



TIB26 – ALGORITMA PEMROGRAMAN

U N I V E R S I T A S B U N D A M U L I A

Capaian Pembelajaran Mata Kuliah

- Mahasiswa mampu menjelaskan konstruksi dasar algoritma, tipe data dan penulisan notasi algoritma (C2, A2)
- Mahasiswa mampu mengaplikasikan berbagai konsep dasar algoritma (C3, A3)
- Mahasiswa mampu menerapkan keyword dan syntax dasar bahasa pemrograman pada pembuatan program (C3, A3)
- Mahasiswa mampu menerapkan prinsip-prinsip dasar pemrograman dalam penyelesaian masalah pemrograman (C3, A3)



Pendahuluan

(TIB26 – ALGORITMA PEMROGRAMAN)

Pertemuan 1, 2

Sub-CPMK

- Mahasiswa mampu menjelaskan pengertian algoritma serta alat design algoritma (C2, A2)

Materi

- Pengertian Algoritma
- Kalimat Deskriptif
- Flowchart
- Pseudo-code

Perhatian

- Tidak disarankan copy-paste kode program dari presentasi ini, karena ada beberapa symbol yang dianggap sebagai Unicode oleh editor yang anda gunakan, sehingga akan dianggap sebagai symbol yang salah oleh compiler, sebaiknya diketik ulang saja



1.

Pengertian Algoritma

1.1. Pengertian Algoritma

- **Algoritma** adalah urutan langkah-langkah untuk menyelesaikan suatu persoalan.

1.2. Sejarah Singkat Algoritma

- Algoritma adalah inti dari ilmu komputer atau informatika. Ditinjau dari asal usul kata, kata “algoritma” sendiri mempunyai sejarah cukup panjang.
- Kata algoritma tidak muncul pada kamus Webster sampai akhir 1957. Kata yang ditemukan adalah algorism yang artinya: proses menghitung dengan angka arab 1,2,3...dst, sedangkan angka I,II,III...dst adalah angka romawi.

1.2 Sejarah Singkat Algoritma (lanj...)

- Kata algorism berasal dari nama penulis buku arab yang terkenal yaitu: Abu Ja'far Muhammad ibnu Musa al-Khuwarismi (al-Khuwarizmi dibaca orang Barat menjadi Algoritma).

1.2. Sejarah Singkat Algoritma (lanj...)

- Perubahan dari kata algorism menjadi algorithm muncul karena kata algorism sering dikelirukan dengan arithmetic, sehingga akhirnya –sm berubah menjadi –thm.
- Dalam bahasa Indonesia, kata algorithm diserap menjadi “algoritma”.
- Pada tahun 1950, kata algoritma pertama kali disandingkan pada “Algoritma Eculiden” (Eculid’s algorithm).
- Eculid, seorang matematikawan Yunani (lahir pada tahun 350 M).

1.2. Sejarah Singkat Algoritma (lanj...)



- Asal kata algoritma sendiri berasal dari nama Abu Ja'far Mohammed Ibn Musa al-Khowarizmi, ilmuwan Persia yang menulis buku berjudul “Al Jabr W’Al-Muqabala” (Rules of Restoration and Reduction) yang diterbitkan pada tahun 825 M.

1.3. Karakteristik Algoritma

- Algoritma sendiri memiliki beberapa ciri penting:
 - Algoritma harus berhenti setelah mengerjakan sejumlah langkah tertentu.
 - Setiap langkah harus didefinisikan dengan tepat dan tidak ambigu.
 - Algoritma memiliki masukan berjumlah nol atau lebih.

1.3. Karakteristik Algoritma (Lanj...)

- Algoritma memiliki keluaran berjumlah nol atau lebih.
- Algoritma harus efektif. Maksudnya setiap langkah yang tertulis harus sederhana sehingga dapat dikerjakan dalam waktu singkat dan masuk akal.

1.4 Pengertian Pemrograman

- Algoritma yang ditulis dalam bahasa komputer dinamakan **program**.
- Bahasa komputer yang digunakan untuk menulis program disebut **bahasa pemrograman**.
- Orang yang menulis program dinamakan **programmer**.
- Kegiatan mulai dari mendisain hingga menulis program dinamakan **pemrograman**.

