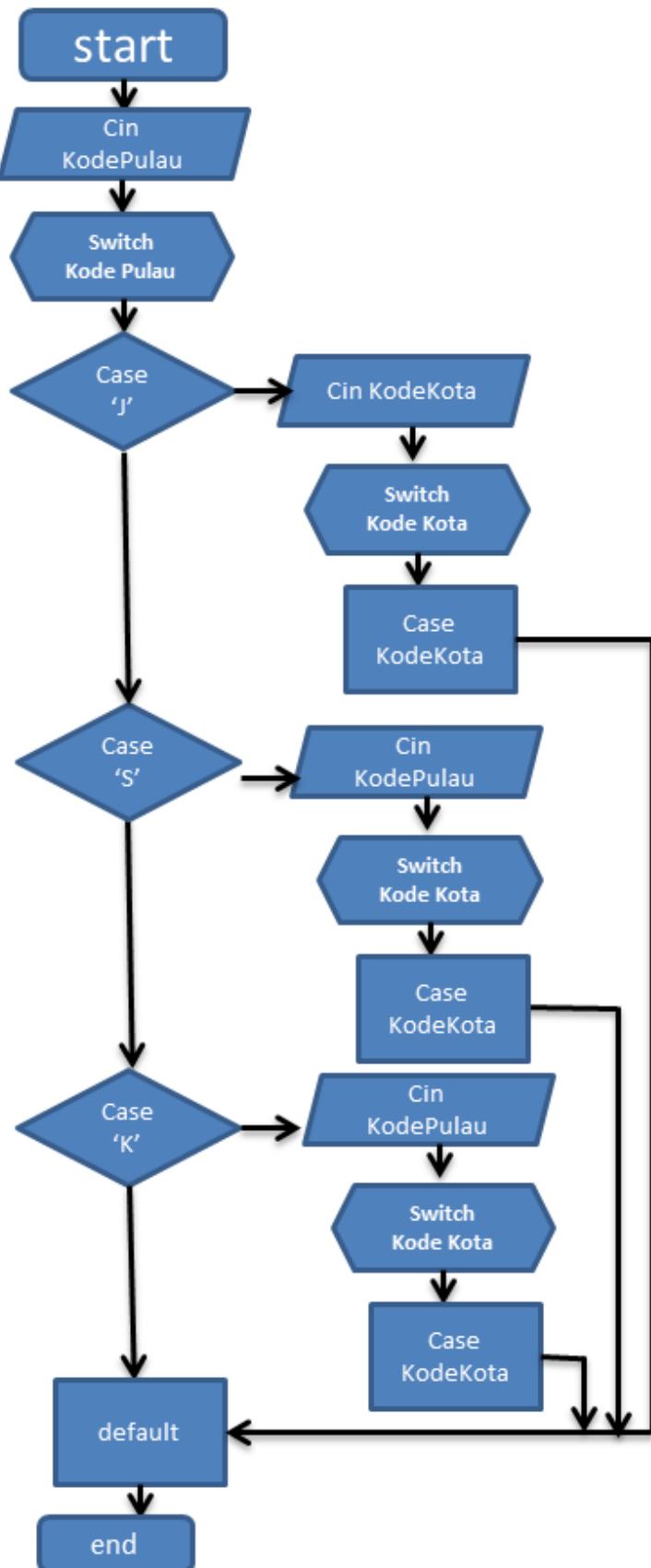
Contoh:

```
#include <iostream>
using namespace std;
main()
{
    char KodePulau; int KodeKota;
    // char adalah variabel yang isinya karakter diikuti dengan nama variabel
    // yang akan dibuat
    cout<<"Masukan Kode Pulau : ";
    //sourch diatas berfungsi untuk menampilkan " Masukan Kode Pulau"
    cin>>KodePulau;
    // berfungsi untuk inputan kode pulau sesuai dengan case masing-masing
    switch(KodePulau)
    {
        case 'J':
            cout<<"Masukan Kode Pulau Jawa :";
            cin>>KodeKota;
            switch (KodeKota)
            {
                case 1: cout<<"\n Jakarta"; break;
                case 2: cout<<"\n Surabaya"; break;
                case 3: cout<<"\n Jakarta"; break;
                case 4: cout<<"\n Jakarta"; break;
                default: cout<<"Kode Kota diPulU AWA SALAH";
            }
            break;
        case 'S':
            cout<<"Masukan Kode Pulau Sumatra ";
            cin>>KodeKota;
            switch (KodeKota)
            {
                case 1: cout<<"\n Medan"; break;
                case 2: cout<<"\n Palembang"; break;
                case 3: cout<<"\n Padang"; break;
                case 4: cout<<"\n Bengkulu"; break;
                default: cout<<"Kode Kota di Sumatra Salah";
            }
            break;
        case 'K':
            cout<<"Masukan Kode Pulau Kalimantan ";
            cin>>KodeKota;
            switch (KodeKota)
            {
                case 1: cout<<"\n Banjarmasin"; break;
                case 2: cout<<"\n Pontianak"; break;
                case 3: cout<<"\n Lolong"; break;
                default: cout<<"Kode Kota di Kalimantan Salah";
            }
            break;
        default: cout<<"Kode Kota di Pulau Salah";
    }

    cout<<endl;
    return 0;
}
```



```
Select C:\Users\user\Documents\Project2.exe
Masukan Kode Pulau : J
Masukan Kode Pulau Jawa :1
Jakarta
-----
Process exited after 5.23 seconds with return value 0
Press any key to continue . . .
```



C. Soal Latihan

Susunlah program untuk menginput empat buah bilangan bulat kemudian mencetak salah satu bilangan yang nilainya terbesar, serta buat flowchartnya!

D. Referensi

- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 10

PERULANGAN MENGGUNAKAN FOR, WHILE

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa dapat:

1. Menjelaskan dan membedakan konsep dasar penggunaan perintah-perintah perulangan (looping) pada bahasa pemrograman.
2. Mengerti tentang penggunaan control statement menggunakan for dan while
3. Dapat menggunakan dan mengaplikasikannya dalam program.

B. Uraian Materi

Pengulangan atau perintah looping dalam bahasa pemrograman yang biasa kita kenal dipakai untuk melakukan perintah secara berulang-ulang apabila suatu kondisi sudah terpenuhi atupun sebaliknya. Dalam praktek di pemrograman computer, pengulangan biasa dipakai untuk mengulang proses pada perhitungan, mengulang proses input data dan banyak lagi terkait dalam proses pengulangan. Perintah dalam bahasa pemrograman yang akan kita bahas kali ini adalah pengulangan for dan while

1. For

a. Definisi

FOR adalah salah satu Jenis loop dalam bahasa C++. Instruksi ini dipakai apabila kita sudah mengetahui berapa kali perulangan pernyataan akan dilakukan.. contoh dibawah adalah contoh format perulangan FOR:

```
for loop.  
for (initialization; kondisi; rubah_kondisi)  
{  
    Pernyataan1;  
    Pernyataan2;  
    // tempat banyaknya pernyataan
```

Pernyataan loop atau perulangan bias juga dinyatakan sebagai berikut:

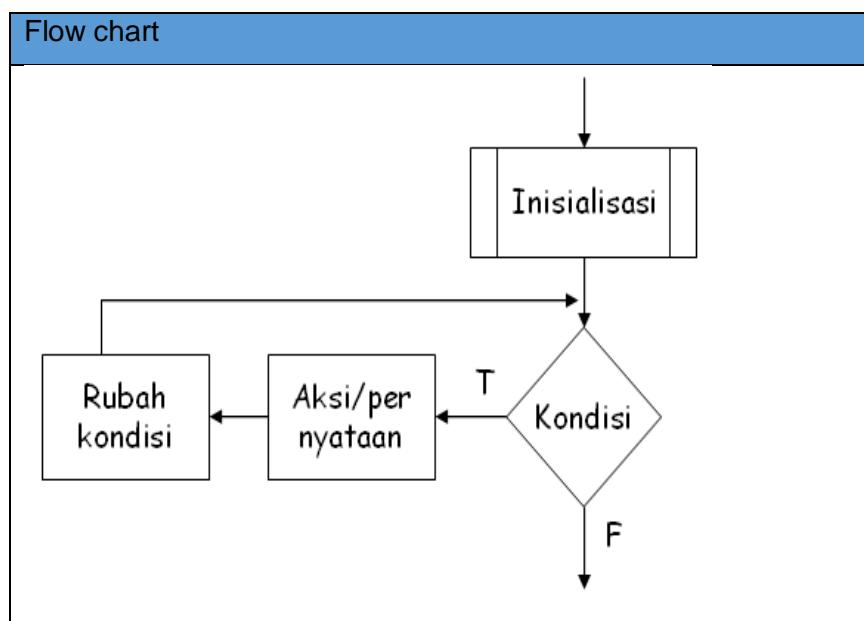
```
for (inisialisasi; kondisi; rubah_kondisi)
```

pernyataan;

Atau ditulis seperti dibawah ini::

```
for(init;kondisi;)
{
    pernyataan;
    pernyataan;
    rubah_kondisi;
}
init;
for(;kondisi;)
{
    pernyataan;
    pernyataan;
    rubah_kondisi;
}
```

Adapun flowchart yang menjelaskan mengenai format penulisan dasar for pada table-tabel diatas adalah sebagai berikut :

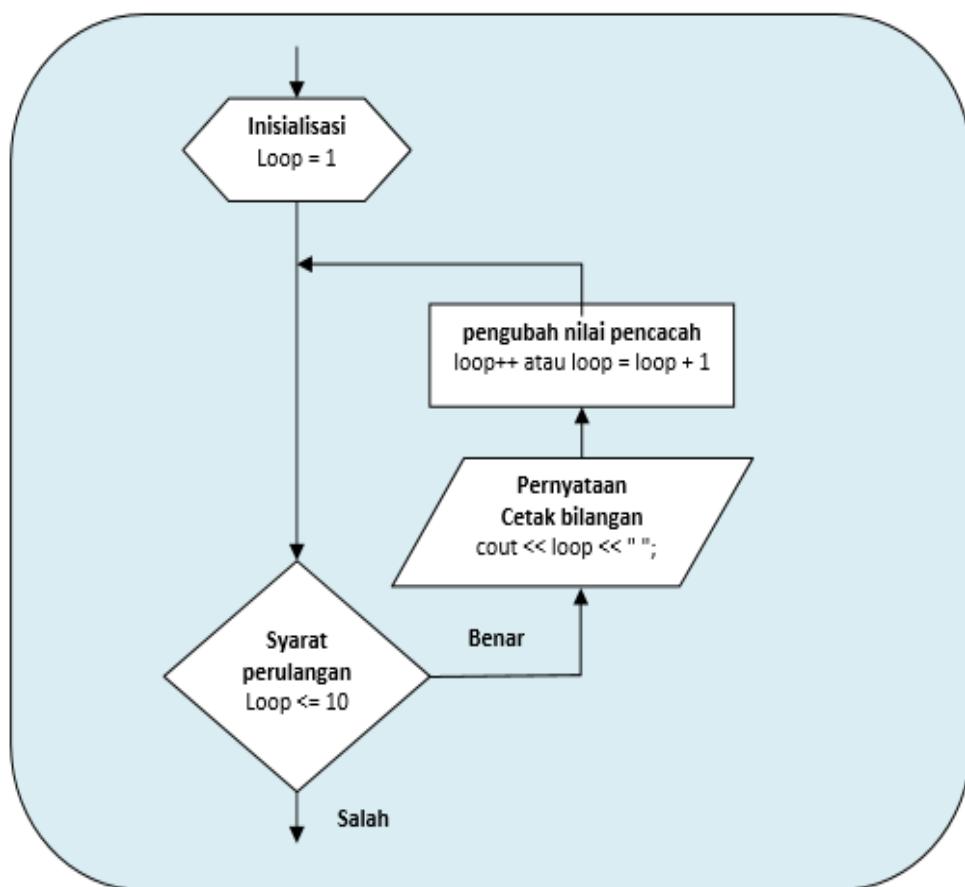


Pernyataan FOR dari flowchar diatas digambarkan sebagai berikut::

- 1) inisialisasi: pernyataan ini biasa dioakai untuk menginisialisasikan untuk pengendalian pengulangan tersebut..

- 2) dalam pengulangan, kondisi dipakai untuk menyatakan dan menentukan apakah proses perulangan akan dilanjutkan atau tidak..
- 3) Ubah_kondisi : tahap ini biasanya digunakan untuk merubah variable dalam pengatur atau pengendali perulangan.

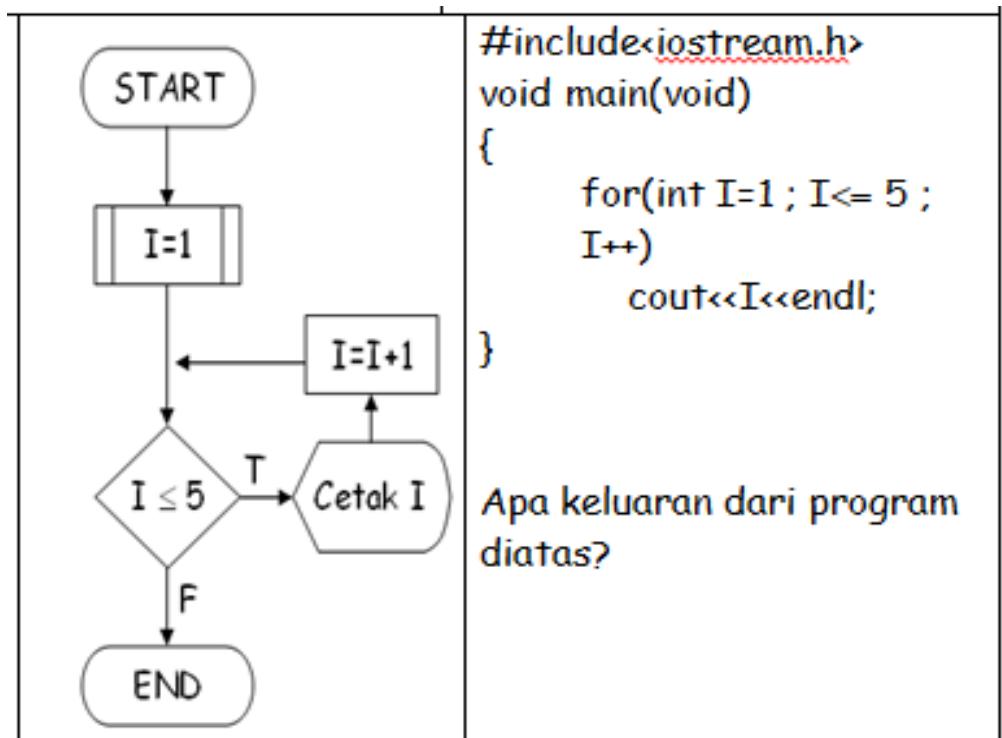
Berikut adalah contoh lain diagram alir dari pengulangan FOR sebagai berikut :



Dalam pengulangan, perubahan nilai bisa terjadi secara increment dan decrement. Ungkapan ungkapan yang ada dudalam FOR tersebut harus diisahkan oleh semicolon atau titik koma. Didalam FOR pernyataan yang ada bisa dalam bentuk tunggal dan jamak.. apabila pernyataan didalam perulangan tersebut sifatnya banyak maka harus kita kelompokan dan dikurung menggunakan kurung kurawal. Dibawah ini contoh pernyataan yang menggunakan kurung kurawal.

```
for (inisialisasi; initialization; kondisi; rubah_kondisi)
{
    case_1;
    case_2;
    .....
    case_n;
//Tempat Banyaknya case
}
```

Dibawah ini merupakan contoh diagram alir dalam pernyataan FOR :

