**Perulangan 1**

Tim Ajar Dasar Pemrograman 2023

# Tujuan

Di akhir pertemuan, mahasiswa diharapkan mampu :

* Memahami algoritma perulangan (for, while, do-while)
* Memberikan contoh sederhana perulangan
* Menggambarkan permasalahan studi kasus perulangan dengan menggunakan flowchart

# Definisi Perulangan

* Perintah perulangan atau iterasi (loop) adalah perintah untuk mengulang satu atau lebih statement sebanyak beberapa kali
* Loop statement digunakan agar kita tidak perlu menuliskan satu/sekumpulan statement berulang-ulang. Dengan begitu maka kesalahan pengetikan bisa dikurangi
* Tipe perulangan:
* Definite loop
* Indefinite loop

# Tipe Perulangan – Definite Loop

* Perulangan yang jumlah eksekusinya **telah diketahui sebelumnya**
* Biasanya ditandai dengan “ulangi sebanyak \_\_ kali”
* Contoh:
* Ulangi pernyataan ini sebanyak n kali
* Ulangi pernyataan ini untuk setiap bilangan genap antara 8 dan 26

# Tipe Perulangan – Indefinite Loop

* Perulangan yang jumlah eksekusinya **tidak dapat ditentukan sebelum dilakukan**
* Perulangan dieksekusi selama kondisi bernilai benar (TRUE), atau sampai kondisi menjadi salah (FALSE)
* Contoh:
* Ulangi pernyataan ini selama bilangan n bukan bilangan prima
* Ulangi pernyataan ini sampai pengguna memasukkan bilangan bulat yang valid

# Jenis Perintah Perulangan

Dalam bahasa Java, ada 3 macam perintah perulangan yang umum digunakan yaitu: • Perintah FOR

* Perintah WHILE
* Perintah DO-WHILE

|  |
| --- |
| **Struktur Perulangan FOR** |

# Perulangan FOR

* FOR umumnya digunakan pada pengulangan yang jumlah perulangannya sudah pasti atau sudah diketahui sebelumnya
* Sintaks FOR

for (inisialisasi; kondisi; update) statement; atau:

for (inisialisasi; kondisi; update){ statement1; statement2;

……

}

# Perulangan FOR

* inisialisasi: deklarasi dan inisialisasi variabel counter

(variabel pengontrol perulangan)

* kondisi: batas atau syarat agar perulangan tetap dieksekusi
* update: perubahan nilai variabel counter pada setiap putaran perulangan (increment atau decrement)

inisialisasi dan update bersifat optional (boleh ada atau tidak)

# Alur Perulangan FOR

1. Perulangan diawali dengan melalukan inisialisasi
2. Evaluasi kondisi
   * Jika kondisi bernilai TRUE, eksekusi semua statement di dalam perulangan. Lakukan update. Ulangi kembali langkah nomor 2
   * Jika kondisi bernilai FALSE, hentikan perulangan

# Flowchart FOR

**inisialisasi**

**update**

**statements**

**kondis**

**i**

**true**

**false**

# Contoh Perulangan FOR

i

1

=

i

++

Hasil

kuadrat

dari

i

adalah

i

\*

i

I

<

=

5

true

false

Start

Stop

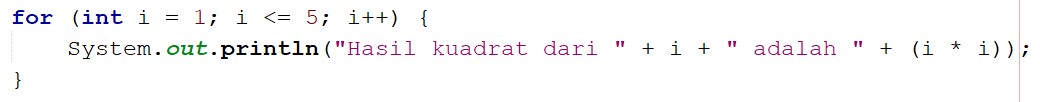
int

i

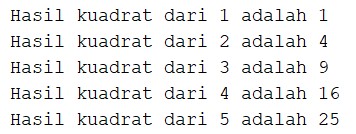
Buatlah flowchart dan kode program untuk menampilkan bilangan dan hasil kuadratnya dengan rentang nilai bilangan 1 sampai 5!

