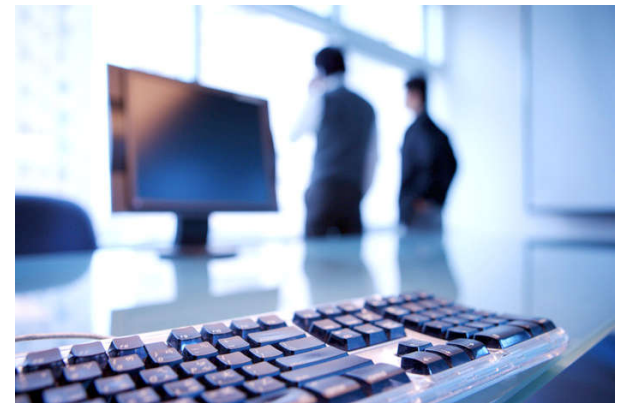


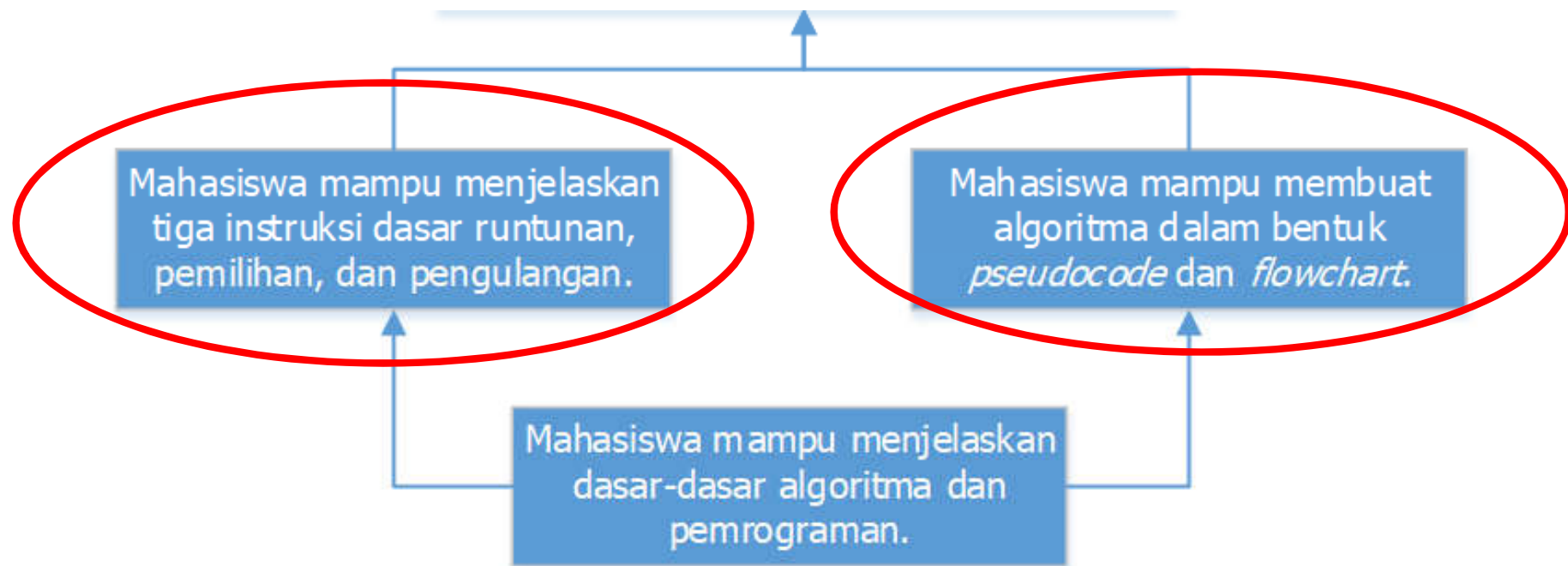
Konstruksi Dasar Algoritma

ALGORITMA DAN PEMROGRAMAN [IF6110202]