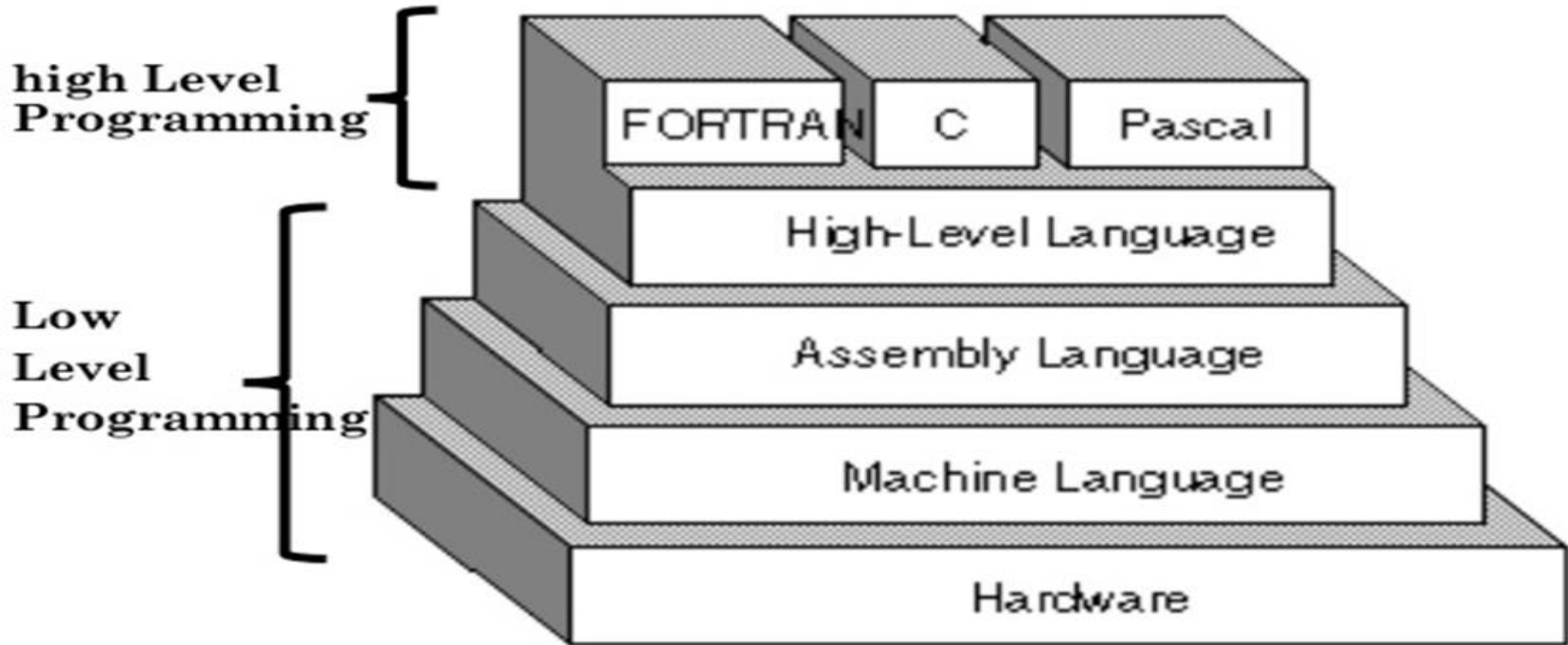
1.5. Bahasa Pemrograman

- Hingga saat ini jumlah bahasa pemrograman sangat banyak. Berdasarkan tujuan aplikasinya, bahasa pemrograman dapat digolongkan menjadi:
- Bahasa pemrograman bertujuan khusus (cobol, fortran, assembly, prolog, dll)
- Bahasa pemrograman bertujuan umum (C, C++, C#, Java, VB.Net, dll)

1.6. Kategori Bahasa Pemrograman

- Berdasarkan kedekatan bahasa pemrograman dengan bahasa alami (bahasa manusia), maka bahasa pemrograman juga dapat dikelompokkan menjadi:
 - Bahasa Tingkat Rendah (Low Level Language).
 - Bahasa tingkat menengah (Medium Level Language / MLL).
Contohnya: Assembly
 - Bahasa Tingkat Tinggi (High Level Language).
 - Dalam perkembangannya Bahasa pemrograman dikembangkan agar semakin mudah tanpa aturan-aturan yang kaku, dinamakan Bahasa pemrograman generasi 4

1.6. Kategori Bahasa Pemrograman (lanj...)



1.6. Kategori Bahasa Pemrograman (lanj...)

High-Level Language

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



Assembly Language

```
lw St0, 0(S2)
lw St1, 4(S2)
sw St1, 0(S2)
sw St0, 4(S2)
```

MIPS Assembler



Machine Language

0000	1001	1100	0110	1010	1111	0101	1000
1010	1111	0101	1000	0000	1001	1100	0110
1100	0110	1010	1111	0101	1000	0000	1001
0101	1000	0000	1001	1100	0110	1010	1111

1.6.1. Bahasa Tingkat Rendah

- Bahasa jenis ini dirancang agar setiap instruksinya langsung dikerjakan oleh komputer, tanpa harus melalui penerjemah (translator).
- Yang termasuk bahasa tingkat rendah adalah bahasa mesin.
- Bahasa mesin adalah sekumpulan kode binar (0 dan 1). Setiap bahasa mesin langsung dikenali dan dikerjakan.