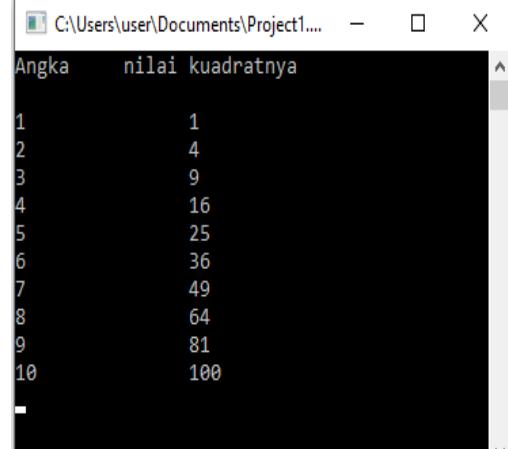
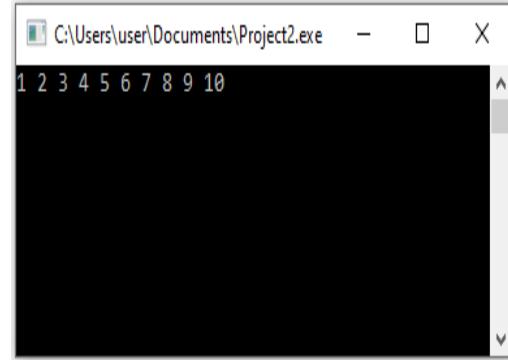


Analisis perkembangan nilai i dari diagram alir dan program sederhana diatas adalah sebagai berikut :

nilai i	Cond i<=5	Display	Nilai i baru (Setelah I=I+1)
1	True	1	2
2	True	2	3
3	True	3	4
4	True	4	5

5	True	5	6
6	False	Out from Loop	

Contoh program sederhana dari perulangan:

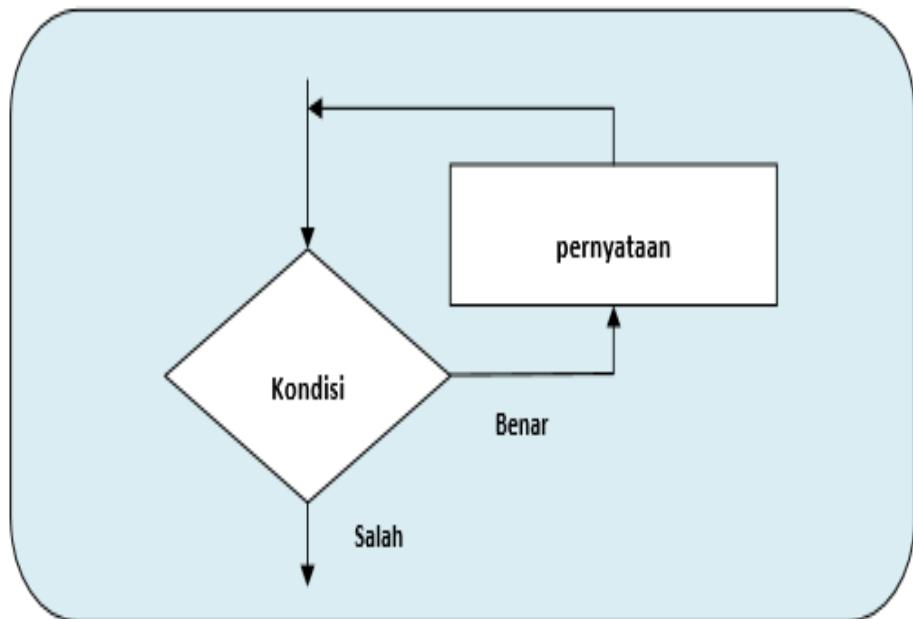
<pre>#include <conio.h> #include <iostream> using namespace std; int main() { int nilai; cout << "Angka nilai kuadratnya" << endl; cout << "\n"; for (nilai = 1; nilai <= 10; nilai++) cout << nilai << "\t\t" << (nilai * nilai) << endl; getch (); return 0; }</pre>	 <table border="1"> <thead> <tr> <th>Angka</th> <th>nilai kuadratnya</th> </tr> </thead> <tbody> <tr><td>1</td><td>1</td></tr> <tr><td>2</td><td>4</td></tr> <tr><td>3</td><td>9</td></tr> <tr><td>4</td><td>16</td></tr> <tr><td>5</td><td>25</td></tr> <tr><td>6</td><td>36</td></tr> <tr><td>7</td><td>49</td></tr> <tr><td>8</td><td>64</td></tr> <tr><td>9</td><td>81</td></tr> <tr><td>10</td><td>100</td></tr> </tbody> </table>	Angka	nilai kuadratnya	1	1	2	4	3	9	4	16	5	25	6	36	7	49	8	64	9	81	10	100
Angka	nilai kuadratnya																						
1	1																						
2	4																						
3	9																						
4	16																						
5	25																						
6	36																						
7	49																						
8	64																						
9	81																						
10	100																						
<pre>#include <conio.h> #include <iostream> using namespace std; int main() { int pengulangan; for (pengulangan = 1; pengulangan <= 10; pengulangan++) cout << pengulangan << " "; getch (); return 0; }</pre>	 <table border="1"> <thead> <tr> <th>1 2 3 4 5 6 7 8 9 10</th> </tr> </thead> </table>	1 2 3 4 5 6 7 8 9 10																					
1 2 3 4 5 6 7 8 9 10																							

2. While

Selanjutnya didalam perulangan yang menggunakan pernyataan while yaitu perulangan yang perulangannya mirip dengan pernyataan pada FOR yang sudah kita bahas diatas. Untuk perulangan pada pernyataan for digunakan untuk menyatakan perulangan yang sudah kita ketahui sebelumnya, berapa kali perulangan tersebut akan terjadi. Sedangkan pada perulangan while, pada kasus ini perulalangn belum kita ketahui samasekali maka pernyataan yang kita pakai bukannlah pernyataan FOR melainkan pernyataan while. Pernyataan while biasanya dipakai saat suatu kondisi perulangannya yang kita periksa terlebih dahulu sebelum program yang kita buat menjalankan suatu pernyataan. Berikut ini adalah format perulangan WHILE :

```
while (keadaan) pernyataan;  
atau  
while(keadaan)  
{  
    Pernyataan;  
    Pernyataan;  
}
```

Keadaan diatas hanya menggambarkan ketika suatu kondisi dinyatakan benar atau salah.. dan juga operator yang dapat dipakai adalah operator logika dan operator relasi yang nilainya hanya ada dua yaitu benr dan juga salah. Dan operator yang digunakan adalah operator gabungan atau salah satu saja antara operator logika atau operator relasi. Dibawah ini contoh singkat dari pernyataan while dengan satu kondisi.

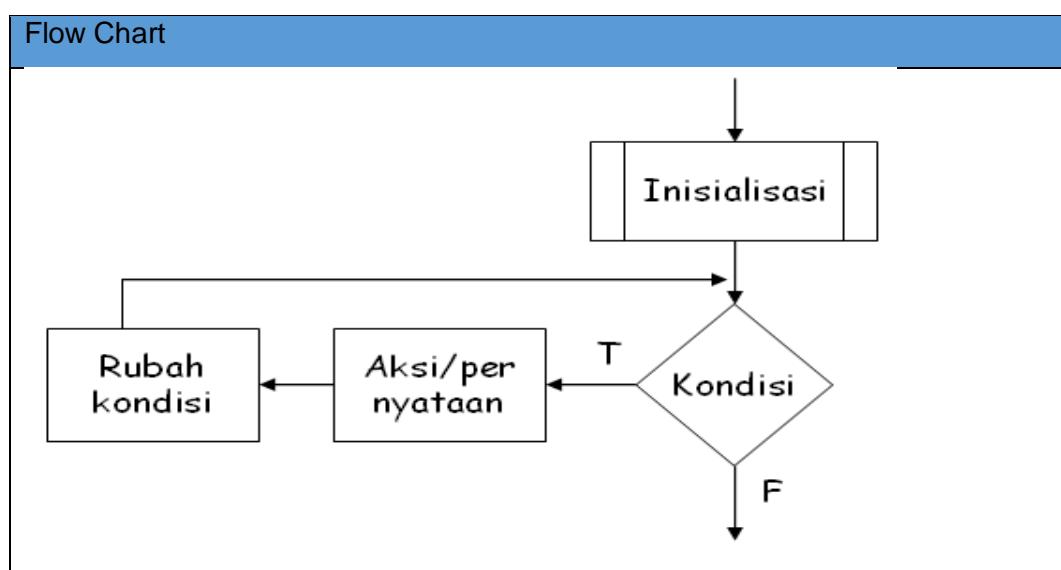


Atau dapat ditulis juga sebagai berikut :

```

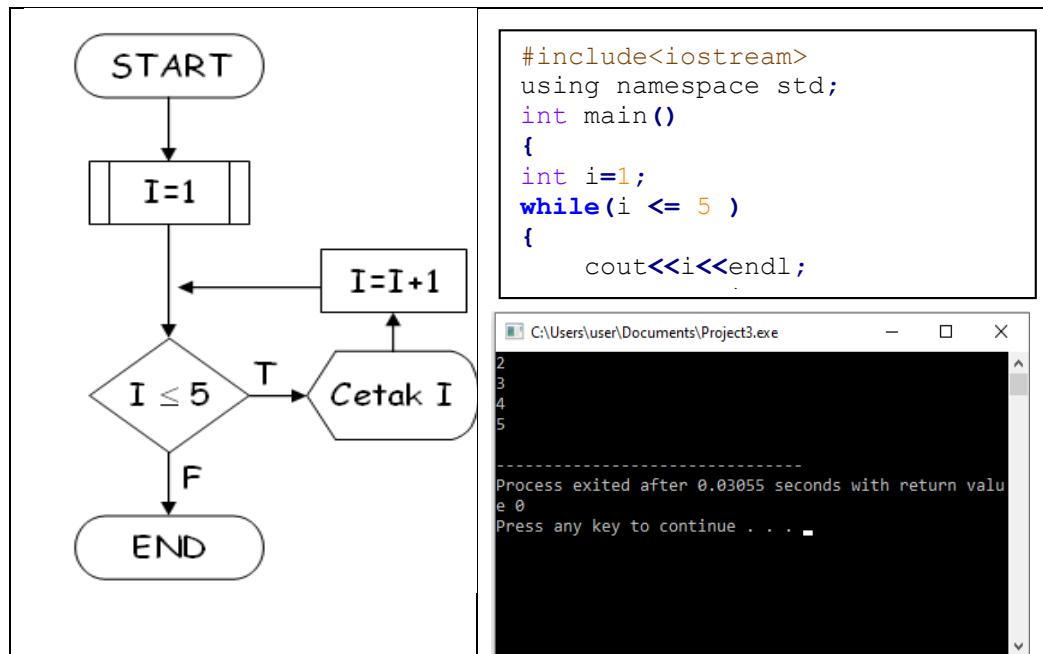
init;
while(kondisi)
{
    pernyataan;
    pernyataan;
    rubah_kondisi;
}
  
```

Adapun flowchart yang menjelaskan mengenai format penulisan dasar for pada tabel-tabel diatas adalah sebagai berikut :



Dari flowchart diatas kita bisa membaca bahwa kondisi/ keadaan diatas adalah dimana kondisi perulangan while pernyataannya lebih dari satu dengan kondisinya benar atau salah. Jika pernyataan yang dijalankan lebih dari satu, maka semua pernyataan tersebut harus digabung menjadi satu menggunakan kurung kurawal.

Dibawah ini adalah contoh flowchart dan juga program sederhana mengenai pernyataan while:



nilai I	Kondisi $I \leq 5$	Tercetak	Nilai I baru (Setelah $I=I+1$)
1	T	1	2
2	T	2	3
3	T	3	4
4	T	4	5
5	T	5	6
6	F	Keluar dari Loop	

Beberapa program sederhana menggunakan while:

```
#include <iostream>
using namespace std;
int main()
{
    int w;

    w = 0;

    while (w<6){
        cout<<"Diulang\n";

        w++;
    }

    return 0;
}
```

The screenshot shows a code editor on the left and a terminal window on the right. The terminal window title is 'C:\Users\user\Documents\Project3.exe'. It displays the output of the program: 'ama Saya Resti Amalia' repeated six times, followed by a separator line '-----', the message 'process exited after 0.1615 seconds with return value 0', and the instruction 'press any key to continue . . .'. The code itself is a simple while loop that prints the string 'ama Saya Resti Amalia' six times.

```
#include <iostream>
using namespace std;
int main()
{
    int w;
    //w hanya contoh
    //variabel, bisa diganti
    //dengan apapun

    //inisialisasi
    //terhadap variabel w
    w = 0;

    while (w<8){
        cout<<"nama
saya Resti Amalia\n";

        // dibaris
        // ini tidak ada statement
        // increment

        //mengakibatkan
        //variabel w terus
        //looping

    }

    return 0;
}
```

The screenshot shows a code editor on the left and a terminal window on the right. The terminal window title is 'C:\Users\user\Documents\Proj...'. It displays the output of the program: 'ama saya Resti Amalia' repeated eight times. The code includes comments explaining the purpose of the variable 'w' and the structure of the loop, including the lack of an increment statement within the loop body.

```
#include <iostream>
using namespace std;
int main(){
    int deret;

    cout<<"Masukan jumlah deret: ";
    cin>>deret;
    cout<<"Jumlah deretnya adalah:";
    cout<<endl;

    while(deret>0){
        cout<<deret<< " ";
        deret--;
    }

    return 0;
}
return 0;
}
```

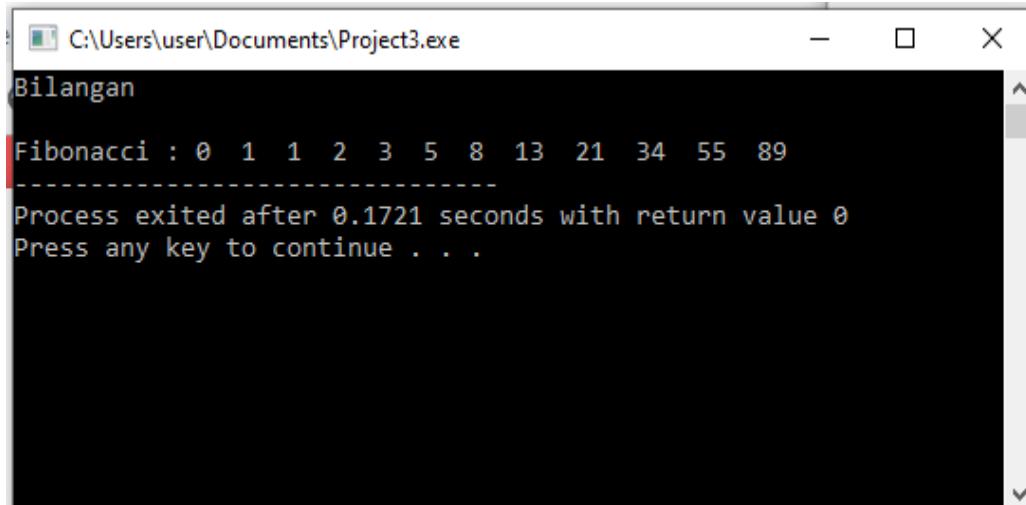
```
Masukan jumlah deret: 12
Jumlah deretnya adalah:
12 11 10 9 8 7 6 5 4 3 2 1
-----
Process exited after 2.589 seconds with return value 0
Press any key to continue . . .
```

C. Soal Latihan :

1. Buatlah program sederhana menggunakan:
 - a. pengulangan for
 - b. pengulangan while dengan tampilan seperti dibawah ini :

```
Bilangan
Ganjil : 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35
Genap : 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 .
-----
Process exited after 0.1517 seconds with return value 0
Press any key to continue . . .
```

- c. Buatlah program menampilkan bilangan Fibonacci menggunakan perintah while, Tampilan Output sebagai berikut ini.