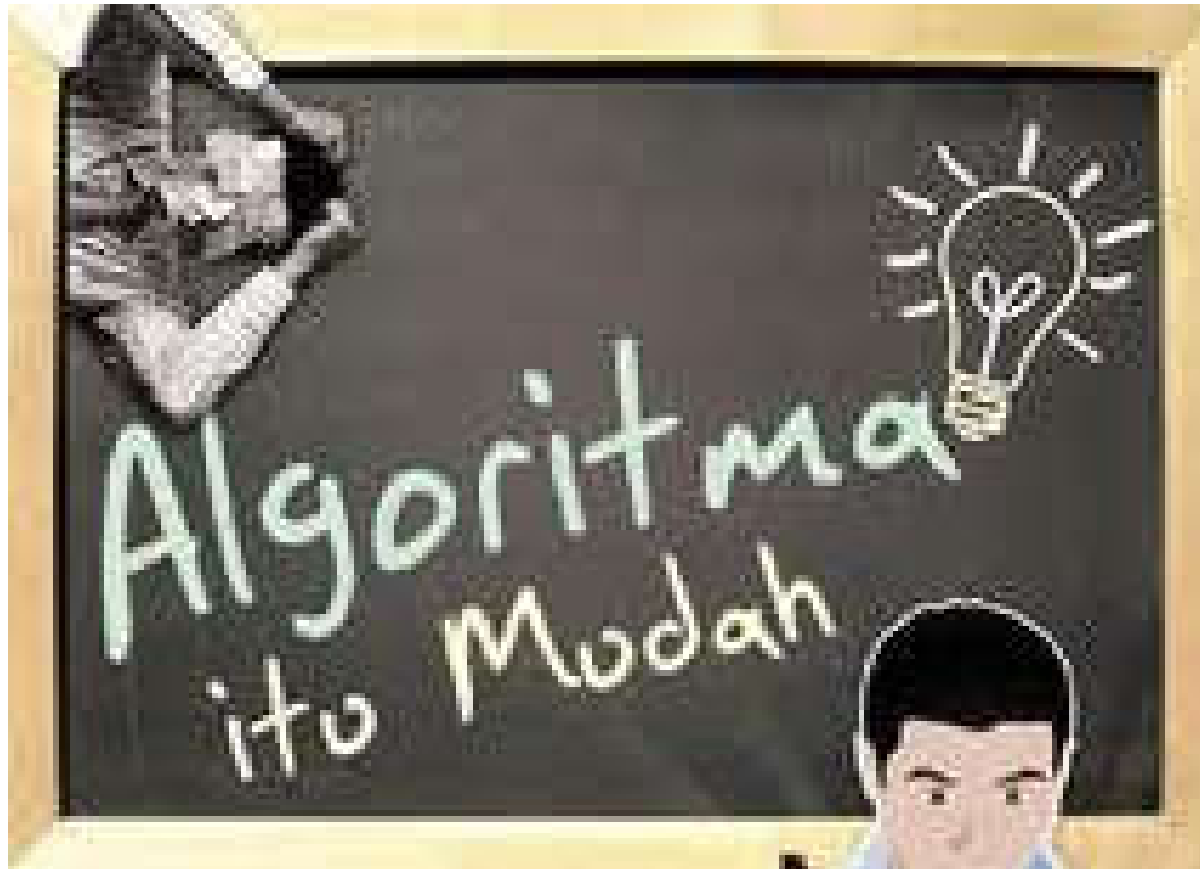
Agus Priyanto, S.Kom., M.Kom



Sub-Capaian Pembelajaran MK



Pendahuluan



Flow Chart

- ***Flow chart*** suatu bagan/diagram yang menggambarkan aliran proses yang dikerjakan program dari awal sampai akhir.
- ***Flow chart*** adalah algoritma yang digambarkan dengan diagram.
- Fungsi dari ***flow chart*** adalah mendeskripsikan urutan pelaksanaan suatu proses (sama dengan fungsi algoritma).



Flowchart Vs Algoritma

▪ *Flow Chart*

- *Flow chart* adalah suatu bagan/diagram yang menggambarkan aliran proses yang dikerjakan suatu program dari awal sampai akhir.
- *Flow chart* adalah algoritma yang digambarkan dengan diagram.
- Fungsi dari *flow chart* adalah mendeskripsikan urutan pelaksanaan suatu proses (sama dengan fungsi dari algoritma).

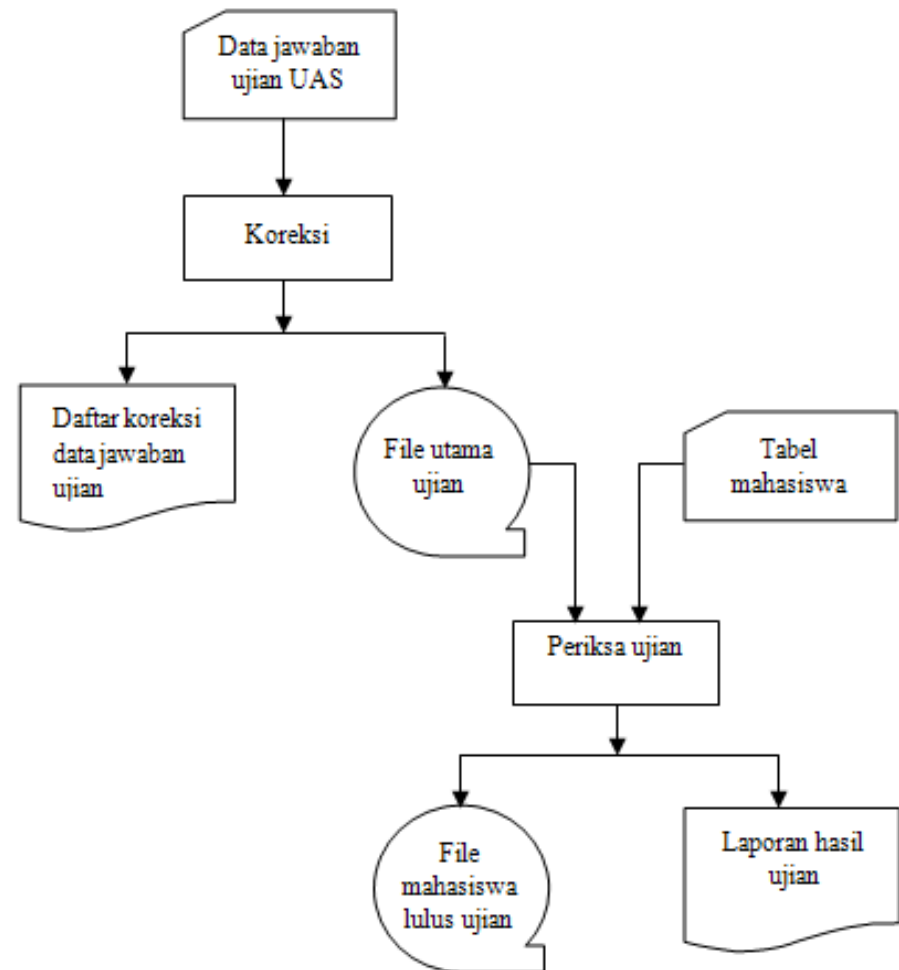
▪ *Algoritma*

- berisi langkah-langkah penyelesaian masalah yang ditulis dengan bahasa yang mudah dipahami.