1.6.1.1. Contoh Bahasa Tingkat Rendah

- Contoh bahasa tingkat rendah adalah bahasa mesin.

Program Fragment: $Y = Y + X$
Machine Language Code
(Binary Code)

Opcode	Adress
1100 0000	0010 0000 0000 0000
1011 0000	0001 0000 0000 0000
1001 0000	0010 0000 0000 0000

Memory Cell Definitions:

Addr.	Name	Cell Contentes
1000	X	32
2000	Y	16

1.6.2. Bahasa Tingkat Menengah

- Bahasa jenis ini dirancang agar lebih mudah dari bahasa mesin.
- Yang termasuk bahasa tingkat menengah adalah bahasa assembly.
- Dengan menggunakan bahasa assembly sedikit lebih mudah dalam membuat program bila dibandingkan dengan menggunakan bahasa mesin.
- Instruksi yang digunakan seperti: MOV, SUB, CMP, JMP, JGE, JL, LOOP, JNZ, dan yang lainnya

1.6.2.1 Contoh Bahasa Tingkat Menengah

- Contoh bahasa tingkat menengah, seperti bahasa assembler

Assembly Language

```
lw St0, 0(S2)
lw St1, 4(S2)
sw St1, 0(S2)
sw St0, 4(S2)
```

MIPS Assembler

1.6.3. Bahasa Tingkat Tinggi

- Bahasa jenis ini membuat program menjadi lebih mudah dipahami oleh manusia, karena secara sintaks lebih dekat dengan bahasa manusia.
- Kelemahannya, program tidak langsung dimengerti oleh komputer. Ia perlu diterjemahkan kedalam bahasa mesin sebelum dieksekusi oleh CPU.

1.6.3. Bahasa Tingkat Tinggi (lanj...)

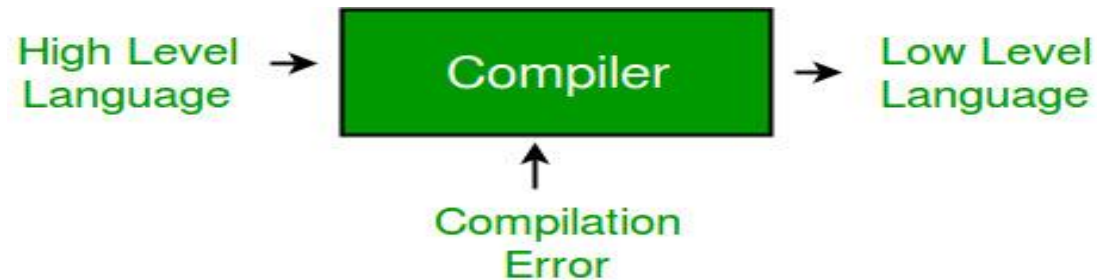
- Semua bahasa pemrograman, kecuali bahasa mesin dan bahasa assembly adalah bahasa tingkat tinggi (pascal, basic, C, C++, Java, dll).

High-Level Language

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

1.7. Cara Kerja Compiler & Interpreter

- **Compiler:** Program yang dibuat dalam bahasa tingkat tinggi (source code) diterjemahkan menjadi bahasa mesin.



- **Interpreter:** Program yang dibuat dalam bahasa tingkat tinggi (source code) diterjemahkan menjadi bahasa mesin per perintah (perbaris).