The screenshot shows a command-line interface window. The title bar reads "C:\Users\user\Documents\Project3.exe". The window contains the following text:
Bilangan
Fibonacci : 0 1 1 2 3 5 8 13 21 34 55 89
Process exited after 0.1721 seconds with return value 0
Press any key to continue . . .

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*.
BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*.
Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*.
Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 11
PERULANGAN MENGGUNAKAN DO WHILE DAN NESTED LOOP

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu:

1. Mampu menerapkan perulangan menggunakan bentuk do ... while ...
2. Mampu menerapkan perulangan dalam perulangan atau nested loop
3. Membuat aplikasi sederhana dengan menggunakan do while dan nested loop

B. Uraian Materi

1. Do ... While

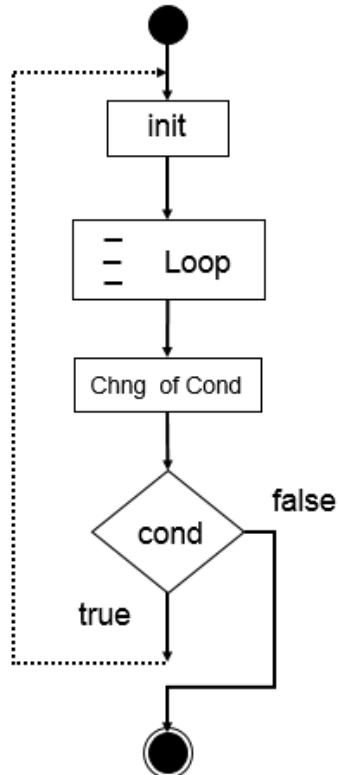
Intruksi do ... while ... merupakan intruksi perulangan yang sama dengan intruksi perulangan while. Perbedaan pada while dan do .. while adalah pada intruksi while kondisi akan di evaluasi atau di uji terlebih dahulu sebelum intruksi perulangan di kerjakan, sedangkan pada intruksi do ... while kondisi akan di evaluasi atau di uji setelah intruksi perulangan dikerjakan.

Syntax Perulangan Menggunakan **do ... while ...**

```
Init;  
do  
{  
- Statement 1  
- Statement 2  
-  
-  
- Statement n  
Change Of Cond;  
}while (cond);
```

Cara kerja:

1. Melakukan inisialisasi awal atau memberikan nilai awal
2. Mengerjakan looping dan mengerjakan
3. Memeriksa kondisi pada intruksi while
 - a. Jika kondisi benar, maka mengerjakan looping kembali
 - b. Jika kondisi salah, maka keluar dari looping

**Gambar 1.** Flowchar do ... while ...

Contoh penggunaan do ... while ...

Listing 11.1 : dowhile.cpp

```

1 #include<iostream>
2 using namespace std;
3 int main()
4 {
5     int i = 1;
6     do
7     {
8         cout<< i;
9         i++;
10    }while(i <= 4);
11 }

```

Output : 1 2 3 4 5



Nilai (i)	tercetak	i++	Kondisi
1	1	2	True
2	2	3	True
3	3	4	True
4	4	5	False

Dimulai dari inisialisasi, di mana nilai i di isi 1, kemudian langsung melaksanakan intruksi yang ada pada looping sehingga di cetak 1, kemudian melaksanakan intruksi (i++) yaitu nilai i tambah 1 dilanjutkan dengan memeriksa kondisi (i <=5) karena nilai i lebih kecil dari 5 maka menghasilkan nilai benar. Intruksi akan terus berulang sampai dengan nilai kondisi bernilai salah baru akan keluar dari looping.

Contoh penggunaan do ... while ...

Listing 11.2 : dowhile.cpp

```

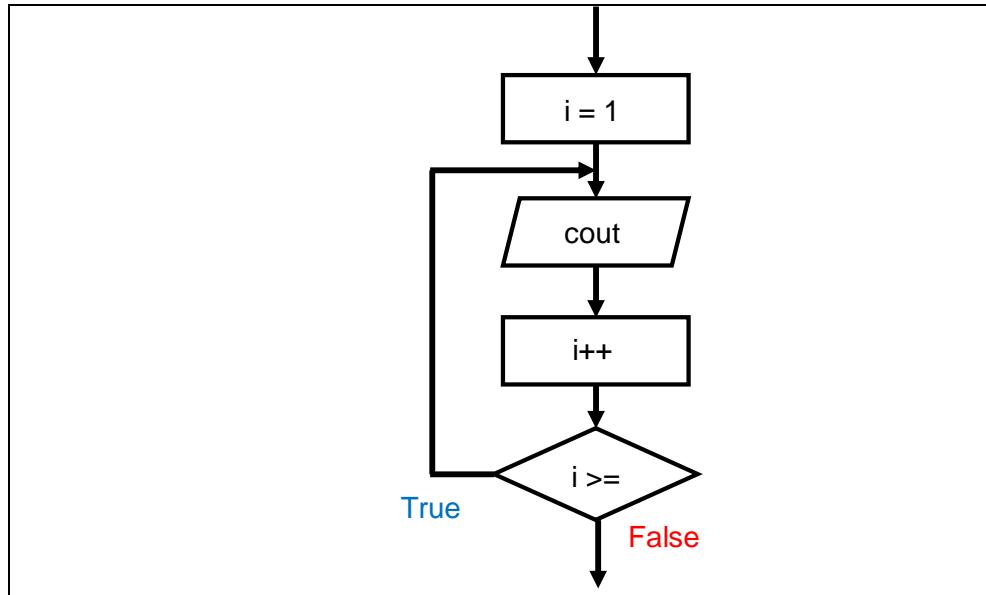
1 #include<iostream>
2 using namespace std;
3 int main()
4 {
5     int i = 1;
6     do
7     {
8         cout<< i;
9         i++;
10    }while(i >= 3);
11 }
```

Output : 1



Nilai (i)	tercetak	i++	Kondisi
1	1	2	False

Dari program diatas minimal akan mencetak 1, karena akan melakukan looping atau intruksi yang ada pada looping akan dikerjakan terlebih dahulu kemudian baru akan memeriksa kondisi yang ada pada intruksi while



2. Loop dalam Loop (Nested Loop)

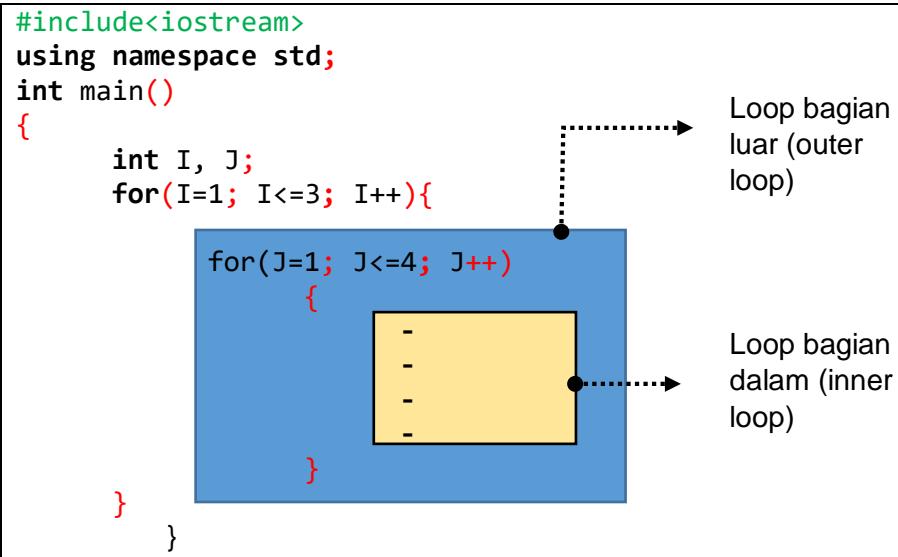
Intruksi loop yang berada dalam intruksi loop yang lain di kenal dengan nested loop atau loop tersarang, untuk ilustrasi perhatikan gambar berikut:

Terdapat dua buah program, yaitu **program A** dan sebuah penggalan **program B**:

Program A	Penggalan Program B
<pre>#include<iostream> using namespace std; int main() { int I, J; for(I=1; I<=3; I++){ - - - Loop - } }</pre>	<pre>for(J=1; J<=4; J++) { - - - Loop - }</pre>
Loop program A akan dikerjakan sebanyak 3 kali	Loop Program B akan dikerjakan sebanyak 4 kali

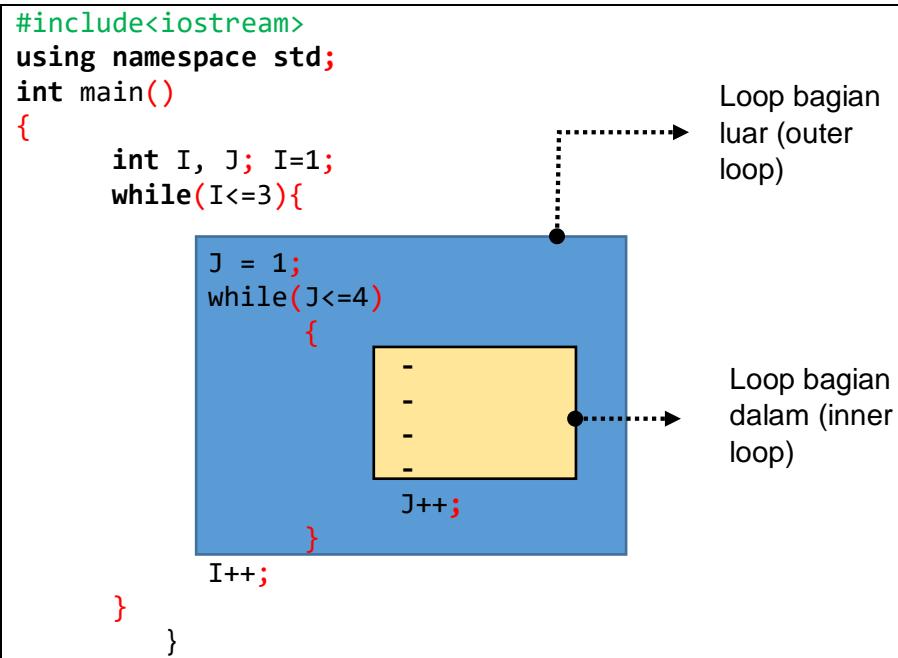
Pada program di atas **program A** terdapat loop menggunakan for yang dikerjakan sebanyak 3 kali, sedangkan pada penggalan **program B** terdapat loop menggunakan for yang dikerjakan sebanyak 4 kali. Apabila penggalan program B dipindahkan atau dimasukkan ke dalam program A mengantikan **kotak 1**, maka akan terbentuk loop di dalam loop, biasanya disebut dengan

NESTED Loop atau **loop tersarang**, menjadi program yang baru sebagai berikut:



Pada program di atas terdapat loop di dalam loop di mana pada outer loop (loop luar) intruksi loop akan dilaksanakan sebanyak 3 kali sedangkan pada inner loop (loop dalam) intruksi loop akan dilaksanakan sebanyak 4 kali, sehingga program di atas jika dijalankan akan melaksanakan loop sebanyak $3 \times 4 = 12$ kali loop.

Jika bentuk nested loop dituliskan menggunakan intruksi while maka akan terbentuk sebagai berikut:

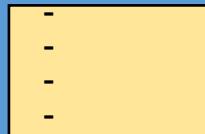


Jika bentuk nested loop dituliskan menggunakan gabungan while dan for maka akan terbentuk nested loop sebagai berikut:

While untuk outer loop dan for untuk inner loop

```
#include<iostream>
using namespace std;
int main(){
    int I, J;
    while(I<=3){
```

```
        for(J=1; J<=4; J++)
            {
```



```
        }
    I++;
}
```

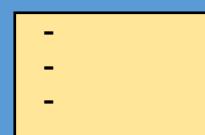
For untuk outer loop dan while untuk inner loop

```
#include<iostream>
using namespace std;
int main()
{
```

```
    int I, J;
    for(I=1; I<=3; I++){

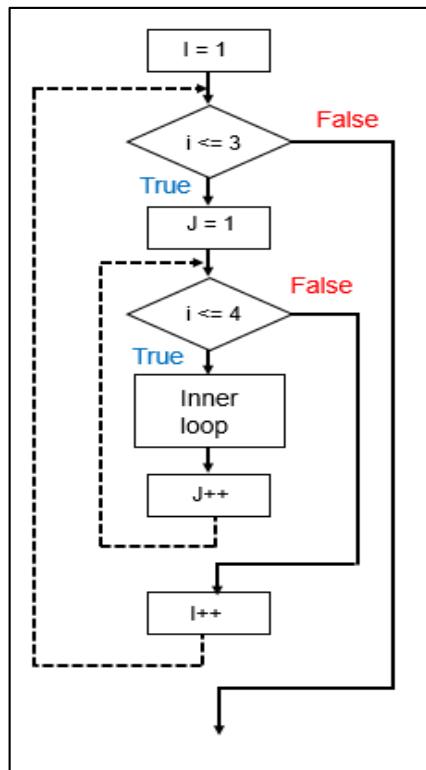
```

```
        J = 1;
        while(J<=4)
            {
```



```
            }
        J++;
    }
```

```
}
```



Gambar 11. 1 Flowchart Nested Loop

3. Contoh – Contoh Penggunaan Nested Loop

Contoh 1: Loop 2 x 3

```

int I, J;
for(I=1; I<=2; I++){
    for(J=1; J<=3; J++){
        {
            cout<<"\nUnpam";
        }
    }
}
  
```



Akan tercetak Unpam
banyak 6 baris:
Unpam
Unpam
Unpam
Unpam
Unpam
Unpam

Perhatikan tabel berikut, tabel berikut memerlukan perkembangan/perubahan nilai I dan J sebagai pembentuk loop.

I	J	Cetak
1	1 2 3 4	Unpam Unpam Unpam Keluar dari loop dalam dan kembali ke loop luar
2	1 2 3 4	Unpam Unpam Unpam Keluar dari loop dalam dan kembali ke loop luar
$I = 3 \rightarrow$ keluar dari loop luar (<i>outer loop</i>)		

Pelaksanaan intruksi dimulai dari inisialisasi **nilai I = 1**, I di beri nilai 1 kemudian dilanjutkan dengan masuk ke loop luar (*outer loop*) dan memeriksa **kondisi I <= 2**, jika kondisi **benar** maka akan inisialisasi **nilai J=1** dan masuk kedalam loop dalam (inner loop) kemudian memeriksa **kondisi J <= 3**, jika kondisi bernilai **benar** maka akan melaksanakan intruksi looping yang ada pada loop dalam (*inner loop*) yang di ulang sebanyak 3 kali untuk mencetak perkataan **Unpam**, dan ketika **nilai J = 4** maka akan keluar dari loop dalam (*inner loop*) dan kembali ke loop luar (*outer loop*). Proses ini terus berulang sampai dengan kondisi **nilai I = 3** maka akan keluar dari outer loop (*loop luar*) dan looping berhenti secara keseluruhan.

Contoh 2.a: loop 2 x 3

```
int I, J;
for(I=1; I<=2; I++){
    for(J=1; J<=3; J++){
        {
            cout<<<J<"\n";
        }
    }
}
```

Hasil cetakan akan menurun
ke bawah karena ada “\n”,

Tercetak:
1
2
3
1
2
3

Contoh 2.b: loop 2 x 3

```
int I, J;
for(I=1; I<=2; I++){
    for(J=1; J<=3;
        J++)
    {
        cout<<J;
    }
}
```

Tercetak: 1 2 3 1 2 3

"\n" digunakan untuk intruksi pindah baris. Setiap kali setelah mencetak nilai J, maka akan pindah baris.