Jenis Flowchart

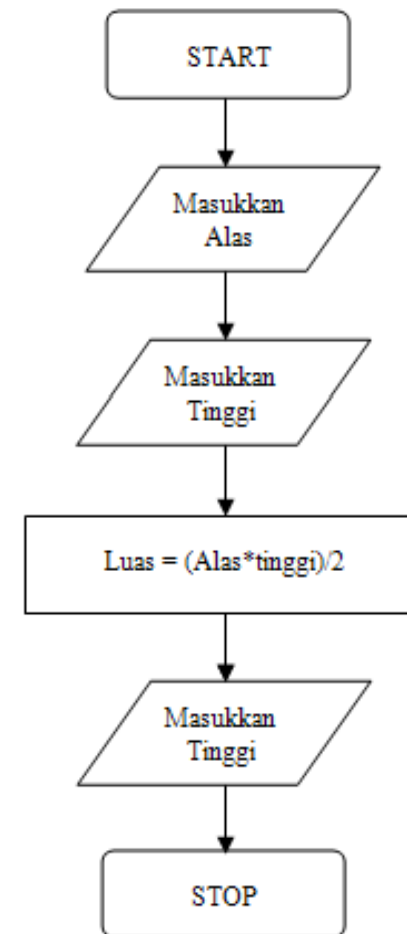
■ Sistem Flowchart

Urutan proses dalam sistem dengan menunjukkan alat media input, output serta jenis media penyimpanan dalam proses pengolahan data.



▪ Program Flowchart


Urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program










Pembuatan Flowchart

- Dalam pembuatan flowchart tidak ada kaidah yang baku.
 - Flowchart = gambaran hasil analisa suatu masalah.
 - Flowchart dapat bervariasi antara satu pemrogram dengan pemrogram lainnya.
- Secara garis besar ada 3 bagian utama:
 - Input
 - Proses
 - Output



- 
- Beberapa hal yang perlu diperhatikan dalam pembuatan flowchart, yaitu :
 - Hindari pengulangan proses yang tidak perlu dan logika yang berbelit sehingga jalannya proses menjadi singkat.
 - Jalannya proses digambarkan dari atas ke bawah dan diberikan tanda panah untuk memperjelas.
 - Sebuah flowchart diawali dari satu titik START dan diakhiri dengan END.



	<p>Terminator</p> <p>Sebagai simbol 'START' atau 'END' untuk memulai atau mengakhiri flowchart.</p>
	<p>Input/Output</p> <p>Digunakan untuk menuliskan proses menerima data atau mengeluarkan data</p>
	<p>Proses</p> <p>Digunakan untuk menuliskan proses yang diperlukan, misalnya operasi aritmatika</p>
	<p>Conditional / Decision</p> <p>Digunakan untuk menyatakan proses yang membutuhkan keputusan</p>
	<p>Preparation</p> <p>Digunakan untuk memberikan nilai awal</p>
	<p>Arrow</p> <p>Sebagai penunjuk arah dan alur proses</p>
	<p>Connector</p> <p>Digunakan untuk menyatukan beberapa arrow</p>

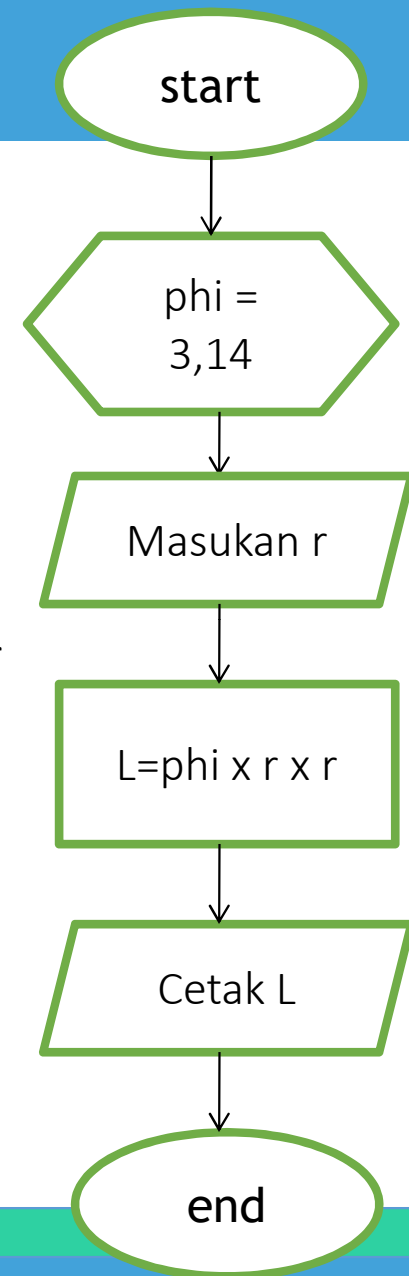
Flowchart
Simbol

Latihan I

Buatlah algoritma untuk menghitung luas dan keliling lingkaran. Dengan masukan jari-jari lingkaran.

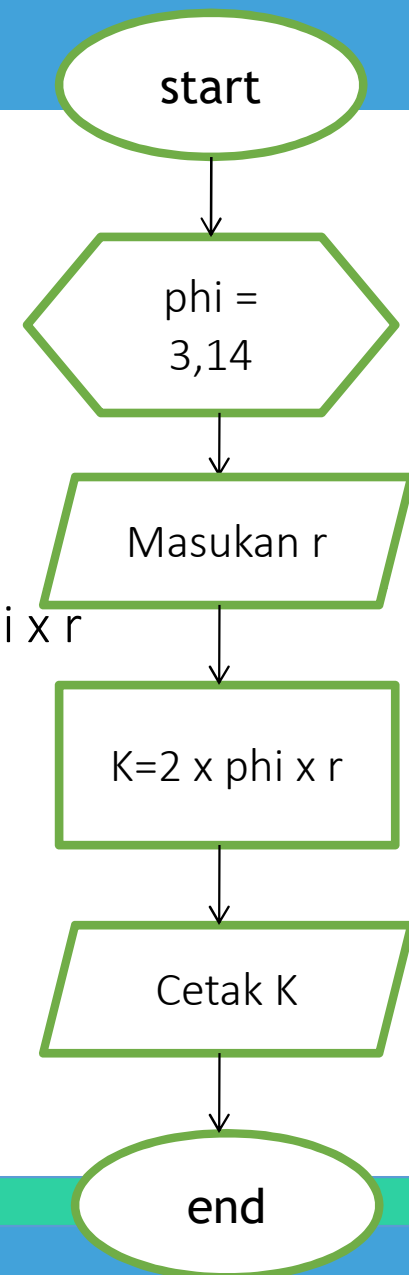
□ Menghitung Luas Lingkaran

1. Start
2. Menetapkan nilai Phi yaitu 3.14
3. Masukan jari-jari lingkaran
4. Menghitung luas lingkaran dengan rumus $L = \text{phi} \times r \times r$
5. Mencetak nilai Luas
6. Finish