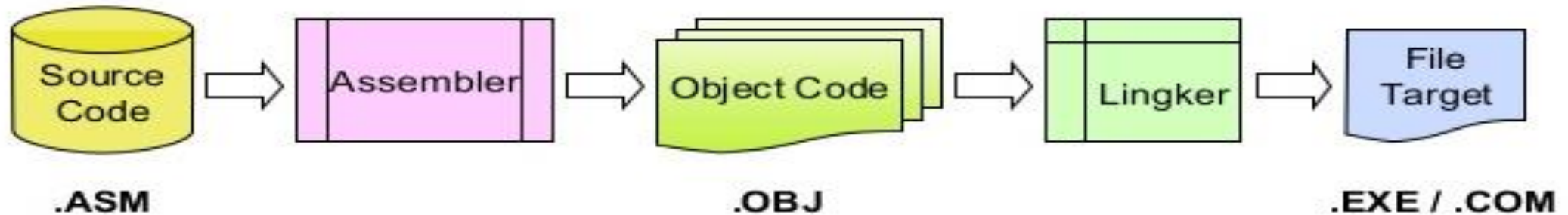
```
graph TD; PS[Program Sumber] --> PLS[Penganalisa Leksikal scanner]; PLS --> PAS[Penganalisa Sintaks parser]; PAS --> PSC[Penganalisa Semantik / Pembangkit Kode antara]; PSC --> PBK[Pembentuk kode]; PBK --> PKO[Pengoptimal kode]; PKO --> PSas[Program Sasaran]; PSas --> PS; TSS[TABEL SIMBOL] <--> PLS; TSS <--> PAS; TSS <--> PSC; TSS <--> PBK; TSS <--> PKO;
```

The diagram illustrates the compilation process, starting with **Program Sumber** (Source Program) and ending with **Program Sasaran** (Target Program). The process is divided into two main phases: **ANALISA** (Analysis) and **SINTESA** (Synthesis). The **ANALISA** phase includes **Penganalisa Leksikal (scanner)**, **Penganalisa Sintaks (parser)**, and **Penganalisa Semantik / Pembangkit Kode antara** (Semantic Analyzer / Intermediate Code Generator). The **SINTESA** phase includes **Pembentuk kode** (Code Generator) and **Pengoptimal kode** (Code Optimizer). A **TABEL SIMBOL** (Symbol Table) is used throughout the process, interacting with the scanner, parser, semantic analyzer, code generator, and code optimizer.

Bagan pokok proses kompilasi

1.7.2. Proses Kompilasi

Assembler



Proses Sebuah Kompilasi pada
Bahasa Assembler

- Source Code adalah bahasa Assembler, Object Code adalah bahasa mesin
- Object Code dapat berupa file object (.OBJ), file .EXE, atau file .COM
- Contoh : Turbo Assembler (dari IBM) dan Macro Assembler (dari Microsoft)

1.8. Bahasa Pemrograman Generasi 4

- Pada perkembangannya muncul Bahasa pemrograman generasi 4 dengan sintak yang lebih sederhana sehingga memudahkan programming.
- Beberapa Bahasa pemrograman generasi 4 antara lain; PHP, JAVA, Phyton, R dsb

1.9 Struktur Penulisan Algoritma

- **Judul algoritma;** Bagian yang terdiri atas nama algoritma dan penjelasan (spesifikasi) tentang algoritma tersebut. Nama sebaiknya singkat dan menggambarkan apa yang dilakukan oleh algoritma tersebut.
- **Deklarasi;** Bagian untuk mendefinisikan semua nama yang digunakan di dalam program. Nama tersebut dapat berupa nama tetapan, peubah, tipe, prosedur dan fungsi.

1.9. Struktur Penulisan Algoritma (Lanj...)

- **Deskripsi;** Bagian ini berisi uraian langkah-langkah penyelesaian masalah yang ditulis dengan menggunakan notasi yang akan dijelaskan selanjutnya.

1.10. Remark

- Remark merupakan bagian kode program yang tidak ikut dikompilasi
- Biasanya Remark digunakan untuk memberi catatan pada satu baris kode program atau Sebagian kode program
- Remark dapat diterapkan menggunakan `//` dan `{}`
- Penggunaan `//` untuk remark mengakibatkan kode di belakang tanda `//` tersebut tidak ikut dieksekusi
- Penggunaan tanda `{}` untuk remark mengakibatkan kode setelah tanda `{` sampai kode sebelum tanda `}` tidak akan ikut dikompilasi

1.10.1. Contoh remark

```
a = 10; //variable a diisi 10
b = a + 20; //jumlahkan variable a dg 20,
//simpan hasilnya di variable b
{
    catatan: kode di atas hanya contoh
    penggunaan remark
    meskipun dengan syntax yang sebenarnya
}
```

1.11. Konstruksi Dasar Algoritma

algoritma dibangun dari tiga konstruksi atau struktur dasar, yaitu:

- ***Sequence*** (runtunan/berurutan).
- ***Selection*** (pemilihan/pencabangan).
- ***Repetition*** (pengulangan/*loop*).

1.12. Alat Design Algoritma

- Kalimat Deskriptif
- Flowchart
- Pseudo-Code



2.