Contoh 2.c : loop 2 x 3

```
int I, J;
for(I=1; I<=2; I++){
    for(J=1; J<=3; J++)
    {
        cout<<<J;
    }
    cout<<endl;
}
```

I	J
1	1
	2
	3
2	1
	2
	3

Tercetak

1 2 3
1 2 3

Setiap 3 kali mencetak nilai J, akan turun satu baris dengan intruksi

cout<<endl;

Contoh 2.d : loop 3 x 2

```
int I, J;
for(I=1; I<=3; I++){
    for(J=1; J<=2; J++)
    {
        cout<<<J;
    }
    cout<<endl;
}
```

I	J
1	1
	2
2	1
	2
3	1
	2

Tercetak

1 2
1 2
1 2

Setiap 2 kali mencetak nilai J, akan turun satu baris dengan intruksi

cout<<endl;

Contoh 2.e : loop 2 x 3

```
int I, J;
for(I=1; I<=2; I++){
    for(J=1; J<=3; J++)
    {
        cout<<<I;
    }
    cout<<endl;
}
```

I	J
1	1
	1
	1
2	2
	2
	2

Tercetak

1 1 1
2 2 2

Yang di cetak adalah nilai I.

Contoh 3 : loop 4 x 4

```

int I, J;
for(I=1; I<=4; I++){
    for(J=I; J<=4; J++)
        {
            cout<<J<<" ";
        }
    cout<<endl;
}

```

I	J
1	1
	2
	3
	4
2	2
	3
	4
	4
3	3
	4
4	4

Tercetak

1 2 3 4
2 3 4
3 4
4

Nilai awal dari J di buat menjadi nilai I (J = I), Ketika nilai I = 1, nilai J yang di cetak mulai dari 1 – 4, ketika nilai I = 2, nilai j yang di cetak 2 – 4 , ketika nilai I = 3, nilai J yang di cek 3 – 4, ketika nilai I = 4, nilai J yang dicetak 4.

Contoh 4

```

int N, X, T, I, J;
X = 1; T = 0; N = 4;
for(I=1; I<6; I+=2){
    for(J=I; J<10; J+=3)
        {
            T = T + N;
            N = N + X;
            X = X + 2;
            cout<<T;
        }
    cout<<endl;
}

```

I	J	T = 0	N = 4	X = 1
		T = T+N	N = N+X	X = X+2
1	1	4	5	3
	4	9	8	5
	7	17	13	7
3	3	30	20	9
	6	50	29	11
	9	79	40	13
5	5	119	53	15
	8	172	68	17

Perhatikan perubahan masing – masing nilai di atas,

Yang di cetak adalah nilai T

Tercetak : 4 9 17
30 50 79
119 172

Contoh 5 : loop 5 x 3

```
int var, I, J;
var = 65;
for(I=1; I<6; I++){
    for(J=1; J<4; J++)
    {
        cout<<(char)var<<" ";
        var++;
    }
    cout<<endl;
}
```

Tercetak : A B C
D E F
G H I
J K L
M N O

I	J
1	A B C
2	D E F
3	G H I
4	J K L
5	M N O

Nilai **65** adalah karakter **A** dalam **ASCII**.

Dengan intruksi:

cout<<(char)var<<" "

";

yang dicetak adalah karakternya bukan nilainya angkanya.

65 = Karakter **A**

67 = Karakter **B**

68 = Karakter **C**

69 = Karakter **D**

Dan seterunya sampai dengan

79 = Karakter **O**

Contoh 6 :

```
int var, I, J;
var = 'A';
for(I=1; I<6; I++){
    for(J=1; J<I; J++)
    {
        cout<<var<<" ";
        var++;
    }
    cout<<endl;
}
```

Tercetak: A
B C
D E F
G H I J
K L M N O

I	J
1	A
2	B C
3	D E F
4	G H I J
5	K L M N O

Pada loop dalam (inner loop), kondisi di buat **J < I**, sehingga jumlah yang di cetak sejumlah dengan nilai **I** yang sekarang.

misal:

Ketika nilai **I** = 1 → yang di cetak hanya 1, yaitu karakter **A** yang bernilai sama dengan 65 nilai numeriknya.

Ketika nilai **I** = 2 → yang di cetak berjumlah 2, yaitu karakter **B** dan Karakter **C**.

Sampai dengan

Nilai **I** = 5 → yang di cetak berjumlah 5, yaitu Karakter **K**, **L**, **M**, **N**, **O**

Contoh 7 :

```

int var, I, J;
var = 'A';
for(I=1; I<6; I++){
    for(J=1; J<7-I; J++)
    {
        cout<<(char)var<<""
        ;
        var++;
    }
    cout<<endl;
}

```

Tercetak:

A	B	C	D	E
F	G	H	I	
J	K	L		
M	N			
O				

I	J
1	A B C D E
2	F G H I
3	J K L
4	M N
5	O

Pada loop dalam (inner loop), kondisi di buat $J < 7 - I$, sehingga jumlah yang di cetak sejumlah dengan 7 dikurangi dengan nilai I yang sekarang.

Misal:

Ketika nilai $I = 1 \rightarrow (7 - 1 = 6)$ karena batas nilainya kurang dari 6, sehingga yang di cetak sejumlah 5, yaitu karakter A, B, C, D, E.

C. Soal Latihan / Tugas

1. Apa yang tercetak bila penggalan program berikut di jalankan, serta berikan penjelasan dengan menggunakan tabel:

A. <pre> int I, J; for(I=0; I<=4; I+=2){ for(J=1; J<=8; J+=3) { cout<<J<<" "; } cout<<endl; } </pre>	B. <pre> int I, J; for(I=0; I<=4; I+=2){ for(J=1; J<=8; J+=3) { cout<<I<<" "; } cout<<endl; } </pre>
C. <pre> int T = 0, I, J; for(I=0; I<=4; I+=2){ for(J=1; J<=8; J+=3) { cout<<T<<" "; } cout<<endl; } </pre>	D. <pre> int I, J; for(I=0; I<=4; I+=2){ for(J=1; J<=8; J+=3) { T = T + J; } cout<<T<<" "<<endl; } </pre>

2. Susun program untuk mencetak nilai-nilai, sehingga tercetak sebagai berikut:

A.		B.	
1 2 3 4 5		1 1 1 1 1	
1 2 3 4 5		2 2 2 2 2	
1 2 3 4 5		3 3 3 3 3	
C.		A.	
1 1 1 1 1		1 2 3 4 5	
2 2 2 2		6 7 8 9	
3 3 3		10 11 12	
4 4		13 14	
5		15	

3. Seorang nasabah meminjam uang sebesar 10 juta rupiah, dengan bunga sebesar 1.5 persen dihitung dari sisa hutang. Setiap akhir bulan nasabah tersebut harus mencicil 10 persen dari saldo hutangnya.

Susun program untuk mencetak daftar cicilan yang harus dibayar tiap akhir bulan sampai sisa hutangnya kurang dari 1 juta rupiah.

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mitra Wacana Media.

PERTEMUAN 12 ARRAY SATU DIMENSI

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu:

1. Memahami konsep array 1 dimensi
2. Merekayasa array 1 dimensi dalam Bahasa pemrograman
3. Menggunakan array 1 dimensi dalam Bahasa pemrograman

B. Uraian Materi

Array adalah sekelompok lokasi memori yang berdekatan yang semuanya memiliki tipe data yang sama. Untuk merujuk pada suatu lokasi atau elemen tertentu dalam array, kita bisa menentukan nama, jumlah elemen dan posisi array tertentu didalamnya.(Edition n.d.)

Ada beberapa sifat yang dimiliki oleh sebuah array :

1. Sekelompok lokasi memori (data) yang mempunyai tipe data yang sama dan menggunakan nama yang sama
2. Sekelompok variabel dapat menggunakan nama yang sama
3. Perbedaan antara varibel dengan variable yang lain adalah apa yang terkandung didalam subscript.
4. Subscript adalah bilangan yang terdapat di dalam kurung kotak []
5. Cara mengakses sebuah array adalah dengan memanggil variable yang ada dalam subscript

1. Array Satu Dimensi

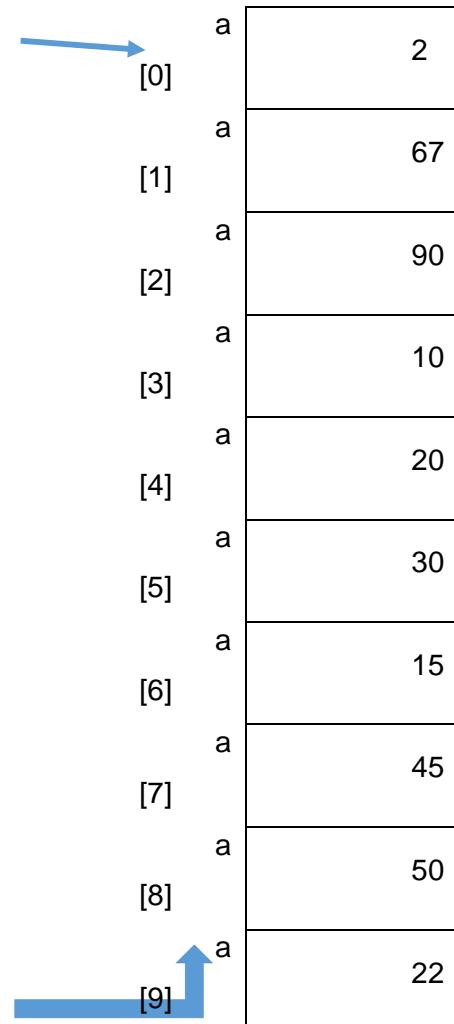
Seperti yang sudah dikatakan diatas, bahwa elemen – elemen diatas mempunyai susunan tertentu. Susunan tersebut bisa berupa satu dimensi, dua dimensi bahkan N dimensi. Tentu susunan yang termudah adalah satu dimensi atau berdimensi satu, sehingga biasa disebut array satu dimensi atau array dimensi satu (one dimension array).

Gambar dibawah ini menunjukkan bahwa array tersebut mempunyai nama a, yang mengandung 10 elemen. Elemen – elemen ini diberikan nama berdasarkan letak atau posisi nomor elemen yang ditandai dengan kurung kotak ([]). Array pertama didalam urutan atau tata letak, selalu dimulai dari urutan ke Nol (0). Dalam pemberian nama pada sebuah array bisa

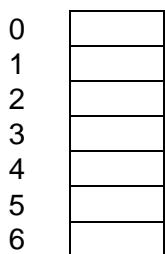
menggunakan huruf-huruf, angka-angka dan garis bawah tetapi tidak bisa menggunakan hanya 1 angka saja.

Bisa dilihat, semua dalam array mempunyai nama yang sama yaitu a.

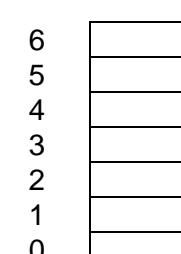
Posisi penumeran dalam array a



Array satu dimensi atau biasa disebut *vector* (karena mempunya satu arah) dapat digunakan untuk berbagai macam keperluan. Kadang – kadang penggambaran array satu dimensi dapat digambarkan dalam contoh dibawah ini.



Biasanya digunakan untuk menggambarkan suatu daftar atau list, sehingga array satu dimensi ini sering disebut dengan **LIST**



Array satu dimensi yang digambarkan seperti ini biasanya untuk mengilustrasikan struktur data yang bersifat tumpukan atau **STACK**

2. Deklarasi Array

Untuk mendeklarasikan sebuah array satu dimensi dalam sebuah Bahasa pemrograman biasanya digunakan tanda [] (Kurung Kotak) / Bracket yang berfungsi untuk menyatakan berapa banyak jumlah elemen yang akan terbentuk pada sebuah array tersebut.

Adapun bentuk umum dari pendeklarasian array satu dimensi :

Tipe_data <spasi> Nama_array [Jumlah_Elemen];

Contoh :

Int Nilai[5]

Pada contoh yang diberikan diatas dinyatakan bahwa array tersebut mempunyai :

Tipe data Array : Integer

Nama Array : Nilai

Jumlah elemen Array : 5

Atau bisa juga pendeklarasian array satu dimensi ini langsung kita deklarasikan disertai dengan isi atau inisialisasi untuk array tersebut.

Tipe_data <spasi> Nama_array [Jumlah_Elemen] = {elemen 1, elemen 2,, elemen ke N};

Contoh :

Int Nilai[5]={70,80,100,50,10};

Pada contoh yang diberikan diatas dinyatakan bahwa array tersebut mempunyai :

Tipe data Array : Integer

Nama Array : Nilai

Jumlah elemen Array : 5

Nilai Elemen Array : [1] = 70

[2] = 80

[3] = 100

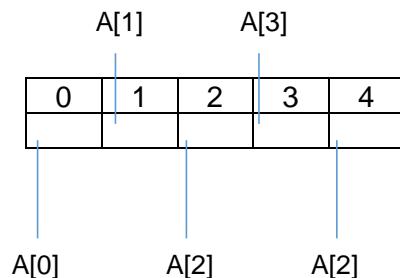
[4] = 50
 [5] = 10

3. Menyiapkan Array satu dimensi pada Bahasa pemrograman C++

Cara menyiapkan array satu dimensi adalah dengan menyebut tipe_data, Nama_array dan jumlah_elemlen array seperti contoh dibawah ini.

```
#include<iostream.h>
Void main ()
{
  Int A[5] ;
```

Dengan perintah A[5]; maka akan disiapkan sebuah array satu dimensi dengan tipe integer yang memuat 5 elemen yang akan diberi nomor indeks dari 0 sampai 4 yang dapat diilustrasikan dengan gambar berikut :



Karena ada 5 elemen, maka masing-masing elemen akan diberikan nama atau sebutan yang berbeda dengan menggunakan nomor indeks, sehingga masing-masing menjadi: A[0], A[1], A[2], A[3], A[4] yang bisa dibaca dengan :

A dengan indeks 0

A dengan indeks 1

A dengan indeks 2

A dengan indeks 3

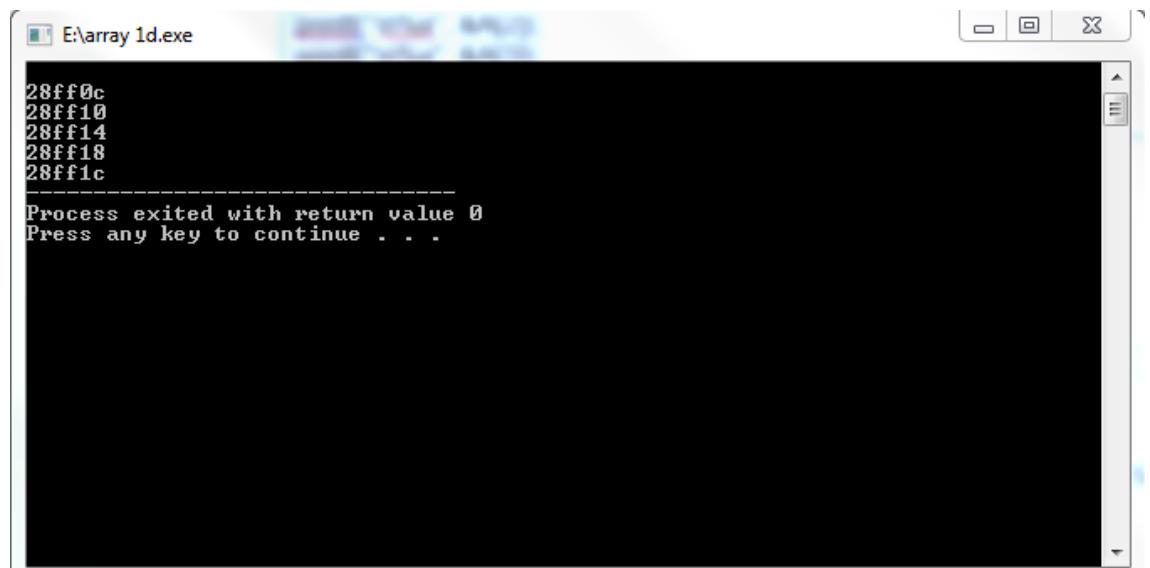
A dengan indeks 4

4. Alamat elemen – elemen Array satu dimensi

Untuk mengetahui besarnya alamat-alamat yang dimiliki oleh setiap elemen array, maka bisa kita gunakan perintah seperti dibawah ini :

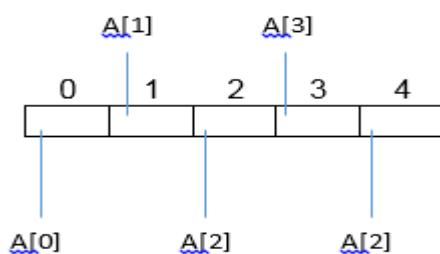
```
#include <stdio.h>
main ()
{
int A[5];
printf("\n%x", &A[0]);
printf("\n%x", &A[1]);
printf("\n%x", &A[2]);
printf("\n%x", &A[3]);
printf("\n%x", &A[4]);
}
```

Maka akan tercetak :



Penjelasan :

Dengan ditulisakannya int A[5] maka akan terbentuk ilustrasi sebagai berikut :



Tanda & digunakan untuk menunjukkan posisi atau alamat. `&A[0]` berarti posisi atau alamat elemen `A[0]`.