# Contoh Perulangan FOR

Kode Program



Output

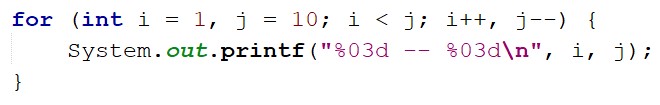


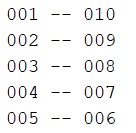
# Variasi Perulangan FOR – Variasi 1

inisialisasi dan update boleh terdiri dari beberapa ekspresi yang dipisahkan dengan tanda koma

Contoh

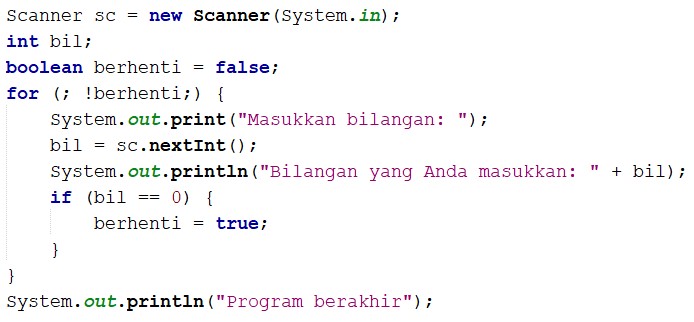
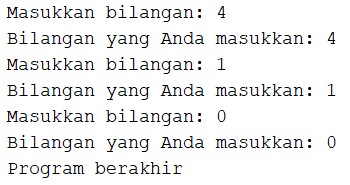
:



Output 

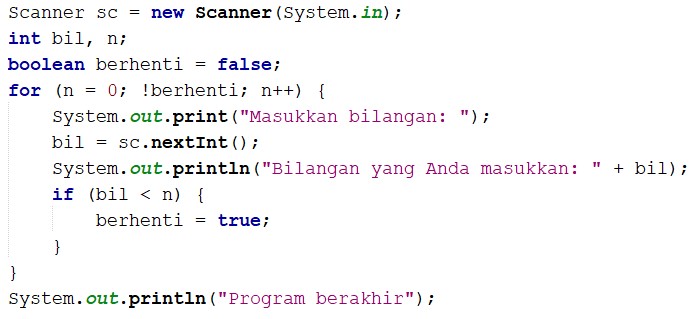
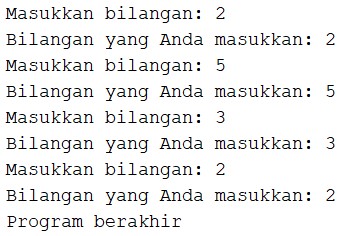
# Variasi Perulangan FOR – Variasi 2

•inisialisasi dan update dapat dikosongi, sesuai dengan kebutuhan •Contoh:

 Output

# Variasi Perulangan FOR – Variasi 3

Seperti halnya kondisi dalam if, kondisi pada for juga dapat menggunakan variable bertipe boolean Contoh:

 Output

|  |
| --- |
| **Struktur Perulangan WHILE** |

# Perulangan WHILE

* WHILE cocok digunakan untuk perulangan yang jumlahnya tidak diketahui sebelumnya (indefinite loop)
* Sintaks WHILE

|  |
| --- |
| • Perulangan dengan while akan terus dijalankan selama kondisi bernilai TRUE |

inisialisasi; while (kondisi){ statement1; statement2;

… update;

}

# Flowchart WHILE

**inisialisasi**

**update**

**statements**

**kondisi**

**true**

**false**

# Perbandingan FOR dan WHILE

WHILE FOR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | inisialisasi***;***  **while (**kondisi**)**  **{ *statement1; statement2;***  ***…*** update  **}** | | **setara** | |  | | --- | | **for (**inisialisasi**;** kondisi***;*** update**)**  **{ *statement1; statement2;***  ***…***  **}** | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | **int x = 1;**  **while (x <= 10) {**  **\_\_\_\_\_\_ \_\_\_\_\_\_**  **x++;**  **}** | | **sama** | |  | | --- | | **for(int x = 1 ; x <= 10 ; x++)**  **\_\_\_\_\_**  **\_\_\_\_\_**  **}** | |

Contoh:

# Contoh Perulangan WHILE

i = 1

i++

Angka i adalah

bilangan genap

i

<=

10

true

false

Start

Stop

i

%2

==

0

Angka i adalah

bilangan ganjil

true

false

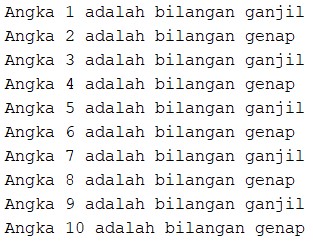
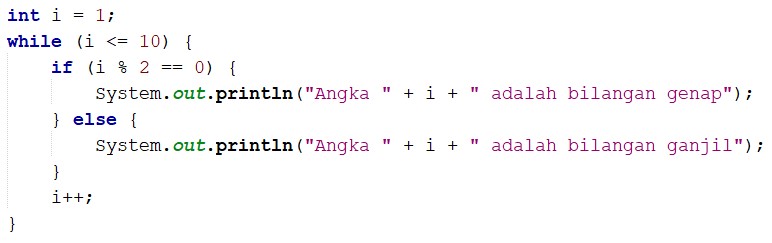
int

i

Buatlah flowchart dan kode program untuk menampilkan keterangan bilangan ganjil dan genap dengan rentang nilai bilangan 1 sampai 10!