Kalimat Deskriptif

2. Kalimat Deskriptif

- Merupakan kalimat yang mendeskripsikan urutan perintah-perintah suatu algoritma dalam bentuk kalimat baku
- Setiap perintah dideskripsikan dalam bentuk kalimat
- Penambahan label nomor pada kalimat deskriptif kadang ada yang mengabaikan tetapi akan sangat membantu jika ada label nomor

2.1. Syntax Kalimat Deskriptif

- Dimulai dengan kalimat yang merupakan judul algoritma
- Berikutnya, Setiap barisnya berisi syntax berikut ini:

[Langkah N]

Kalimat perintah

N:nomor langkah

2.2. Contoh Kalimat Deskriptif

Mencetak angka 1 sampai 10

- [Langkah 1] inisialisasi n dengan 0
- [Langkah 2] ulangi Langkah 3 sampai Langkah 4 jika n kurang dari 10
- [Langkah 3] tambahkan nilai n dengan 1
- [Langkah 4] Cetak n
- [Langkah 5] Selesai



3.

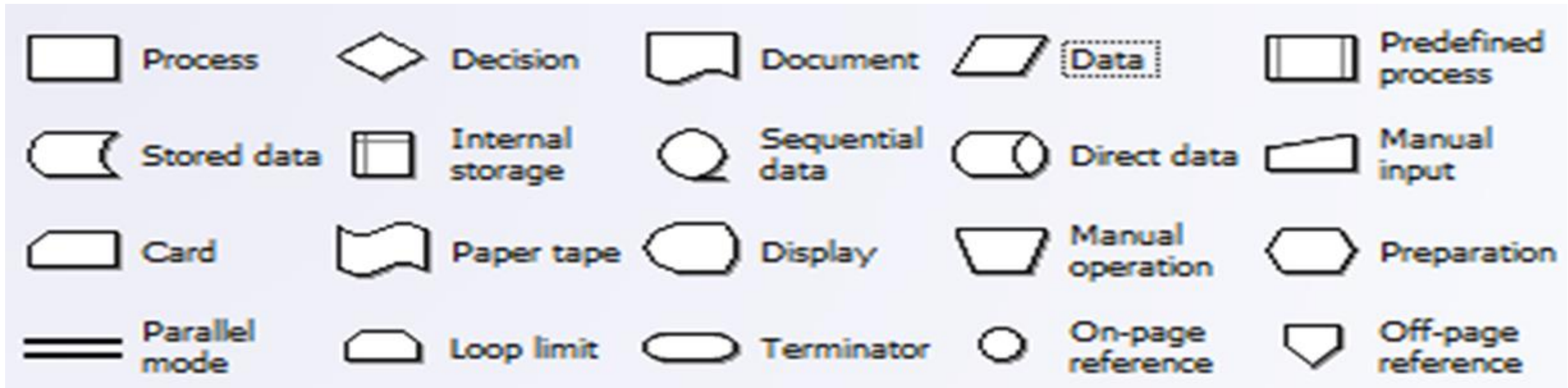
Flowchart

3. Flowchart (Diagram Alur)

- Notasi algoritma dalam bentuk diagram urutan perintah
- Setiap notasi disampaikan dalam bentuk bangun yang di dalamnya berisi teks yang merupakan variabel dan angka
- Flowchart diawali dengan perintah mulai dan selesai yang dilambangkan dengan notasi terminator berbentuk kapsul
- Salah satu keharusan dari flowchart adalah perintah mulai tunggal dan perintah selesai tunggal
- Alur Algoritma ditunjukkan dalam bentuk Panah

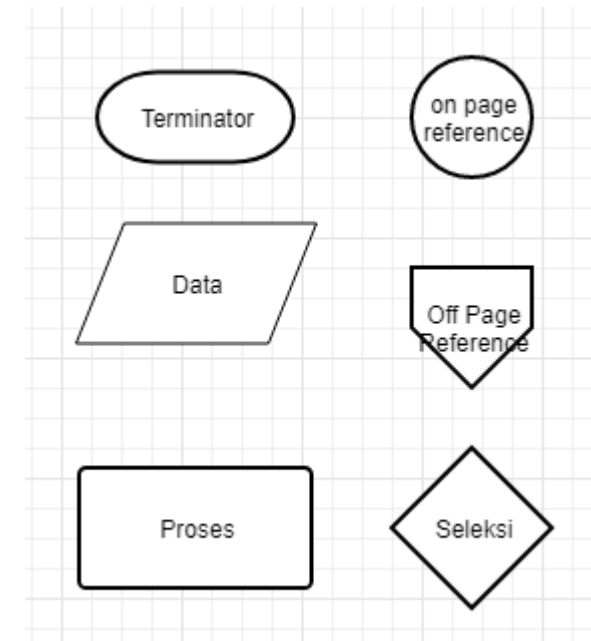
3.1. Notasi-Notasi Flowchart

- Notasi flowchart di hadirkan dalam bentuk bangun-bangun 2 dimensi, berupa persegi empat, persegi empat tumpul, belah ketupat, jajaran genjang dsb.