Untuk mencetak posisi atau alamat suatu area, dapat menggunakan perintah `%x`.

Alamat yang tercetak adalah alamat elemen dalam notasi bilangan Hexadesimal. Sifat dari alamat yang tercetak ini berbeda beda, tergantung dari computer yang digunakan.

Pada contoh yang diberikan diatas, array yang bertipe integer akan membentuk elemen yang mempunyai panjang 4 byte per elemen, sehingga alamat tiap-tiap elemen selalu berselisih 4 :

Alamat `A[1]` merupakan sambungan dari alamat `A[0]`

atau

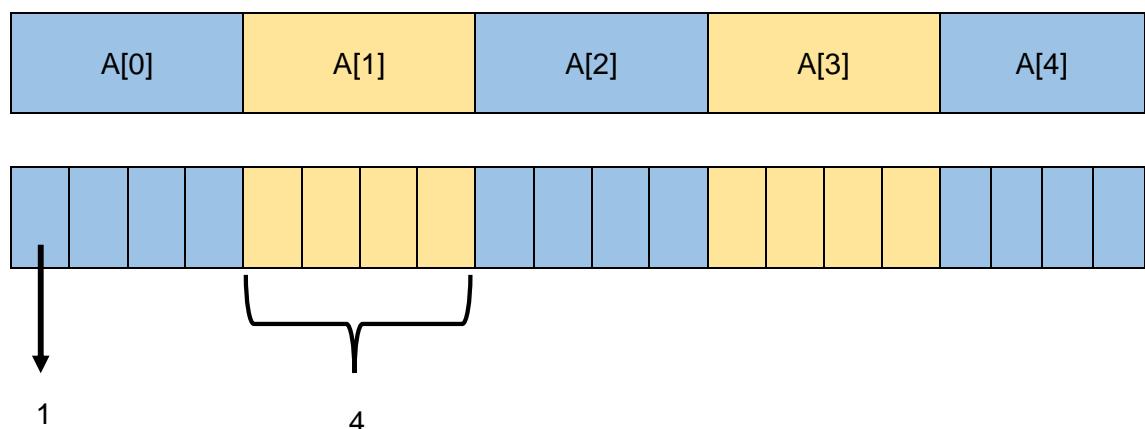
Alamat `A[1] = alamat A[0] + 4`

Alamat `A[2] = alamat A[1] + 4`

Alamat `A[3] = alamat A[2] + 4`

Alamat `A[4] = alamat A[3] + 4`

Contoh tersebut juga dapat diilustrasikan seperti gambar berikut



Jumlah elemen array yang ada adalah = 5 elemen

Panjang array = 5 elemen * 4 byte/elemen = 20 byte

Contoh perhitungan array satu dimensi

- a. Suatu array dideklarasikan dengan A[5], array tersebut mempunyai tipe data integer sehingga masing-masing elemen terdiri dari 4 byte. Jika $28ff0c_{(H)}$ adalah alamat dari elemen pertama, tentukan :
- 1) Jumlah elemen yang terdapat dalam array satu dimensi tersebut
 - 2) Panjang array satu dimensi tersebut dalam satuan byte
 - 3) Alamat dari elemen A[4]
- b. Suatu array dideklarasikan dengan A[9], array tersebut mempunyai tipe data long integer sehingga masing-masing elemen terdiri dari 8 byte. Jika $16FF_{(H)}$ adalah alamat dari elemen pertama, tentukan :
- 1) Jumlah elemen yang terdapat dalam array satu dimensi tersebut
 - 2) Panjang array satu dimensi tersebut dalam satuan byte
 - 3) Alamat dari elemen A[5]

Penyelesaian

- a. Alamat dari elemen A[0] = $28ff0c_{(H)}$
Masing-masing elemen terdiri dari = 4 byte/element
1) Jumlah elemen yang terdapat dalam array satu dimensi tersebut = 5 elemen
2) Panjang array satu dimensi tersebut dalam satuan byte = $5 * 4$ byte/element = 20 byte
3) Alamat dari elemen A[4] = ?
Pergeseran = (4-0) elemen * 4 byte/element
= $16_{(10)} = 10_{(H)}$
 $\&A[4] = 28ff0c_{(H)} + 10_{(H)} = 28ff1c_{(H)}$
- b. Alamat dari elemen A[0] = $16FF_{(H)}$
Masing-masing elemen terdiri dari = 8 byte/element
a) Jumlah elemen yang terdapat dalam array satu dimensi tersebut = 9 elemen
b) Panjang array satu dimensi tersebut dalam satuan byte = $9 * 8$ byte/element = 72 byte
c) Alamat A[5] = ?
Pergeseran = (5-0) elemen * 8 byte/element

$$\begin{aligned} &= 40_{(10)} = 28_{(H)} \\ \&A[4] = 16FF_{(H)} + 28_{(H)} = 1727_{(H)} \end{aligned}$$

5. Membuat Array Numerik Dengan Nilai Awal

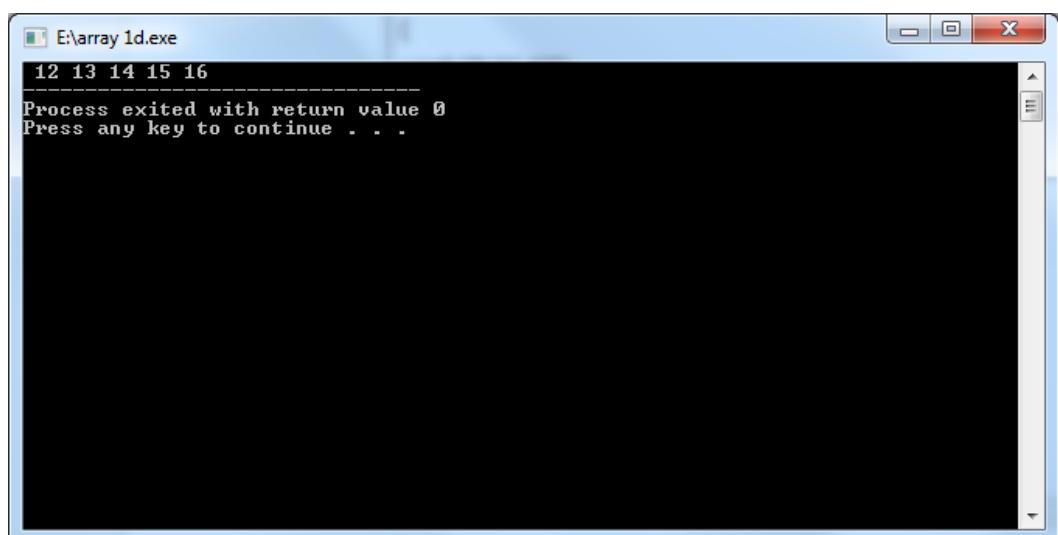
Dalam membuat Array Numerik dengan nilai awal, bisa dilakukan dengan cara mendeklarasikan jumlah elemen array terlebih dahulu maupun tidak dideklarasikan.

Dalam memberikan nilai dalam sebuah elemen array dapat dilakukan dengan cara menuliskan angka-angka yang akan dimuat dalam elemen array dengan dihalusai dan ditutup dengan tanda kurung kurawal ({}) dan antar angka yang satu dengan angka yang lain dipisahkan dengan tanda koma.

Contoh

```
#include <stdio.h>
main ()
{
int I;
int A[5] = {12,13,14,15,16};
for (I=0; I<=4; I++)
{
printf ("%3i",A[I]);
}
}
```

Maka akan tercetak :



Penjelasan :

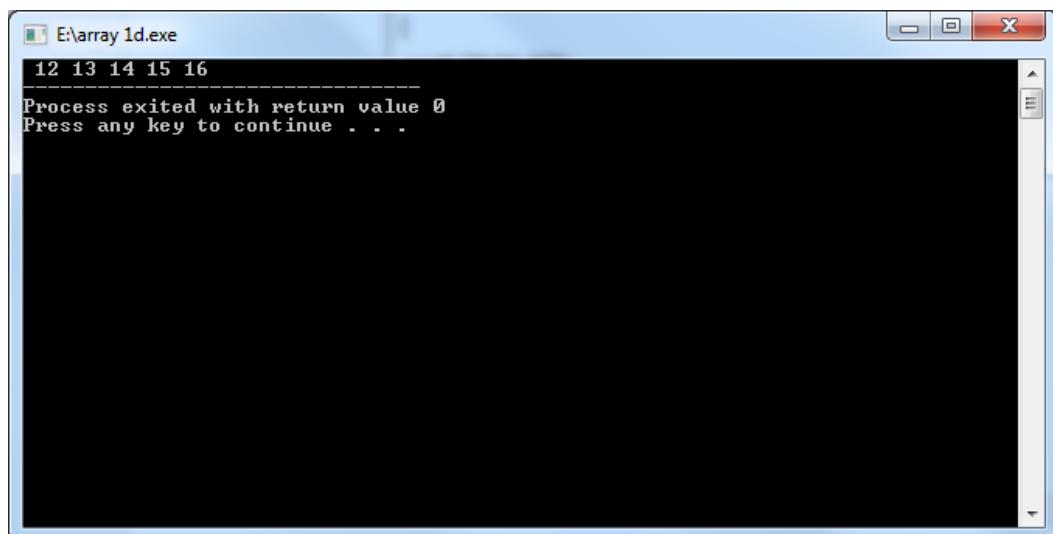
Dengan source code tersebut disiapkan array dengan 5 (Lima) elemen dengan isinya yang ada didalam tanda kurung kurawal seperti ilustrasi gambar dibawah ini :

0	1	2	3	4
12	13	14	15	16
A[0]	A[1]	A[2]	A[3]	A[4]

Ada berbagai cara untuk menyiapkan array satu dimensi yang mengandung isi elemen didalamnya. Seperti contoh dibawah ini

```
#include <stdio.h>
main ()
{
int l;
int A[ ] = {12,13,14,15,16};
for (l=0; l<=4; l++)
{
printf ("%3i",A[l]);
}
```

Maka akan tercetak



Bisa dilihat pada contoh yang diberikan diatas pendeklarasian array satu dimensi dengan tipe data integer dan nama array A tidak disenutkan berapa banyak ruang atau elemen yang harus disiapkan oleh program tersebut. Tetapi karna setelahnya di deklarasikan data yang akan mengisi elemen elemen yang ada dalam array satu dimensi tersebut maka ruangan yang disiapkan adalah sesuai data yang akan dimasukan kedalam elemen-elemen array tersebut. Dalam contoh ada lima data yang akan dimasukan sehingga disiapkanlah lima ruangan atau elemen.

Pembentukan elemen – elemen array satu dimensi juga bisa menggunakan perintah #define di awal penulisan program atau header program. Seperti contoh berikut :

```
#include <stdio.h>
#define n 5
main ()
{
int l;
int A[n] = {1,2,90,98,55};
for (l=0; l<=4; l++)
{
printf ("%3i",A[l]);
}
}
```

Maka akan tercetak



Penjelasan :

n seperti contoh diatas didefinisikan sebagai lima (5) sehingga n tidak dapat diubah dan data yang akan dimasukan kedalam elemen array pun tidak dapat melebihi lima data. Jika terjadi kelebihan dalam memasukan data maka akan ditampilkan pesan error karena ruangannya yang disediakan tidak cukup untuk menampung data yang akan dimasukan.

C. Soal Latihan / Tugas

1. Array harus memiliki tipe data yang
2. Pendeklarasian elemen pada array satu dimensi biasanya menggunakan tanda baca
3. Pada Array satu dimensi, setiap elemen bertipe data integer memiliki jumlah byte.....dan pada long integer berjumlah.....byte
4. Suatu array satu dimensi dideklarasikan dengan A[10], berapa panjang array dalam byte yang ada pada array tersebut jika tipe data yang digunakan adalah long integer....
5. Suatu array mempunyai alamat pertama A[0] = 18FF2_(H), jika array tersebut bertipe data integer tentukan alamat dari elemen A[7]
6. Buatlah source code atau program sederhana untuk memasukkan dan menampilkan array satu dimensi dengan 10 elemen yang didalamnya memuat data 80, 76, 87, 10, 67, 90, 77, 54, 9, 32

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*.
BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*.
Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*.
Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 13 ARRAY DUA DIMENSI

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu :

1. Memahami penerapan dan konsep Array 2 Dimensi
2. Memahami Penerapan tipe data pada Array 2 Dimensi
3. membuat membuat array 2 dimensi dalam pemrograman

B. Uraian Materi

1. Kelebihan & Kekurangan Array

Kelebihan Array :

- a. Array sangat cocok untuk pengaksesan acak, sembarang komponen di array dapat di akses secara langsung tanpa melalui komponen-komponen lain.
- b. Jika berada di suatu lokasi komponen, maka sangat mudah meneruskan ke komponen-komponen tetangga, baik komponen pendahulu atau komponen penerus 3.
- c. Jika komponen-komponen array adalah nilai-nilai independen dan seluruhnya harus terjaga, maka penggunaan dan penyimpanannya sangat efisien.

Kekurangan Array :

- a. Array harus bertipe homogen, kita tidak dapat mempunyai array dimana satu komponen adalah karakter, komponen lain bilangan, dan komponen lain adalah tipe-tipe lain.
- b. Kebanyakan Bahasa pemrograman mengimplementasikan array static yang sulit diubah ukurannya di waktu eksekusi. Bila penambahan dan pengurangan terjadi terus-menerus, maka representasi statis
 - 1) Tidak efisien dalam penggunaan memori
 - 2) Menyimpan banyak waktu komputasi
 - 3) Pada suatu aplikasi, representasi statis tidak dimungkinkan

2. Definisi

Jika sebelumnya telah di jelaskan perihal array 1 dimensi, yakni bahwa array 1 dimensi merupakan metode yang di gunakan untuk menampung sejumlah data dalam bentuk numerik maupun non numerik,

Array dua dimensi biasa disebut sebagai array yang mempunyai dua subskrip yaitu baris dan kolom, array dua dimensi adalah array yang terdiri dari n buah baris dan m buah kolom . Dimana elemen pertama menunjukan baris dan elemen kedua menunjukan kolom.