□ Menghitung Keliling Lingkaran

1. Start
2. Menetapkan nilai Phi yaitu 3.14
3. Masukan jari-jari lingkaran
4. Menghitung keliling lingkaran dengan rumus $K = 2 \times \text{phi} \times r$
5. Mencetak nilai Keliling
6. Finish



Latihan 2

Buatlah algoritma untuk menentukan bilangan genap dan ganjil dari bilangan yang kita masukan



□ Menentukan Bilangan Genap Ganjil

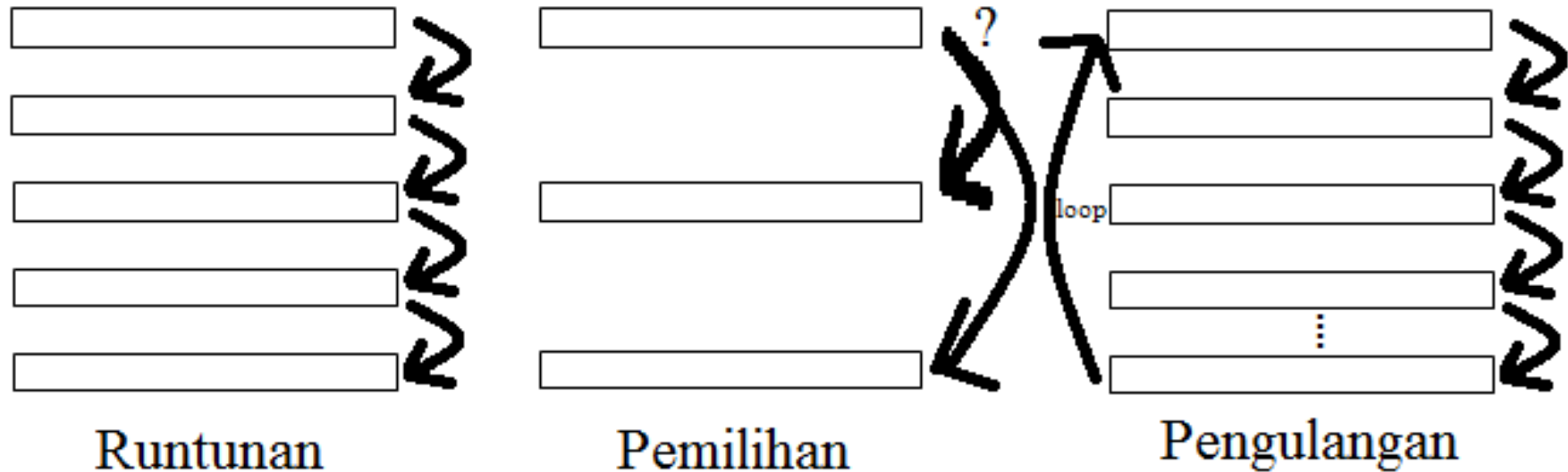
1. Start
2. Masukkan bilangan
3. Jika bilangan $\text{mod } 2 = 0$, maka genap
4. Jika bilangan $\text{mod } 2 = 1$, maka ganjil
5. Cetak bilangan
6. Finish

Instruksi dan Aksi

- Algoritma merupakan deskripsi urutan pelaksanaan suatu **proses**.
- Algoritma **tersusun** oleh sederetan langkah **instruksi** yang logis.
- Tiap langkah instruksi akan mengerjakan suatu **tindakan (aksi)**.
- Bila aksi dilaksanakan, maka sejumlah **operasi** yang bersesuaian akan dikerjakan oleh CPU.

Tiga Konstruksi Dasar

Sebuah algoritma dibangun dari tiga konstruksi dasar, yaitu **runtunan**, **pemilihan**, dan **pengulangan**



Tiga Konstruksi Dasar – Struktur Runtunan (*Sequence*) [1]

- Sebuah runtunan terdiri atas satu atau lebih pernyataan / aksi yang dikerjakan secara berurutan, berarti bahwa:
 1. Tiap instruksi dikerjakan satu per satu.
 2. Tiap instruksi dilaksanakan tepat satu kali; tidak ada instruksi yang di ulang.
 3. Urutan instruksi yang dilaksanakan pemroses (kompiler) sama dengan urutan instruksi sebagaimana yang tertulis di dalam teks algoritmanya.
 4. Akhir dari instruksi terakhir merupakan akhir algoritma.