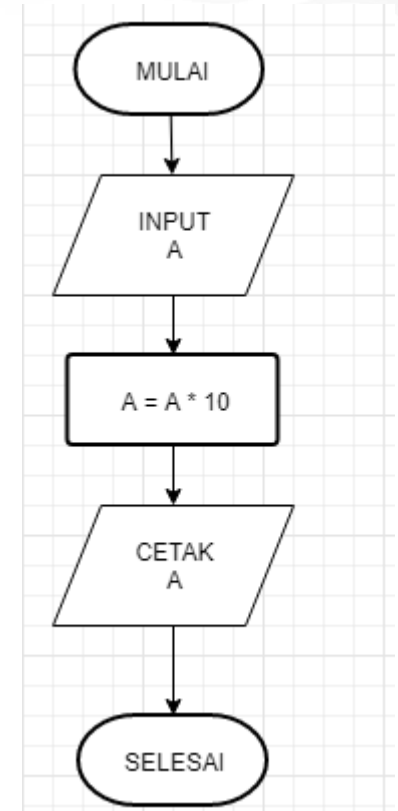
3.2. Notasi Flowchart Untuk Algoritma

- Semua notasi flowchart diperuntukkan bagi algoritma secara umum.
- Notasi-notasi Flowchart yang sering digunakan pada algoritma pemrograman adalah
 - Terminator
 - Data
 - Proses
 - Konektor
 - On-Page Reference
 - Off-Page Reference



3.3. Contoh Flowchart

- Selalu diawali dengan terminator “Mulai”
- Pernyataan Input menggunakan notasi Data berupa bangun jajaran genjang, disertai keterangan input, baca, dsb yang dapat mewakili pernyataan input
- Notasi Proses berisi rumus / perintah proses di dalam notasi berbentuk persegi empat
- Pernyataan Output menggunakan notasi Data disertai keterangan cetak/output/tulis dsb yang dapat mewakili pernyataan output
- Diakhiri dengan terminator “Selesai”



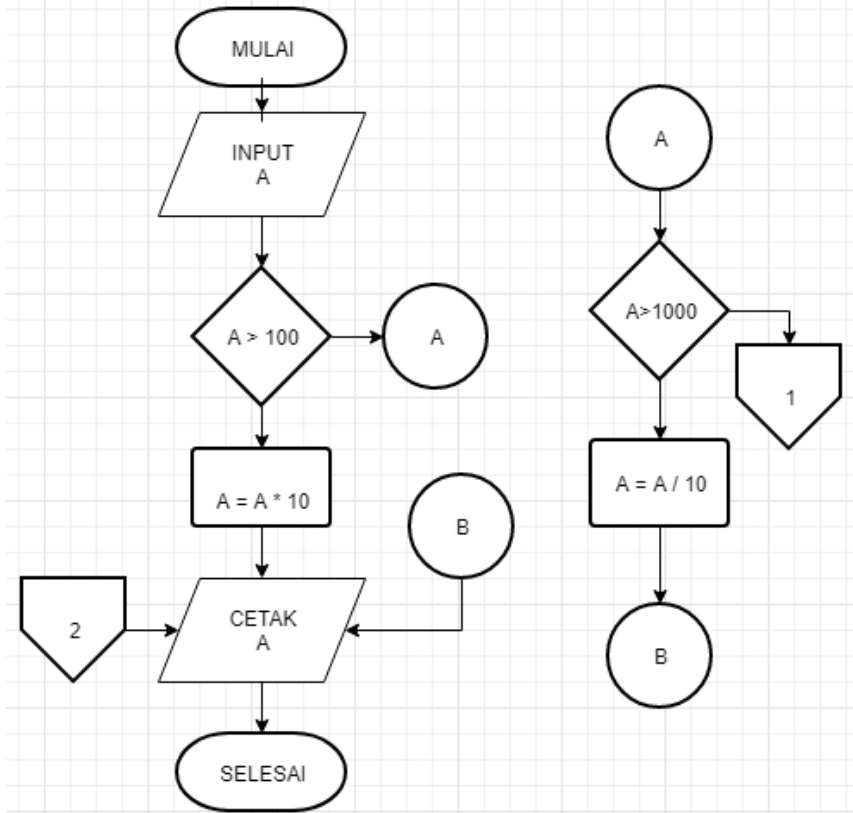
3.4. konektor

Konektor untuk menghubungkan suatu notasi flowchart ke notasi lainnya yang tidak dapat digambarkan langsung karena keterbatasan halaman