Array dua dimensi yang sering digambarkan sebagai sebuah matriks, merupakan perluasan dari sebuah array satu dimensi. Jika array satu dimensi hanya terdiri dari sebuah baris dengan beberapa kolom elemen maka array dua dimensi terdiri dari beberapa baris dan beberapa kolom elemen yang bertipe sama sehingga dapat di gambarkan seperti ini :

```
int B[9][5];
```

	[0]	[1]	[2]	[3]	[4]
[0]	0,0	0,1	0,2	0,3	0,4
[1]	1,0	1,1	1,2	1,3	1,4
[2]	2,0	2,1	2,2	2,3	2,4
[3]	3,0	3,1	3,2	3,3	3,4
[4]	4,0	4,1	4,2	4,3	4,4
[5]	5,0	5,1	5,2	5,3	5,4
[6]	6,0	6,1	6,2	6,3	6,4
[7]	7,0	7,1	7,2	7,3	7,4
[8]	8,0	8,1	8,2	8,3	8,4

Array dua dimensi ini, misalnya mengilustrasikan sebuah bangunan terdiri dari 9 lantai dan masing-masing terdiri dari 5 ruangan . disebut dua dimensi karena menunjukan sebuah ruangan diperlukan dua penunjuk yaitu nomor lantai dan nomor ruang.

Bentuk umum array 2 dimensi :

Nama Matriks : array [1..MaxBaris, 1.. MaxKolom] of TipeData

Tipe_Data : menyatakan jenis elemen matriks (int, float, char, unsigned, dan lain-lain), tidak boleh jenis void.

Nama_Matriks : adalah nama matriks, harus memenuhi ketentuan pengenal.

Baris : menyatakan jumlah maksimal elemen baris matriks,

Kolom : menyatakan jumlah maksimal elemen kolom matriks.

Contoh :

Sebuah matrik X berukuran 2x3 dapat dideklarasikan sebagai berikut :

Algoritma :

a : array [1..2,1..3] of integer

a1= 11 b1 = 12

a2 = 7 b2 = 3

a3 = 4 b3 = 9

C++ :

Int X [2] [3] = {{11, 7, 4} , {12, 3, 9}};

Yang akan menempati lokasi memori dengan susunan berikut :

	0	1	2
0	11	7	4
1	12	3	9

Dan definisi variabel untuk setiap elemen tersebut adalah :

	0	1	2
0	a1	a2	a3
1	b1	b2	b3

Contoh Program 1 :

```
#include<iostream.h>
main ()
{
    int x [3] [3] = { { 1, 2, 3 } , { 4, 5, 6 } , { 7, 8, 9 } };
    int i , j ;

    for (i=0; i < 3; i++)
    {
        for ( j=0; j < 3; j++)
            cout << x [ i ] [ j ] << " ";
            cout << endl;
    }
}
```

Output :

1	2	3
4	5	6
7	8	9

Contoh Program 2 :

```
#include<stdio.h>

void printArray(int[][]3];
main() {
    int matrikA [2][3] = { {2, 3, 4}, {5, 4, 3} };
    int matrikB [2][3] = { {1, 1, 1}, {2, 2} };
    int matrikC [2][3] = { {4, 3}, {2} };
    printArray(matrik1);
    printArray(matrik2);
    printArray(matrik3);
    return 0;
}
void printArray(int a[][]3) {
    int i, j;
    for (i = 0; i <= 1; i++) {
        for (j = 0; j <= 2; j++)
            printf("%d ", a[i][j]);
        printf("\n");
    }
}
```

Output :

2	3	4
5	4	3
1	1	1
2	2	0
4	3	0
2	0	0

Penjelasan :

Dari source code dan table diatas untuk matriks A, inputannya adalah 234 543, sedangkan matriks B inputannya adalah 111 220. 0 yang mempunyai arti tempat yang ada untuk kolom ketiga dari baris kedua tidak diisi, dan pada matriks C disini kita bisa melihat bahwa pada baris pertama kolom ketiga data tidak diisi dan dianggap 0 dan pada baris kedua kolom kedua dan ketiga juga tidak diisi juga diisi 0.

Dalam source code tersebut, juga digunakan fungsi untuk menampung hasil penjumlahan matriks.

Perhatikan contoh lain:

Int datasiswa [4] [3] ;

Deklarasi diatas digunakan untuk mendeklarasikan suatu data produksi kayu yang berbentuk demikian:

No	Bagian	Jumlah Karyawan		
		2016	2017	2018
1	Kayu Jati	20	22	25
2	Kayu Mahoni	80	52	77
3	Kayu Pinus	70	74	97
4	Kayu Sungkai	78	70	77

Dari deklarasi diatas maka angka [4] menyatakan jumlah kelas, dan angka [3] menyatakan tahun.

Data produksi [0][3] adalah kayu jati dan jumlah siswa tahun 2018 yaitu 25. Atau jumlah kayu jati pada tahun 2018 adalah 25.

Bentuk data siswa dapat juga digambarkan sebagai berikut:

	1	2	3
1	20	22	25
2	80	52	77
3	70	74	97
4	78	70	77

Array ini dapat pula diberi nilai tetap dengan *static* seperti pada array dimensi satu. Deklarasinya adalah sebagai berikut:

Static int jumlah [4] ;[3] =

```
{
    20, 22, 25,
    80, 52, 77,
    70, 74, 97,
    78, 70, 77
};
```

3. Alur Kerja Array 2 Dimensi

Pemrograman dasar dari array 2 dimensi adalah bagaimana cara melakukan input data matrik dan cara mencetak atau menampilkan hasilnya di layar komputer. Dalam contoh ini, diberikan 2 buah matriks A dengan ukuran mxn dan matriks B dengan ukuran pxq.

Algoritma :

- Mulai
- Deklarasikan variabel baris dan kolom, array matrik A dan B
- Input nilai baris dan kolom tiap matriks
- Input data matriks A
 - Buat perulangan baris j mulai dari j=0 hingga j
jika tidak, lanjutkan ke langkah 5
 - Buat perulangan kolom k mulai dari k=0 hingga k
jika tidak, ulangi langkah 4a.
- Input data matriks B
 - Buat perulangan baris j mulai dari j=0 hingga j

jika tidak, lanjutkan ke langkah 6

- 2) Buat perulangan kolom k mulai dari $k=0$ hingga k
jika tidak, ulangi langkah 5a.

f. Cetak data Catriks A

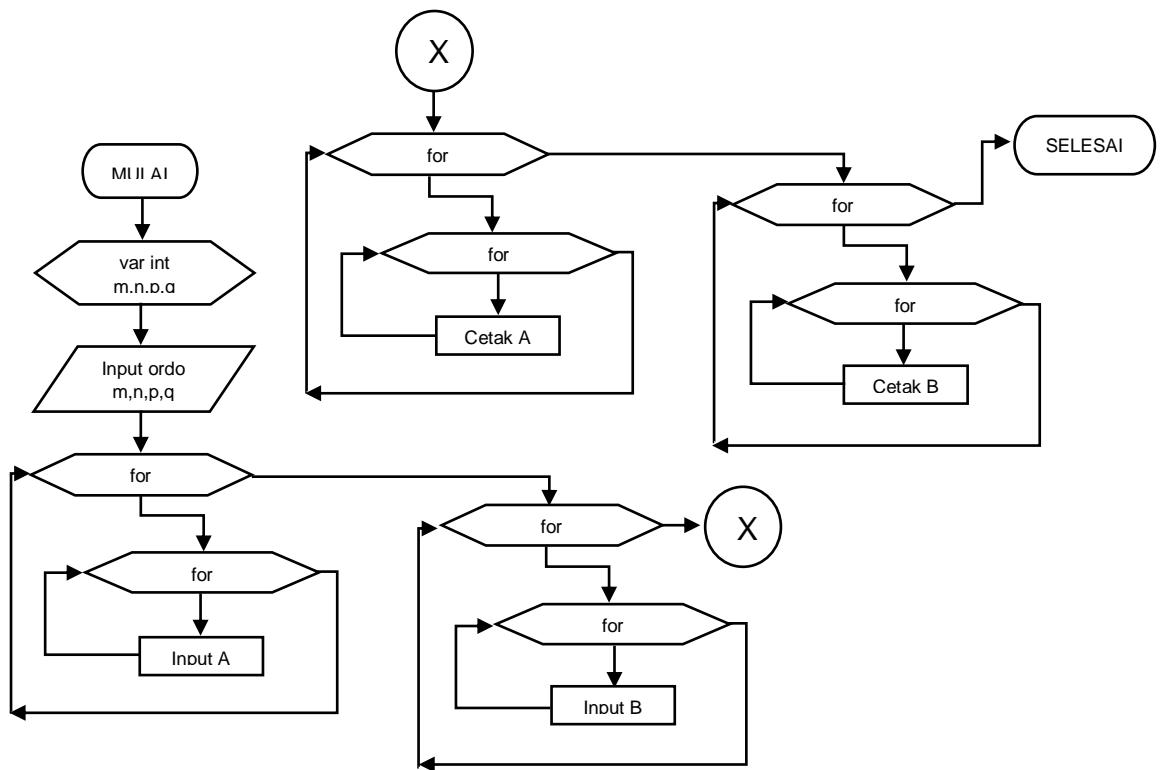
- 1) Buat perulangan baris j mulai dari $j=0$ hingga j
jika tidak, lanjutkan ke langkah 7
- 2) Buat perulangan kolom k mulai dari $k=0$ hingga k
jika tidak, ulangi langkah 6a

g. Cetak data matriks B

- 1) Buat perulangan baris j mulai dari $j=0$ hingga j
jika tidak, lanjutkan ke langkah 8
- 2) Buat perulangan kolom k mulai dari $k=0$ hingga k
jika tidak, ulangi langkah 7a

h. selesai

4. Flowchart



5. Penjumlahan Array

Sudah ada dua buah array 2 dimensi yang dibuat dengan int A [3][2] dan B [3][2] dengan ilustrasi seperti ini :

$$\begin{array}{|c|c|} \hline a & b \\ \hline c & d \\ \hline e & f \\ \hline \end{array} +
 \begin{array}{|c|c|} \hline g & h \\ \hline i & j \\ \hline k & l \\ \hline \end{array} =
 \begin{array}{|c|c|} \hline a+g & b+h \\ \hline c+i & d+j \\ \hline e+k & f+l \\ \hline \end{array}$$

Syarat dua buah matriks (array dua dimensi) dapat dijumlahkan, adalah kedua matriks harus sebangun, jumlah baris dan kolom matriks A harus sama dengan jumlah baris dan kolom matriks B.

Contoh Program 1:

```

#include <iostream>
using namespace std;
int main(){
    // Dengan jumlah elemen baris = 3
    // dan jumlah elemen kolom = 2
    int matrika [3][2];
    int matrikb [3][2];
    int matrikc [3][2];
    
```

```
// Mendeklarasi variabel untuk
// Indeks perulangan
int i,j;

cout<<"\n\tPenjumlahan Array 2 Dimensi (Matrik)\n";

// Mengisi nilai kedalam
// Elemen-elemen array matrika
for(i=0;i<3;i++){
    for(j=0;j<2;j++){
        cout<<"matrik A ["<<i<<"] ["<<j<<"] = ";
        cin>>matrika[i][j];
        cout<<"matrik B ["<<i<<"] ["<<j<<"] = ";
        cin>>matrikb[i][j];
    }
}
cout<<endl;

// Melakukan penjumlahan array matrik A dan matrik B
// Dan menyimpan hasilnya ke array matrik C;
for(i=0;i<3;i++){
    for(j=0;j<2;j++){
        matrikc[i][j]=matrika[i][j] + matrikb[i][j];
    }
}

// Menampilkan matrik A
cout<<"\nMatrik A\n";
for(i=0;i<3;i++){
    for(j=0;j<2;j++){
        cout<<matrika[i][j]<< " ";
    }
    cout<<endl;
}

// Menampilkan matrik B
cout<<"\nMatrik B\n";
for(i=0;i<3;i++){
    for(j=0;j<2;j++){
        cout<<matrikb[i][j]<< " ";
    }
    cout<<endl;
}

// Menampilkan hasil perhitungan / matrik C
// (dalam bentuk matrik dengan ordo 3x2)
cout<<"\nMatrik C (hasil)\n";
for(i=0;i<3;i++){
    for(j=0;j<2;j++){
        cout<<matrikc[i][j]<< " ";
```

```

    }
    cout<<endl;
}
return 0;
}

```

Hasil :

```

C:\Users\AMARTAI7\Documents\cobal.exe
Penjumlahan Array 2 Dimensi <Matrik>
matrik A [0] [0] = 1
matrik B [0] [0] = 9
matrik A [0] [1] = 2
matrik B [0] [1] = 8
matrik A [1] [0] = 3
matrik B [1] [0] = 7
matrik A [1] [1] = 4
matrik B [1] [1] = 6
matrik A [2] [0] = 5
matrik B [2] [0] = 5
matrik A [2] [1] = 6
matrik B [2] [1] = 4

Matrik A
1 2
3 4
5 6

Matrik B
9 8
7 6
5 4

Matrik C <hasil>
10 10
10 10
10 10

Process exited after 24.59 seconds with return value 0
Press any key to continue . . .

```

6. Perkalian Array 2 Dimensi

Sudah ada dua buah array 2 dimensi yang di buat dengan int A [2][3] dan B [3][2] dengan ilustrasi seperti ini :

$$\begin{array}{|c|c|c|} \hline a & b & c \\ \hline d & e & f \\ \hline \end{array} \times \begin{array}{|c|c|} \hline g & h \\ \hline i & j \\ \hline k & l \\ \hline \end{array} = \begin{array}{|c|c|} \hline M & N \\ \hline O & P \\ \hline \end{array}$$

$$M = a \times g + b \times i + c \times k$$

$$N = a \times h + b \times j + c \times l$$

$$O = d \times g + e \times i + f \times k$$

$$P = d \times h + e \times j + f \times l$$

Syarat dua buah matriks (array dua dimensi) dapat di kalikan adalah jumlah kolom matriks A harus sama dengan jumlah baris matriks B.