# Contoh Perulangan WHILE

Kode Program Output



|  |
| --- |
| **Struktur Perulangan DO-WHILE** |

# Perulangan DO-WHILE

•Pada prinsipnya, perintah DO-WHILE sama dengan perintah WHILE

•Perbedaanya:

* DO-WHILE **mengeksekusi statementnya terlebih dahulu**, lalu mengevaluasi kondisi
* WHILE **mengevaluasi kondisi** sebelum mengeksekusi statement

•Oleh karena itu, perintah DO-WHILE akan mengeksekusi block statement minimal 1 kali, meskipun kondisi **tidak terpenuhi**

# Perulangan DO-WHILE

• Sintaks DO-WHILE

Inisialisasi; do{

|  |
| --- |
| Eksekusi statement minimal 1 kali. Selama kondisi bernilai TRUE, maka perulangan akan terus dijalankan |

statement1; statement2; … update;

} while (kondisi);

# Flowchart DO-WHILE

inisialisasi

update

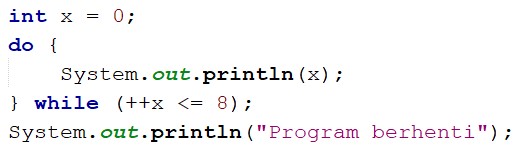
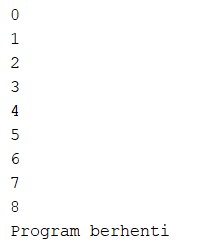
statements

kondisi

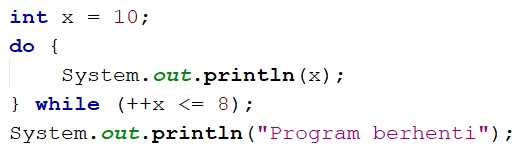
**true**

**false**

# Contoh Perulangan DO-WHILE

Kode Program Output 

Kode Program

 Output

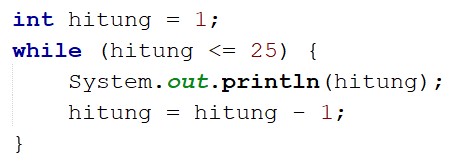
|  |
| --- |
| **Infinite Loop** |

# Infinite Loop

* Saat melakukan eksekusi statement di dalam perulangan, harus terdapat kondisi yang menjadikan kondisi bernilai FALSE
* Jika tidak ada (kondisi terus menerus bernilai TRUE), maka hal ini disebut **infinite loop**, yaitu perulangan yang akan dijalankan terus menerus tanpa batas sampai pengguna menghentikan program
* Logika program harus selalu diperiksa ulang untuk memastikan bahwa loop akan berakhir

# Contoh Infinite Loop

Kode Program



Output



Loop akan berjalan terus menerus sampai pengguna menghentikannya

|  |
| --- |
| **Cara Menghentikan Perulangan** |

# Cara Menghentikan Perulangan

Beberapa cara untuk menghentikan pengulangan untuk program interaktif, di antaranya dapat dilakukan dengan:

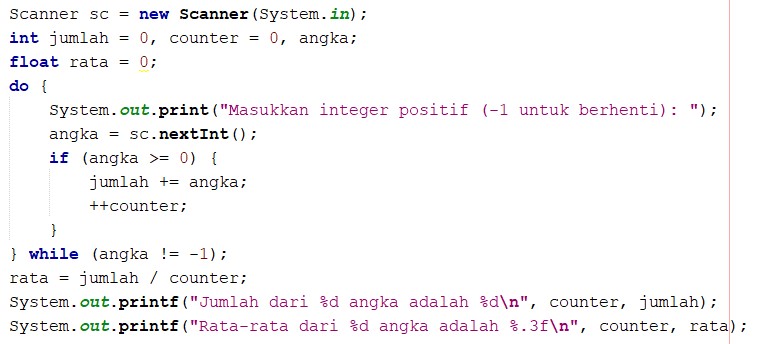
* Menambahkan **Sentinel** atau pembatas dengan kode khusus
* Menambahkan **Pertanyaan** sebagai penentu dilanjutkan atau tidaknya perulangan