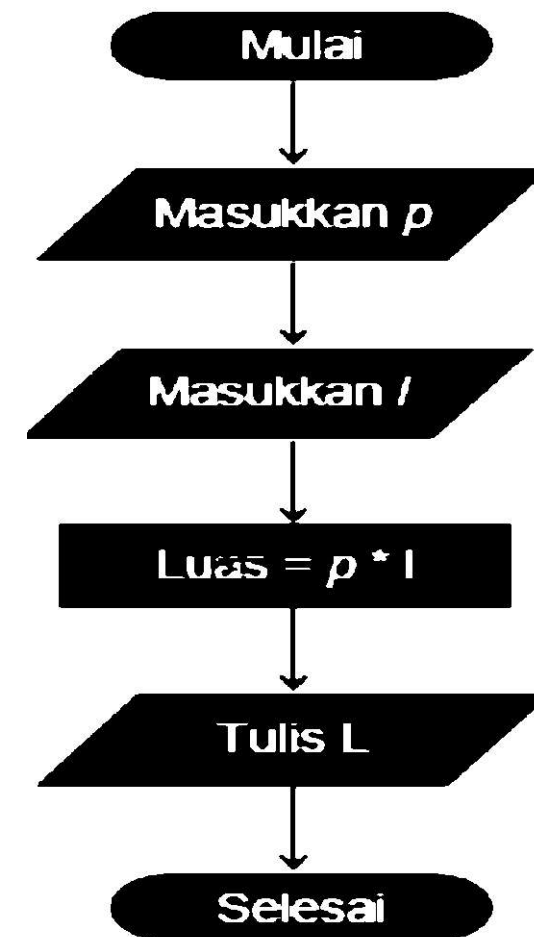


Tiga Konstruksi Dasar – Struktur Runtunan (Sequence) [2]

Contoh Algoritma

Menghitung Luas Persegi Panjang :

1. Masukkan panjang (p)
2. Masukkan lebar (l)
3. Hitung Luas ($p * l$)
4. Tulis Luas



Tiga Konstruksi Dasar – Struktur Runtunan (*Sequence*) [3]

- Urutan instruksi dalam algoritma adalah penting. Urutan instruksi menunjukkan urutan logika penyelesaian masalah.
- Urutan instruksi yang berbeda mungkin tidak ada pengaruh terhadap solusi persoalan, tetapi mungkin juga menghasilkan keluaran yang berbeda, tergantung pada masalahnya



Tiga Konstruksi Dasar – Struktur Runtunan (*Sequence*) [4]

Contoh urutan instruksi yang berbeda tetapi tidak mempengaruhi hasil.

- **Deklarasi :**
A, B, C, D : integer
Deskripsi :
 1. **read (A, B) {1}**
 2. **C A + B {2}**
 3. **D A * B {3}**
 4. **write (C, D) {4}**

- **Deklarasi :**
A, B, C, D : integer
Deskripsi :
 1. **read (A, B)**
 2. **D A * B**
 3. **C A + B**
 4. **write (C, D)**



Tiga Konstruksi Dasar – Struktur Runtunan (*Sequence*) [4]

Contoh urutan instruksi yang berbeda tetapi mempengaruhi hasil.

- {di baca dua buah bilangan integer kemudian hitung penjumlahan dan perkalian dua buah bilangan tersebut, dan tampilkan hasilnya ke layar}

Deklarasi :

A, B, C, D : integer

Deskripsi :

C A + B

D A * B

read (A, B)

write (C, D)

Hasil C dan D akan
berbeda dengan dua
algoritma sebelumnya

Tiga Konstruksi Dasar – Struktur Pemilihan (*Selection*) [1]

- memungkinkan suatu Aksi dieksekusi jika suatu **kondisi terpenuhi** atau **tidak terpenuhi**.
- struktur pemilihan mampu memungkinkan pemroses mengikuti jalur aksi yang berbeda berdasarkan kondisi yang ada.
- Tidak setiap baris program akan dikerjakan.
- Baris program akan dikerjakan jika memenuhi syarat.
- Jadi, struktur pemilihan adalah : struktur program yang melakukan proses pengujian untuk mengambil suatu keputusan apakah suatu baris program atau blok instruksi akan diproses atau tidak.
- Pengambilan keputusan menggunakan pernyataan boolean (true/false) dg menggunakan operator pembandingan (>, <, >=, <=, =, <>) yang bisa di kombinasikan dengan operator boolean (AND, OR dan NOT).

Tiga Konstruksi Dasar – Struktur Pemilihan (*Selection*) [2]

Contoh :

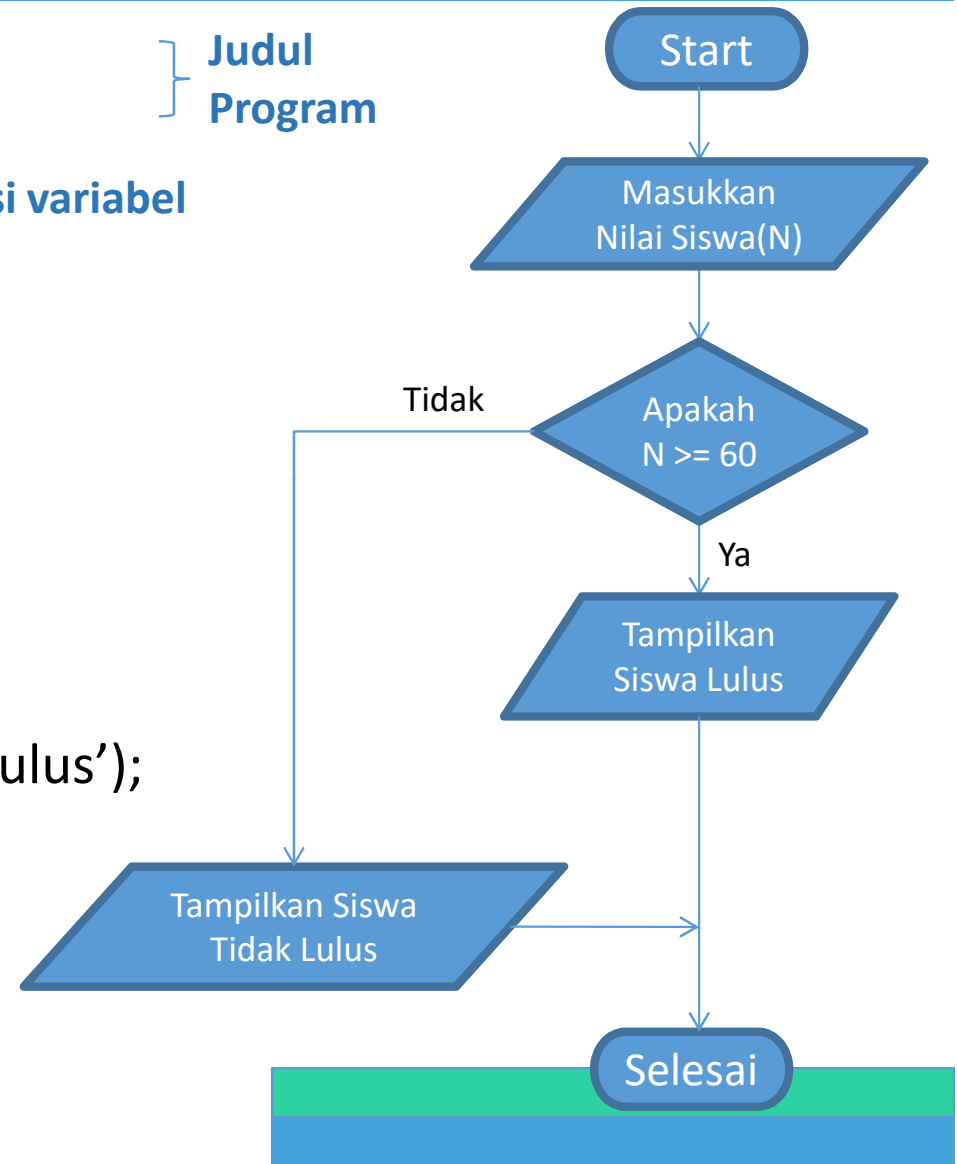
- $5 = 5 \rightarrow \text{true}$, karena 5 sama dengan 5
- $3 = 4 \rightarrow \text{false}$, karena 3 tidak sama dengan 4
- $3 > 1 ?$
- $5 <> 2 ?$
- $A = 5 \rightarrow$ bisa true/false tergantung nilai variabel A
- $(A > 5) \text{ AND } (B = 2) \rightarrow \text{true}$, jika pernyataan $A > 5$ bernilai true, dan pernyataan $B = 2$ juga bernilai true