Contoh Program 1:

```

#include <iostream>
#include <iomanip>
#include <conio.h>
using namespace std;
main(){
int a1[2],b1[2],temp=-1; // Source code pengambil jumlah kolom dan baris
char pil; // Source code membuat pilihan mengulang atau berakhir ulang: // Source code pilihan mengulang
// Alur Memasukkan jumlah baris dan kolom matrik
cout<<"-----\n";
for (char u='A'; u<='B'; u++){
    temp+=1;

    cout<<"Masukan baris matrik "<<u<<" : "; cin>>a1[temp];
    cout<<"Masukan kolom Matrik "<<u<<" : "; cin>>b1[temp];
    cout<<endl;
}
if(b1[0]==a1[1]){
cout<<"-----\n";
int AA [a1[0]][b1[0]]; //Input jumlah kolom dan baris matrik A
int BB [a1[1]][b1[1]]; // Input jumlah kolom dan baris Matrik B
int hasil[a1[0]][b1[1]]; // Hasil Perkalian Matrik
//Memasukkan angka Array A
cout<<"\nMasukkan angka array A\n";
for(int i=0; i<a1[0]; i++){
    for (int c=0;c<b1[0];c++){
        cout<<"A ["<<i<<"]["<<c<<"] = ";
        cin>>AA[i][c];
    }
    cout<<endl;
}
// Memasukkan angka Array B
cout<<"-----\n";
cout<<"\nMasukkan angka array B\n";
for(int i=0; i<a1[1]; i++){
    for (int c=0;c<b1[1];c++){
        cout<<"B ["<<i<<"]["<<c<<"] = ";
        cin>>BB[i][c];
    }
    cout<<endl;
}
//Proses Perhitungan
for(int i=0; i<a1[0]; i++){
    for (int c=0;c<b1[1];c++){
        hasil [i][c] = (AA[i][0] * BB[0][c] )+( AA[i][1] * BB[1][c]);
    }
    cout<<endl;
}
cout<<"-----\n";

```

```

cout<<"Hasil Perkalian Matrik A dan B adalah : \n";
cout<<-----\n";
//Memampulkan hasil Perkalian Matrik
for(int i=0; i<a1[0]; i++){
    cout<<endl;
    for (int c=0;c<b1[1];c++){
        cout<<"["<<hasil[i][c]<<"] " ;
    }
    cout<<"\n\nATAU\n";
    for(int i=0; i<a1[0]; i++){
        cout<<endl;
        for (int c=0;c<b1[1];c++){
            cout<<"["<<i+1<<"]["<<c+1<<"] = "<<hasil[i][c]<<endl;
        }
    }
}
else {
    //Menanyakan Pilihan
    cout<<"Kolom Matrik A harus sama dengan Baris Matrik B";
    cout<<"\nUlangi..? (Y/N) : "; cin>>pil;
    if (pil=='Y'||pil=='y'){
        goto ulang;
    }
}
getch();
}

```

C. Soal Latihan/ Tugas

1. Array dua dimensi int A [3] [5] (tipe interger numerik) belum ada isinya dengan ilustrasi gambar seperti berikut :

	0	1	2	3	4
0					
1					
2					

Gambarkan kembali array tersebut beserta dengan isinya bila diisi dengan intruksi

a. <pre>for (J=0; J <= 4; J++) { for (I=0; I<=2; I++) A [I] [J] = I ; }</pre>	b. <pre>for (I=0; I <= 2; I++) { for (J=0; J<=4; J+=2) A [I] [J] = J ; }</pre>
c. <pre>N = 1; for (I=0; I <= 2 ; I++) { for (J=0; J <= 4 - I ; J++) { A [I] [J] = N; N++; } }</pre>	d. <pre>N = 1; for (I=0; I <= 2 ; I++) { for (J=0; J <= I + 2 ; J++) { A [I] [J] = N; N++; } }</pre>

2. Array dua dimensi yang sudah di buat dengan int A [5][5] & int B [3][3], susunlah program untuk mengisi array sehingga isinya menjadi :

a.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

b.

4	5	3
4	5	5
9	8	7

3. Buatlah 2 buah table, matriks A [3][2] dan matriks B [2] [3] kalikan isi table tersebut sehingga menghasilkan matriks seperti di bawah ini:

9	9	9
8	8	8
7	7	7

D. Referensi

A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.

Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.

Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.

Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.

Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.

Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

PERTEMUAN 14
ALGORITMA DAN PEMROGRAMAN DALAM MATEMATIKA

A. Tujuan Pembelajaran

Setelah mempelajari materi pada pertemuan ini, mahasiswa mampu membuat aplikasi sederhana menggunakan perhitungan matematika dengan menerapkan algoritma secara runtunan, pemilihan dan perulangan

B. Uraian Materi

Telah di bahas dalam pertemuan ke-2, bahwasannya **Algoritma** merepresentasikan apa yang diketahui. Sehingga jika tidak ada yang diketahui maka tidak akan ada algortima.

Dalam bab ini akan menjelaskan tentang implementasi algoritma dalam matematika seperti:

1. Menentukan suatu bilangan genap atau ganjil.
2. Konversi bilangan desimal ke bilangan biner.
3. Menetukan suatu bilangan merupakan bilangan prima.
4. Mencari faktor persekutuan terbesar.
5. Menghitung luas suatu area/bidang yang dibatasi oleh garis

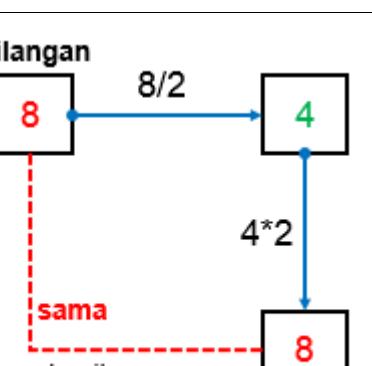
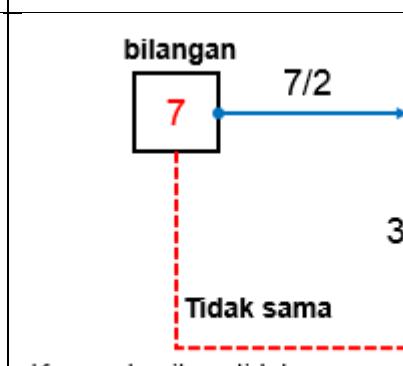
Penjelasannya sebagai berikut.

1. Menentukan suatu bilangan genap atau ganjil.

Untuk dapat mementukan suatu bilangan termasuk bilangan genap atau bilangan ganjil, kita harus mengetahui konsep dari bilangan genap dan bilangan ganjil.

Bilangan genap adalah bilangan yang habis dibagi 0, atau sisa pembagian dari bilangan tersebut sama dengan 0.

Cara 01	Penjelasan
<pre>cin>> bilangan; if (bilangan % 2 == 0) cout<<"bilangan genap"; else cout<<"bilangan ganjil";</pre>	<p>Contoh : bilangan = 8; $8 \% 2 \rightarrow 4$ sisa 0, bilangan genap</p> <p>bilangan = 7; $7 \% 2 \rightarrow 3$ sisa 1, bilangan ganjil</p>

Cara 02	Penjelasan
<pre data-bbox="474 334 846 491"> cin>> bilangan; if(bilangan == (bilangan/2) cout<<"bilangan genap"; else cout<<"bilangan ganjil"; </pre>	<p>apabila bilangan sama dengan bilangan dibagi dua di kali 2 nilai sama dengan bilangan itu sendiri.</p> <p>Contoh:</p> <p>Bilangan = 8; $(8/2)*2 = 8 \rightarrow$ genap</p>
 <p>The diagram illustrates the calculation of 8/2 to 4, then 4*2 to 8. The number 8 is in a box with a red border. An arrow labeled "8/2" points to a box containing the number 4. Another arrow labeled "4*2" points from the box 4 to a final box containing the number 8. A dashed red box labeled "sama" encloses the boxes for 8 and 8, indicating they are equal.</p> <p>Karena hasilnya sama, berarti bilangan genap</p>	 <p>The diagram illustrates the calculation of 7/2 to 3, then 3*2 to 6. The number 7 is in a box with a red border. An arrow labeled "7/2" points to a box containing the number 3. Another arrow labeled "3*2" points from the box 3 to a final box containing the number 6. A dashed red box labeled "Tidak sama" encloses the boxes for 7 and 6, indicating they are not equal.</p> <p>Karena hasilnya tidak sama, berarti bilangan ganjil</p>

2. Konversi bilangan desimal ke bilangan biner

Bila di masukkan suatu bilangan bulat desimal, maka akan menampilkan bilangan dalam bentuk binernya.

Cara 1	Penjelasan
<pre>#include<iostream> using namespace std; int main(){ int bil, I, B, A; cin>>bil; B = 128; for(I=1; I<=8; I++) { if(bil >= B) { bil = bil - B; A = 1; } } }</pre>	<p>Nilai maksimal yang dapat dikonversi adalah 255, tersimpan dalam 8 bit bilangan biner yaitu:</p> <p>1 1 1 1 1 1 1 1</p> <p>Nilai yang paling kiri pada bilangan biner bernilai 128.</p>

<pre> }else{ A = 0; } cout<<A; B = B/2; •-----+ } return 0; } </pre>	<p>Setiap satu bit akan melakukan satu kali loop, di mana looping maksimum adalah 8 kali $I \leq 8$.</p> $128/2 = 64$ $64/2 = 32$ <p>Dan seterusnya Digunakan untuk menyatakan nilai-nilai bit.</p>
--------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Cara 2

Penjelasan

I	N	X	A=bil/B	bil=bil%B	B=B/2
1	175	128	1	47	64
2	47	64	0	47	32
3	45	32	1	15	16
4	15	16	0	15	8
5	15	8	1	7	4
6	7	4	1	3	2
7	3	2	1	1	1
8	2	1	1	0	0

bil % B artinya **bil modulus B**, modulus adalah sisa pembagian.
Apabila nilai yang di masukkan 175 akan mencetak:
1 0 1 0 1 1 1 1

3. Menentukan suatu bilangan merupakan bilangan prima.

Bilangan prima adalah bilangan bulat yang habis dibagi jika hanya di bagi dengan bilangan itu sendiri, kecuali angka 1.

Contoh bilangan prima :

2 3 5 7 11 17 19 23 29 dan seterusnya

terdapat beberapa konsep pemikiran untuk menentukan bilangan prima:

konsep pertama:

Misalnya **bil** merupakan suatu bilangan yang di masukkan, maka akan diperiksa, apakah **Bil** akan habis di bagi dengan salah satu nilai yang ada di bawahnya atau nilai yang lebih kecil dari nilai **bil**

Apabila habis \rightarrow maka **Bil** bukan bilangan prima

Apabila tidak habis \rightarrow maka **Bil** merupakan bilangan prima.

Contoh :

Misal → bil = 9

Bilangan yang ada di bawah nilai 9 adalah : 2, 3, 4, 5, 6, 7, 8

Jika kita bagi nilai 9 dengan nilai atau bilangan yang ada di bawahnya, ternyata 9 habis di bagi dengan 3. Sehingga 9 bukan bilangan prima.

Misal → bil = 11

Bilangan yang ada di bawah nilai 11 adalah : 2, 3, 4, 5, 6, 7, 8, 9, 10, 11

Jika kita bagi 11 dengan nilai atau bilangan yang ada di bawahnya ternyata 11 tidak habis di bagi dengan bilangan yang ada di bawanya.

Sehingga 11 merupakan bilangan prima.

Konsep pertama untuk menentukan bilangan prima, jika di implemetasikan ke dalam program menggunakan bahasa pemrograman C++, dapat di lihat pada listing program berikut:

Listing 14.1: bil prima1.cpp

```
#include<iostream>
using namespace std;
main()
{
    int bil, I, tanda;
    tanda = 0;
    cin>>bil;
    I = 2;
    while(I<=bil-1)
    {
        if(bil % I == 0)
            tanda = 1;
        I++;
    }
    if(tanda == 1)
        cout<<"bukan bilangan prima";
    else
        cout<<"bilangan prima";
}
```



Untuk program di atas:

jika bilangan yang di masukkan 9 maka akan mencetak **bukan bilangan prima**

jika bilangan yang di masukkan 11, maka akan mencetak **bilangan priman**

Konsep Kedua:

Kita ambil contoh bilangan prima berikut: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, ...

Bila kita perhatikan bilangan prima diatas selain angka 2, maka semua bilangan **prima** merupakan bilangan **ganjil**.

Algoritma yang dapat kita buat untuk memeriksa apakah **bil** merupakan bilangan **prima** atau **bukan**, dapat dibuat sebagai berikut:

- a. Jika **bil** = 2, maka cetak “**bilangan PRIMA**” dan proses selesai
- b. Jika **bil** = bukan 2, maka periksa :
 - 1) Jika **bil** bilangan genap (bilangan genap → habis dibagi 2), maka cetak “**BUKAN PRIMA**” dan proses selesai.
 - 2) Jika **bil** bilangan ganjil, maka periksa apakan bilangan tersebut prima atau bukan.

Listing 14.2: bil prima2.cpp

```
#include<iostream>
#include<math.h>
using namespace std;
main()
{
    int bil, batas, X, tanda;
    cin>>bil;
    if(bil==2)
        cout<<"bilangan PRIMA";
    else{
        if(bil%2 == 0)
            cout<<"BUKAN PRIMA";
        else{
            X = 3;
            batas = bil-1;
            tanda = 0;
            while(tanda == 0 && X <= batas)
            {
                if(bil % X == 0)
                    tanda = 1;
                X = X + 2;
            }
            if(tanda == 0)
                cout<<"bilangan PRIMA";
            else
                cout<<"BUKAN PRIMA";
        }
    }
}
```

4. Mencari faktor persekutuan terbesar

Faktor Persebutuan Terbesar (FPB) adalah pembagi yang mempunyai nilai terbesar. Perhatikan contoh berikut:

Misalnya diketahui dua buah bilangan bulat n_1 dan n_2 , masing masing bernilai:

$$n_1 = 30 \text{ dan } n_2 = 105$$

untuk mencari faktor persekutuan terbesar dari kedua bilangan tersebut di lakukan cara berikut:

30 habis dibagi dengan: 1 2 3 5 6 12 15 30

90 habis dibagi dengan: 1 3 5 7 15 21 35 105

Jika kita lihat hasil di atas bahwa nilai 30 dan 105 mempunyai pembagi habis yang sama: 1, 3, 5, 15.

Dan nilai pembagi terbesar adalah 15, jadi faktor persekutuan terbesar dari 30 dan 105 adalah: 15.

Cara diatas adalah cara yang biasa digunakan dalam mencari faktor persekutuan terbesar.

Menggunakan Algoritma Euclidean

Algoritma Euclides adalah implementasi atau penerapan algoritma berulang-ulang atau berkali-kali sampai menghasilkan sisa yang sama dengan nol (sisa = 0).

Misal diketahui dua buah bilangan bulat $n_1 = 105$ dan $n_2 = 30$, dengan menggunakan **euclidean algoritm** dilakukan dengan cara berikut:

$105 = (3)(30) + 15 \rightarrow 3$ didapat dari pembagian $(105/30 = 3)$ \rightarrow dan 15 adalah sisanya

$15 = (1)(15) + 0 \rightarrow 15$ di dapat dari sisa pembagian sebelumnya, 1 didapat dari pembagian $(15/15 = 1)$ \rightarrow sisanya adalah 0.

Jadi faktor pesekutuan terbesar dari 105 dan 30 adalah 15 (merupakan nilai pembagi terakhir).