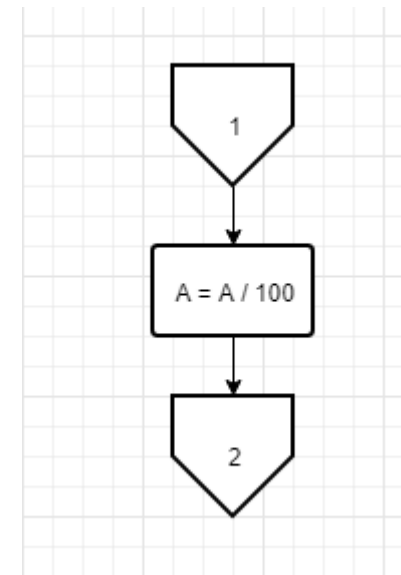
- On-Page Reference
 - Konektor ke flowchart pada halaman yang sama
 - Nomor label biasanya mempergunakan Huruf
- Off-Page Reference
 - Konektor ke flowchart pada halaman yang berbeda
 - Nomor Label biasanya mempergunakan angka

3.4.1 Contoh Penggunaan Konektor

- Halaman 1



- Halaman 2





4.

Pseudo-Code

4.1 Pseudo-Code

- Pseudo-code merupakan kode semu yang tidak terbatas pada syntax dan keyword
- Seringkali yang dipergunakan adalah syntax dan keyword dari suatu Bahasa pemrograman yang dikuasai oleh team tanpa memperdulikan kesalahan syntax dan keyword

4.2.susunan

- Tidak ada aturan baku untuk menyusun pseudo-code, tergantung perjanjian di dalam team
- Setidaknya Pseudo-code harus memiliki judul program
- Deklarasi variable tidak harus ada, tetapi jika ada cukup membantu
- Isi dari algoritma merupakan urutan perintah-perintah mempergunakan syntax dan keyword yang dapat dikenali team

4.3. Contoh Pseudo-code

PROGRAM Luas_lingkaran

DEKLARASI:

phi

luas, jari: real

ALGORITMA:

input(jari)

read (jari)

luas = $\text{phi} \times \text{jari} \times \text{jari}$

write (luas)



5.

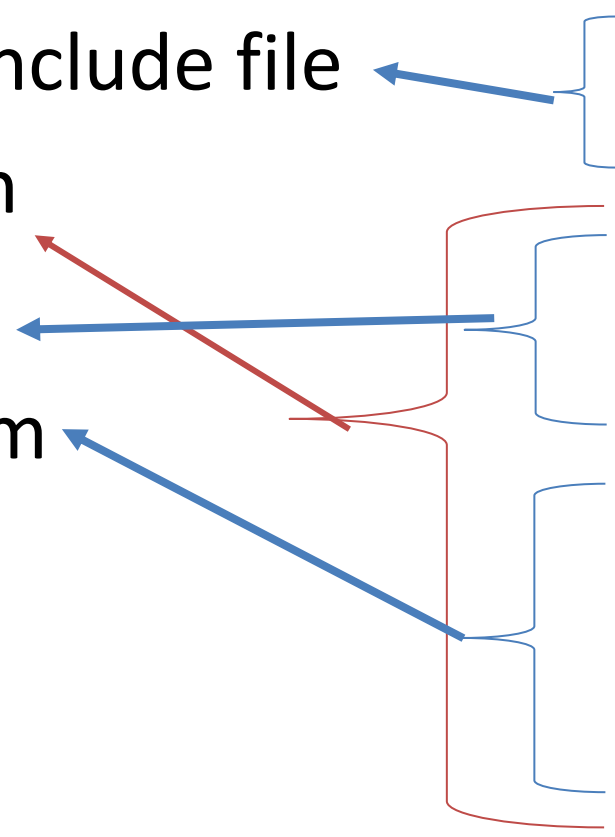
Struktur Bahasa C

5.1 Struktur Bahasa C

Kode Bahasa C secara dasar terdiri dari

- Pernyataan include file
- Blok Program
- Sub-Program
- Main Program

```
1  #include <iostream.h>
2  #include <conio.h>
3  #include <stdio.h>
4
5  int hitung(int a, b;)
6  {
7      return a+b;
8  }
9
10 int main()
11 {
12     clrscr();
13     cout<<"Biemers Is The Best"<<endl;
14     cout<<"Universitas Bunda Mulia"<<endl;
15     getch();
16     return 0;
17 }
```