Tiga Konstruksi Dasar – Struktur Pemilihan (*Selection*) [2]

1. Program penentuan_lulus;
2. Var
3. N: integer;
4. Begin
5. Write('Masukkan Nilai Siswa');
6. Read(N);
7. If $N \geq 60$ Then
8. Write('Siswa Dinyatakan Lulus');
9. Else
10. Write('Siswa Dinyatakan Tidak Lulus');
11. Readln;
12. End.

} Judul
Program

} Deklarasi variabel



Tiga Konstruksi Dasar – Struktur Pemilihan (*Selection*) [3]

Keterangan :

- Tidak semua baris program akan diproses
- Baris program no.8 akan diproses jika kondisi nilai siswa ≥ 60 bernilai benar (true)
- Baris program no.10 akan diproses jika kondisi nilai siswa ≥ 60 bernilai salah (false)



Tiga Konstruksi Dasar – Pengulangan (*Repeatition*)

[1]

- Salah satu kelebihan komputer adalah mampu mengerjakan pekerjaan yang sama berulang kali tanpa kenal lelah
- Struktur pengulangan memungkinkan kita untuk membuat suatu algoritma dari instruksi yang berulang-ulang lebih efektif
- Contoh : mencetak suatu kalimat sebanyak 100 kali



Tiga Konstruksi Dasar – Pengulangan (*Repeatition*)

[2]

Pengulangan adalah instruksi yang dapat mengulang sederetan instruksi secara berulang-ulang sesuai persyaratan yang ditetapkan.

- Salah satu kelebihan komputer adalah mampu mengerjakan pekerjaan yang sama berulang kali tanpa kenal lelah
- Struktur pengulangan memungkinkan kita untuk membuat suatu algoritma dari instruksi yang berulang-ulang lebih efektif
- Contoh : mencetak suatu kalimat sebanyak 100 kali

Tiga Konstruksi Dasar – Pengulangan (*Repetition*)

[3]

Struktur instruksi perulangan pada dasarnya terdiri atas :

- Kondisi perulangan; suatu kondisi yang harus dipenuhi agar perulangan dapat terjadi.
- Badan (body) perulangan; deretan instruksi yang akan diulang-ulang pelaksanaannya.
- Pencacah (counter) perulangan; suatu variabel yang nilainya harus berubah agar perulangan dapat terjadi dan pada akhirnya membatasi jumlah perulangan yang dapat dilaksanakan

Tiga Konstruksi Dasar – Jenis Pengulangan

1. For – Next
2. While – Do
3. Repeat - Until

Pengulangan: FOR – NEXT [1]

Bentuk umum :

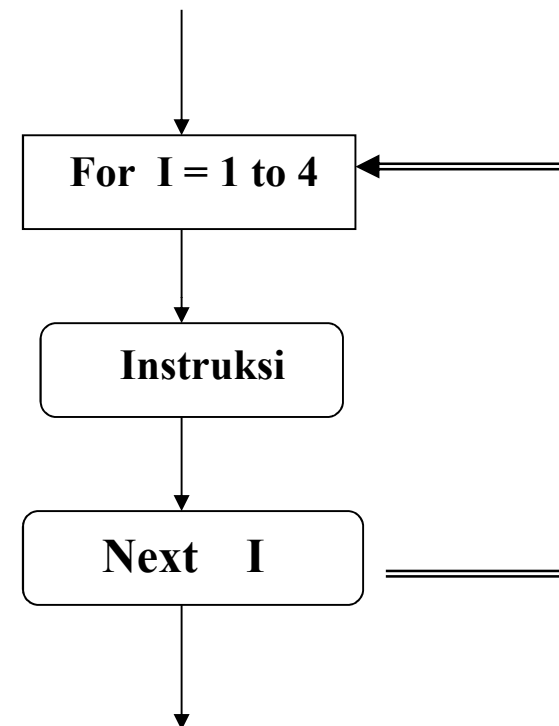
For var=awal **to** akhir

.....

instruksi-instruksi

.....