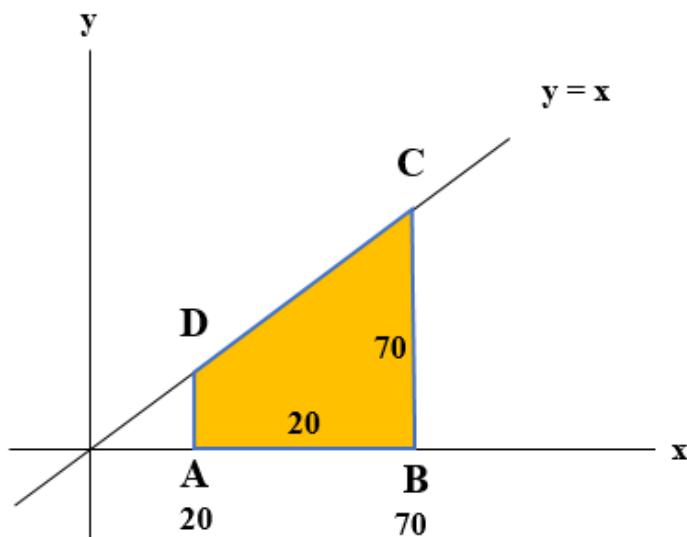
Implementasi Euclidean Algoritm ke dalam bahasa pemrograman C++, dapat di lihat dengan progam berikut:

Listing 14.3: fpb.cpp

```
#include<iostream>
using namespace std;
main()
{
    int n1, n2, x, sisa;
    cin>>n1>>n2;
    if(n1<n2)
    {
        x=n1; n1=n2; n2=x;
    }
    while(n2 != 0)
    {
        sisa = n1%n2;
        n1=n2;
        n2 = sisa;
    }
    cout<<"FPT : "<<n1;
}
```

5. Menghitung luas suatu area/bidang yang dibatasi oleh garis

Sebagai contoh terdapat sebuah trapesium ABCD dengan gambar sebagai berikut:



Trapesium di atas di bentuk oleh empat buah garis:

$$y = 0$$

$$y = x$$

$$x = 20$$

$$x = 70$$

secara matematika luas trapesium dapat dihitung dengan

$$= (70 + 20) * 50 / 2$$

$$= 2250$$

Jika menggunakan integral:

$$\begin{aligned} \text{Luas} \\ \text{trapesium} &= \int_{20}^{70} x \, dx \\ &= \left[\frac{1}{2} x^2 \right]_{20}^{70} \\ &= 0.5 * 4900 - 0.5 * 400 \\ &= 0.5 * 4500 \\ &= 2250 \end{aligned}$$

Listing 14.4: luastrapesium.cpp

```
#include<iostream>
using namespace std;
main()
{
    float awal = 20, akhir=70, dx, LuasTotal;
    float Luasdx, X, Y;

    dx = (akhir-awal)/50;
    LuasTotal= 0;
    while(awal<akhir){
        X = awal+(0.5*dx);
        Y=X;
        Luasdx = Y * dx;
        LuasTotal = LuasTotal + Luasdx;
        awal = awal + dx;
    }
    cout<<"Luas Trapesium : "<<LuasTotal;
}
```



Jika program diatas dijalankan akan mencetak:
2250

C. Soal Latihan / Tugas

1. Susun program untuk mengkonversi bilangan biner ke bilangan desimal ?
2. Susun algoritma dan buat program untuk memasukkan dua buah bilangan bulat positif yang berbeda, kemudian cetak nilai persekutuan kelipatan terkecil (KPK) kedua buah bilangan tersebut.
3. Susun algoritma dan buat program untuk mencetak bilangan segitiga pascal dengan memanfaatkan atau menggunakan array bedimensi satu.

D. Referensi

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*. Bandung: Penerbit Informatika.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.

GLOSARIUM

Arithmetic Expression	: Ekspresi Matematika
Algoritma euclidean	: merupakan suatu algoritma untuk mencari Faktor Persekutuan Terbesar (FPB) dari dua bilangan, khususnya untuk bilangan-bilangan yang nilainya sangat besar sehingga tidak diperlukan untuk mencari faktorisasi prima dari kedua bilangan tersebut
Array	: Array adalah sekelompok lokasi memori yang berdekatan yang semuanya memiliki tipe data yang sama
Assigment Statement	: Pernyataan Penugasan
Bit	: binary digit, bilangan biner
Byte	: Satuan memory yang terdiri dari 8 bit
Compiler	: program yang berfungsi untuk mengubah bahasa pemrograman menjadi bahasa mesin atau bahasa yang di mengerti oleh komputer
Flowchart	: Diagram Alir
Homogen	: Sama, sejenis
Input	: Masukkan
int (integer)	: bilangan bulat
Keyword	: Kata Kunci
Kode ASCII	: Kode Standar Amerika untuk Pertukaran Informasi atau sebuah standar internasional dalam pengkodean huruf dan simbol seperti Unicode dan Hex tetapi ASCII lebih bersifat universal
Kondisi(cond)	: suatu pernyataan yang mengandung nilai benar atau salah
konversi	: perubahan dari satu sistem pengetahuan ke sistem yang lain
manipulator	: digunakan untuk mengatur tampilan data
output	: Keluaran
Program	: Serankaian intruksi untuk melakukan suatu fungsi
Pseudocode	: Kode semu
Sequence	: Runtunan
source code	: sandi Sumber, kode sumber, kode program
static	: Statik
String	: kumpulan karakter
Syntax	: Tata bahasa, Aturan menulis kode program

REFERENSI

- A.S, R. (2018). *LOGIKA ALGORITMA dan PEMROGRAMAN DASAR*. BANDUNG: MODULA.
- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Data, R. (t.thn.). *Data, Refsnes*. Dipetik October 22, 2019, dari https://www.w3schools.com/cpp/cpp_booleans.asp
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Haltermann, R. L. (2019). *Fundamental of C++ Programming*.
- Kirch-Prinz, U., & Kirch-Prinz, U. (2002). *A Complete Guide to Programming in C++*. Sudbury: Jones and Bartlett Publishers.
- Kristanto, A. (2003). *Algoritma & Pemograman Dengan C++*. Yogyakarta: Graha Ilmu.
- Lestari, F. D. (2017). Analisa Algoritma Faktor Persekutuan Terbesar (FPB) Menggunakan Bahasa Pemrograman C++. *Jurnal Evolusi*, 63-68.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal danC*. Bandung: Penerbit Informatika.
- S, R. A. (2018). *Logika Algoritma dan Pemrograman Dasar*. Bandung: Modula.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mltra Wacana Media.
- Yuniati, S. (2012). Menentukan Kelipatan Persekutuan Terkecil (Kpk) Dan Faktor Persekutuan Terbesar (Fpb) Dengan Menggunakan Metode “PEBI”. *Beta*, 149-165.

RENCANA PEMBELAJARAN SEMESTER (RPS)

Program Studi	: S-1 Teknik Informatika	Sks	: 2 Sks
Mata Kuliah/Kode	: Algorithma & Pemrograman I / TPL0022	Prasayarat	: -
Semester	: I	Kurikulum	: KKNI
Deskripsi Mata Kuliah	: Mata kuliah ini merupakan mata kuliah wajib Program Studi S-1 Teknik Infomatika yang membahas tentang Pengantar Algoritma, Dasar-dasar Algoritma, Tipe Data, Ekspresi, Operator, Array dan Pengambilan Keputusan.	Capaian Pembelajaran	: Setelah menyelesaikan mata kuliah ini mahasiswa mampu membuat aplikasi sederhana dengan menerapkan algoritma secara runtunan, pemilihan dan pengulangan
Penyusun	: 1. Hendri Ardiansyah, S.Kom., M.Kom(Ketua); 2. Agus Budi Prasetyo, S.Kom., M.Kom(Anggota 1); 3. Resti Amalia, S.Kom., M.Kom(Anggota 2);		

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
(1)	(2)	(3)	(4)	(5)	(6)	(7)
1	Mahasiswa mampu menuliskan algoritma dalam tiga bentuk flowchart, Pseudo Code dan kalimat deskriptif	Pengantar Algoritma dan Pemrograman.	Diskusi, simulasi dan penugasan	Membuat Algoritma dalam bentuk Flowchart, Pseudo Code dan kalimat deskriptif.	Kesesuaian Penulisan Algoritma dalam bentuk flowchart, Pseudo Code dan kalimat deskriptif.	6%
2	Mahasiswa mengetahui tentang bilangan biner. Mengetahui tentang konversi bilangan biner dan operasi penjumlahan bilangan biner	Sistem Bilangan Biner	Diskusi, simulasi dan penugas	Mengkonversikan bilangan desimal ke biner, melakukan operasi penjumlahan dan pengurangan bilangan biner	Kesesuaian hasil konversi bilangan biner dan hasil operasi penjumlahan bilangan biner	6%
3	Mahasiswa mampu Membedakan jenis-jenis tipe data dasar dalam pemrograman	Tipe Data, Variabel dan Konstanta.	Diskusi, Simulasi dan Penugasan	Mendeklarasikan varabel sesuai dengan tipe data yang digunakan	Kesuaian pendeklarasian variabel dengan	6%

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
	Mahasiswa mampu Menggunakan jenis-jenis tipe data dasar dalam pemrograman Mahasiswa mampu Memahami penggunaan variabel dan konstanta dalam pemrograman. Mahasiswa mampu mendeklarasikan variabel menggunakan jenis-jenis tipe data dasar				tipe data yang digunakan	
4	Mahasiswa mampu menuliskan dan membuat Assigment Statement, Aritmetic Expression dan Operator dalam pemrograman	Assignment Statement, Aritmetic Expression dan Operator.	Diskusi, Simulasi dan Penugasan	Membuat assigment statement, aritmatic expression dan operator dalam pemrograman	Kesesuaian dan ketepatan dalam membuat assignment statement dan Aritmetic Expression. Kesesuaian penggunaan operator dalam pemrograman.	6%
5	mahasiswa mampu memahami dan mengerti dari penggunaan dari setiap fungsi pada preprocessor dan library function. Serta mahasiswa mampu menerapkan penggunaannya di dalam pemrograman.	Preprocessor dan Library Function.	Diskusi, Simulasi dan Penugasan	Menggunakan preprocessor dan library function dalam pemrograman.	Kesesuaian penggunaan preprocessor dan library function dalam membuat program	6%

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
6	Mahasiswa mampu Membedakan operasi input dan output, Membuat program dengan operasi input dan output, Memahami penggunaan variabel dan konstanta dalam operasi input dan output, Memahami persoalan dan memahami penyelesaiannya.	Input, Output, Algoritma dan Pengetahuan Terkait.	Diskusi, Simulasi dan Penugasan	Membuat input dan output untuk membuat program sederhana. Menerapkan pengetahuan dalam pemrograman	Ketepatan dan kesesuaian dalam membuat input dan output untuk membuat program.	7%
7	Memahami fungsi statement IF, Dapat memahami logika dengan fungsi IF, Dapat membuat program sederhana menggunakan fungsi IF	Control Statement menggunakan if	Diskusi, Simulasi dan Penugasan	Membuat program sederhana menggunakan control statement if.	Ketepatan dan kesesuaian penggunaan if dalam membuat program	8%
UTS						
8	Mahasiswa Mampu Mengimplementasikan algoritma menggunakan control statement IF dalam pemecahan suatu masalah dengan berbagai alternatif jawaban yang tersedia dengan mengambil keputusan untuk memilih jawaban yang tepat.	Control Statement menggunakan IF (Lanjutan)	Diskusi, Simulasi dan Penugasan	Menggunakan control statement menggunakan if untuk membuat program sederhana. Membuat algoritma dalam bentuk flowchar menggunakan selection if	Ketepatan dan kesesuaian dalam membuat program control statement IF. Ketepatan dan kesesuaian dalam membuat flowchart	6%

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
9	Setelah mempelajari materi ini, mahasiswa mampu mahasiswa mengerti tentang penggunaan control statement menggunakan NESTED if dan switch dan Logical Operator dapat menggunakan dalam pemrograman	Control Statement menggunakan NESTED IF dan Logical operator	Diskusi, Simulasi dan Penugasan	Menerapkan control statement if untuk membuat program sederhana	Ketepatan dan kesesuaian dalam membuat program menggunakan control statement NESTED IF dan logical operator	7%
10	Mahasiswa mampu membedakan konsep dasar penggunaan perintah-perintah perulangan (looping) pada bahasa pemrograman, Mengerti tentang penggunaan control statement menggunakan for dan while, dan dapat menggunakan dan mengaplikasikannya dalam program	Perulangan menggunakan for, while	Diskusi, Simulasi dan Penugasan	Menerapkan control statement menggunakan for dan while untuk membuat program sederhana	Ketepatan dan kesesuaian dalam membuat program menggunakan control for dan while	8%
11	Mahasiswa Mampu menerapkan perulangan menggunakan bentuk do ... while ... Mahasiswa Mampu menerapkan perulangan dalam perungan atau nested loop Mahasiswa mampu Membuat aplikasi	Perulangan menggunakan do While dan Nested loop	Diskusi, Simulasi dan Penugasan	Menerapkan control statement menggunakan do while dan nested loop untuk membuat program sederhana	Ketepatan dan kesesuaian dalam membuat program menggunakan control statement do while dan nested loop	8%

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
	sederhana dengan menggunakan do while dan nested loop					
12	Mahasiswa Memahami konsep array 1 dimensi Mahasiswa mampu membuat array 1 dimensi dalam Bahasa pemrograman Mahasiswa mampu Menggunakan array 1 dimensi dalam Bahasa pemrograman	Array Satu dimensi	Diskusi, Simulasi dan Penugasan	Membuat array satu dimensi dan mengimplementasikan array satu dimensi dalam pemrograman	Ketepatan dan kesesuaian dalam membuat program menggunakan Array satu dimensi	8%
13	Mahasiswa mampu Memahami penerapan dan konsep Array 2 Dimensi Mahasiswa mampu Memahami Penerapan tipe data pada Array 2 Dimensi Mahasiswa mampu membuat membuat array 2 dimensi dalam pemrograman	Array Dua Dimensi	Diskusi, Simulasi dan Penugasan	Membuat array dua dimensi dan menerapkan array dua dimensi dalam program sederhana (perkalian matriks)	Ketepatan dan kesesuaian dalam membuat program menggunakan Array dua Dimensi	8%
14	Mahasiswa mampu membuat aplikasi sederhana menggunakan perhitungan matematika dengan menerapkan algoritma secara runtunan, pemilihan dan perulangan	Algoritma dan Pemrograman dalam Matematika	Diskusi, Simulasi dan Penugasan	Membuat algoritma dan program untuk menyelesaikan persoalan matematika menggunakan bahasa pemrograman C++	Kesesuaian dan ketepatan dalam membuat algortima dan program	10%

PERTEMUAN KE-	KEMAMPUAN AKHIR YANG DIHARAPKAN	BAHAN KAJIAN (MATERI AJAR)	METODE PEMBELAJARAN	PENGALAMAN BELAJAR MAHASISWA	KRITERIA PENILAIAN	BOBOT NILAI
			UAS			

Referensi

- Charibaldi, N. (2004). *Modul Kuliah Algoritma Pemrograman II Edisi Kedua*. Yogyakarta.
- Data, R. (t.thn.). *Data, Refsnes*. Dipetik October 22, 2019, dari https://www.w3schools.com/cpp/cpp_booleans.asp
- Davis, S. R. (2014). *C++ For Dummies* (7th ed.). John Wiley & Sons, Inc.
- Deitel, P., & Deitel, H. (2014). *C++ How To Program* (9th ed.). United State of America: Pearson.
- Halterman, R. L. (2019). *Fundamental of C++ Programming*.
- Kirch-Prinz, U., & Kirch-Prinz, U. (2002). *A Complete Guide to Programming in C++*. Sudbury: Jones and Bartlett Publishers.
- Kristanto, A. (2003). *Algoritma & Pemograman Dengan C++*. Yogyakarta: Graha Ilmu.
- Lestari, F. D. (2017). Analisa Algoritma Faktor Persekutuan Terbesar (FPB) Menggunakan Bahasa Pemrograman C++. *Jurnal Evolusi*, 63-68.
- Munir, R. (2005). *Algoritma dan Pemrograman dalam Bahasa Pascal danC*. Bandung: Penerbit Informatika.
- S, R. A. (2018). *Logika Algoritma dan Pemrograman Dasar*. Bandung: Modula.
- Sjukani, M. (2014). *Algoritma dan Struktur Data 1 dengan C, C++ dan Java* (Edisi 9 ed.). Jakarta: Mlta Wacana Media.
- Yuniati, S. (2012). MENENTUKAN KELIPATAN PERSEKUTUAN TERKECIL (KPK) DAN FAKTOR PERSEKUTUAN TERBESAR (FPB) DENGAN MENGGUNAKAN METODE “PEBI”. *Beta*, 149-165.

Ketua Program Studi
Teknik Informatika S-1

Syaeful Bakhri, ST., M.Eng.Sc., Ph.D
NIDN. 0421127402

Tangerang Selatan, 28 November 2019
Ketua Tim Penyusun
Mata Kuliah Algoritma dan Pemrograman 1

Hendri Ardiansyah, S.Kom., M.Kom
NIDN. 0401038601