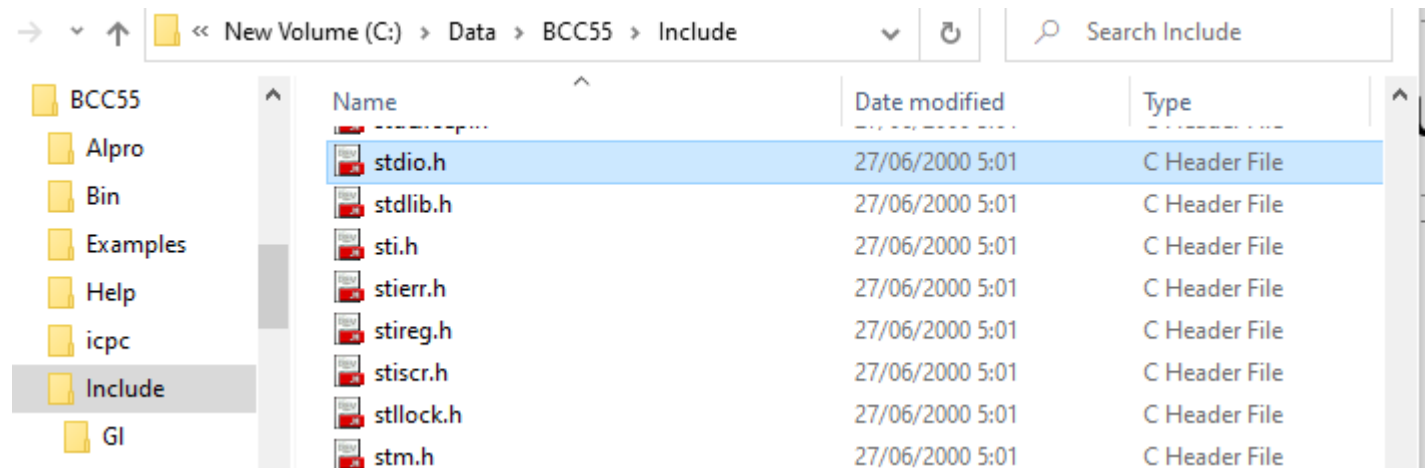


5.1. Pernyataan Include

- Blok ini mendefinisikan include dari header file yang akan dipergunakan.
- Header file ini berisi perintah-perintah untuk memanggil library-library yang diperlukan pada pemrograman C

5.2. File Header

- File Header umumnya terdapat pada folder Include.
- Semua file Header dapat dipanggil menggunakan keyword `#include <namaFile>`



5.3. Blok Program

- Pada Blok Program terdapat sub-program dan main program
- Main program terdapat fungsi `main()`;
- Ketika di kompilasi, maka `void main` akan dipanggil pertama kali setelah mengeksekusi pemanggilan `include` file.
- pada bagian blok program dapat di deklarasikan `variable-variable` yang diperlukan

5.4. Reserved Words C

- C mempunyai 32 buah kata yang dipesan (*reserved words*),
- Kata kunci kelompok pertama ini merupakan turunan dari Bahasa C pada C++

auto	const	double	float	int	short	struct	unsigned
break	continue	else	for	long	signed	switch	void
case	default	enum	goto	register	sizeof	typedef	volatile
char	do	extern	if	return	static	union	while

5.5 Reserved Words C++

- Kata yang dipesan kelompok kedua berjumlah 30. Kata-kata ini adalah baru dan hanya ada di bahasa C++

asm	dynamic_cast	namespace	reinterpret_cast	try
bool	explicit	new	static_cast	typeid
catch	false	operator	template	typename
class	friend	private	this	using
const_cast	inline	public	throw	virtual
delete	mutable	protected	true	wchar_t

Ringkasan

- **Algoritma** adalah urutan langkah-langkah untuk menyelesaikan suatu persoalan.
- Kalimat deskriptif merupakan kalimat yang mendeskripsikan urutan perintah-perintah suatu algoritma dalam bentuk kalimat baku
- Flowchart adalah Notasi algoritma dalam bentuk diagram urutan perintah
- Pseudo-code merupakan kode semu yang tidak terbatas pada syntax dan keyword



Terimakasih

TUHAN Memberkati Anda

Teady Matius Surya Mulyana (tmulyana@bundamulia.ac.id)