Next var



Pengulangan: FOR – NEXT [2]

Makna dari bentuk perulangan di atas adalah ulangi instruksi tersebut berdasarkan variabel perulangan mulai dari nilai awal hingga nilai akhir.

Dari gambar di atas instruksi akan dikerjakan sebanyak 4 kali, karena perulangan dimulai dari 1 sampai 4.

Pengulangan: WHILE – DO [1]

Bentuk umum :

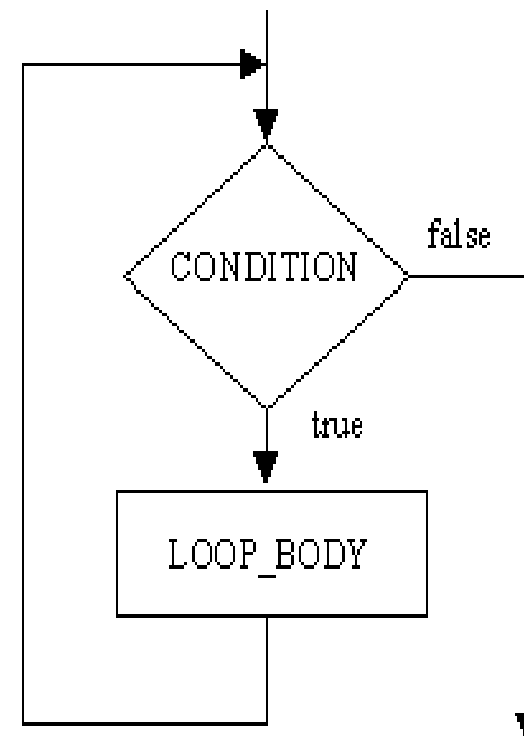
While {kondisi} **do**

.....

instruksi-instruksi

.....

Endwhile



Pengulangan: WHILE – DO [2]

Dari bentuk pengulangan di atas instruksi akan dilaksanakan berulang kali selama kondisi bernilai TRUE , jika FALSE maka badan pengulangan tidak akan dilaksanakan yang berarti pengulangan selesai.

- Algoritma Cetak_Angka
- {mencetak 1, 2, .., 8 ke piranti keluaran}
- Deklarasi :
 - K: integer
- Deskripsi :
 - K = 1 {inisialisasi}
 - while k <= 8 do
 - write (k)
 - k = k + 1
- endwhile

Pengulangan: REPEAT – UNTIL [1]

Instruksi (atau runtunan instruksi) akan dilaksanakan berulang kali sampai kondisi bernilai true, jika kondisi bernilai false maka pengulangan masih terus dilakukan

- Algoritma Cetak_Angka
- {mencetak 1, 2, .., 8 ke piranti keluaran}
- Deklarasi :
 - K: integer
- Deskripsi :
 - $K = 1$ {inisialisasi}
 - repeat
 - write (k)
 - $k = k + 1$
 - until $k > 8$

Notasi Pseudo-Code

- Kode atau tanda yang menyerupai (**pseudo**) program atau merupakan penjelasan cara menyelesaikan suatu masalah.
- **Pseudocode** sering digunakan oleh manusia (*programmer*) untuk menuliskan algoritma sebab mudah dipahami dan digunakan karena mirip dengan kode-kode program sebenarnya.

Struktur Pseudo-code

PROGRAM Nama Program

{Penjelasan tentang algoritma, berisi uraian singkat mengenai masalah yang akan diselesaikan}

DEKLARASI

{semua nama yang dipakai, meliputi nama tipe, nama konstanta, nama peubah, nama prosedur, dan nama fungsi}

ALGORITMA

{semua langkah/aksi algoritma dituliskan disini}

Format Pseudo-code Lengkap

Judul program/algorithm

PROGRAM Euclidean

*Program untuk mencari GCD dari dua buah bilangan bulat positif m dan n ($m \geq n$).
GCD dari m dan n adalah bilangan bulat positif terbesar yang habis membagi m dan n*

DEKLARASI:

m, n : integer
 r : integer

{bil bulat}
{sis hasil bagi}

Deklarasi variable

→ komentar

Algoritma:

read(m, n) { $m \geq n$ }

while $n \neq 0$ do

$r \leftarrow m \text{ MOD } n$

$m \leftarrow n$

$n \leftarrow r$

end while

{kondisi selesai pengulangan $n=0$, maka $\text{gcd}(m, n) = m$ }

Algoritma

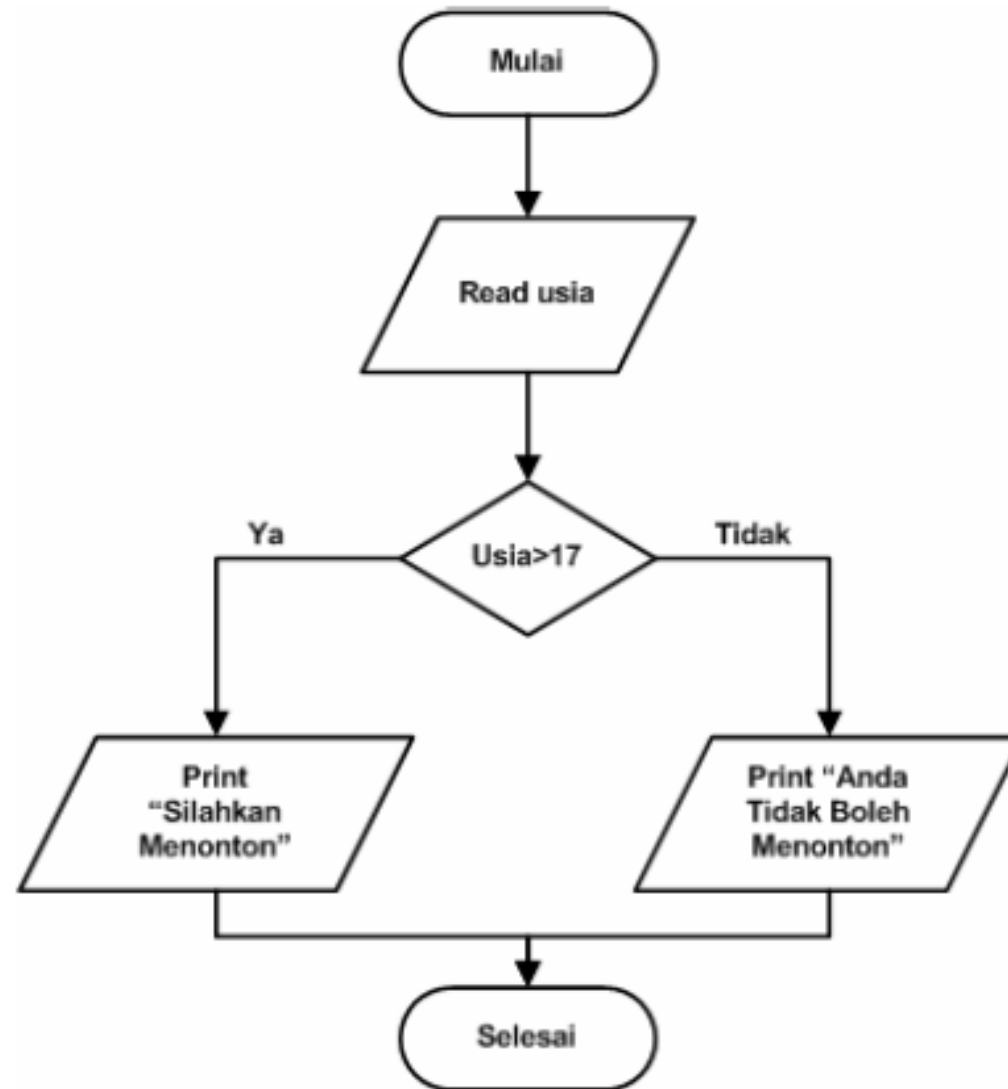
Latihan

1. Buatlah algoritma untuk menentukan bilangan terbesar dari 3 buah bilangan!!
2. Buatlah flowchart untuk aturan menonton sebuah film tertentu sebagai berikut, jika usia penonton lebih dari 17 tahun maka penonton diperbolehkan dan apabila kurang dari 17 tahun maka penonton tidak diperbolehkan nonton.
3. Buat algoritma menampilkan deret 2, 4, 6, ..., N. N adalah masukan berupa bilangan genap.

Pembahasan No 1

```
01|  ALGORITMA Menentukan_terbesar_dari_3_bilangan
02|  Deklarasi:
03|  a,b,c, terbesar : integer
04|
05|  Deskripsi:
06|    Read(a,b,c)
07|    If (a>b) and (a>c) then
08|        Terbesar ← a
09|    Else
10|        If b>c then
11|            Terbesar ← b
12|        Else
13|            Terbesar ← c
14|        Endif
15|    Endif
16|    Write(terbesar)
```


Pembahasan No 2



Pembahasan No 3

Algoritma deret

Deklarasi

$N, x : \text{integer}$

Deskripsi

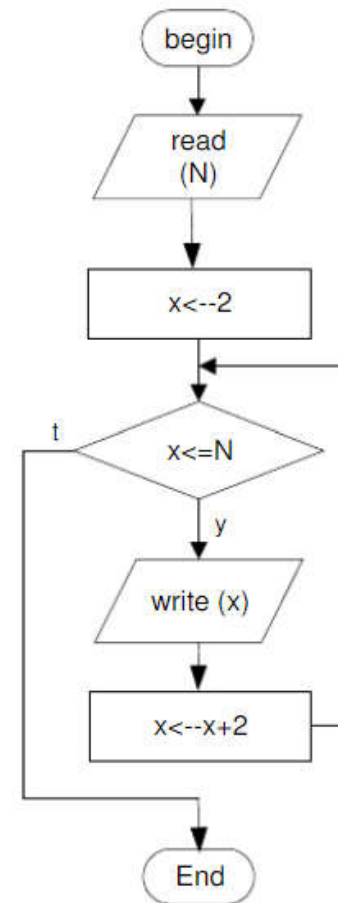
read(N)

$x \leftarrow 2$

while $x \leq N$ do

Write(x)

$x \leftarrow x + 2$



Tugas

- Silahkan bentuk kelompok yang terdiri dari anggota 3 orang
- Buatlah algoritma dalam *flowchart* dan *pseudocode* untuk suatu masalah tertentu
- Silahkan tentukan sendiri masalahnya
- Dikirim melalui email agus_priyanto@ittelkom-pwt.ac.id nanti malam maksimal pukul 23.59 WIB
- Pertemuan selanjutnya presentasi tiga kelompok dipilih secara acak

Kesimpulan

- Sebuah algoritma pada hakekatnya berisi sekumpulan **instruksi** yang menggambarkan langkah-langkah penyelesaian suatu persoalan.
- Instruksi adalah perintah untuk melakukan **aksi** tertentu.
- Di dalam bahasa pemrograman, instruksi dinyatakan sebagai **pernyataan**.
- Sebuah algoritma dibangun dari tiga konstruksi dasar, yaitu **runtunan (*sequence*)**, **pemilihan (*selection*)**, dan **pengulangan (*repetition*)**.
- Sebuah runtunan (*sequence*) terdiri atas satu atau lebih pernyataan yang dikerjakan secara berurutan.
- Pada penyeleksian (*selection*), sebuah aksi dikerjakan jika kondisi tertentu terpenuhi.
- Pada pengulangan (*repetition*), memungkinkan banyak aksi dikerjakan dengan satu instruksi.