# Menambahkan Sentinel

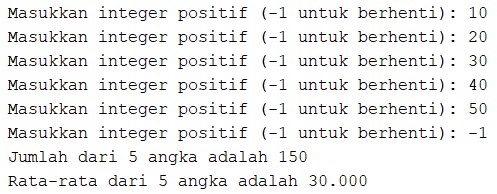
* **Sentinel** adalah nilai yang menandakan akhir dari input pengguna
* **Sentinel loop** menyatakan perulangan yang akan terus berjalan sampai nilai sentinel ditemukan

# Menambahkan Sentinel – Contoh Kode Program

Tuliskan kode program untuk menerima input (integer positif) dari pengguna sampai pengguna memasukkan -1 untuk berhenti. Cetak jumlah dan rata-rata dari angka-angka yang telah dimasukkan.



Output



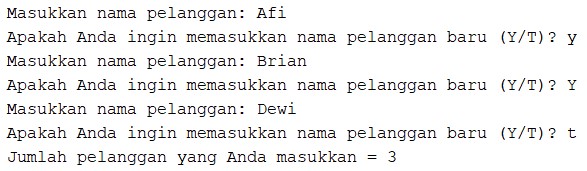
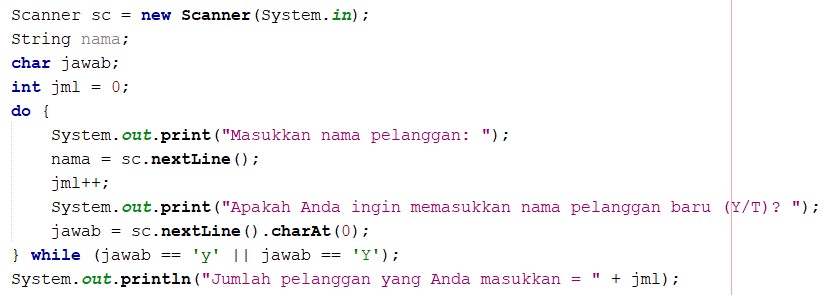
# Menambahkan Pertanyaan

* **Pertanyaan** digunakan untuk memberikan pilihan kepada pengguna apakah pengguna masih akan melanjutkan perulangan
* Apabila kondisi pada perulangan bernilai TRUE berdasarkan jawaban pertanyaan dari pengguna, maka perulangan dilanjutkan
* Contoh:
* Apakah Anda akan melanjutkan perulangan?
* Apakah Anda akan menambahkan barang baru?

# Menambahkan Pertanyaan – Contoh Kode Program

Tuliskan kode program untuk menerima input sejumlah nama pelanggan. Cetak jumlah pelanggan yang telah dimasukkan.

Output



|  |
| --- |
| **Statement BREAK dan CONTINUE** |

# Statement BREAK

* Terkadang suatu program perlu untuk keluar dari perulangan
* Pernyataan BREAK akan **menghentikan paksa** perulangan, kemudian kode di luar perulangan akan dieksekusi
* Selain digunakan untuk keluar dari SWITCH (pemilihan SWITCHCASE), BREAK juga digunakan untuk keluar dari perulangan (FOR, WHILE dan DO-WHILE)

# Contoh Penggunaan BREAK

i = 0

i++

Program berhenti

i<10

true

false

Start

Stop

i==5

i

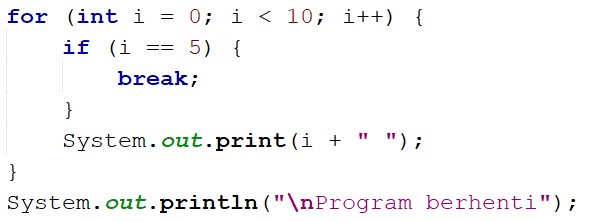
true

false

int

i

break



**Keluar dari**

**loop**

Kode Program

Output

# Statement CONTINUE

* Digunakan untuk menghentikan perulangan yang saat ini terjadi (1 iterasi saja), kemudian melanjutkan perulangan iterasi berikutnya
* Melewati (skip) sisa statement dalam loop, dan eksekusi loop berjalan ke tahap selanjutnya

# Contoh Penggunaan CONTINUE

i = 0

i++

Program berhenti

i<10

true

false

Start

Stop

i==5

i

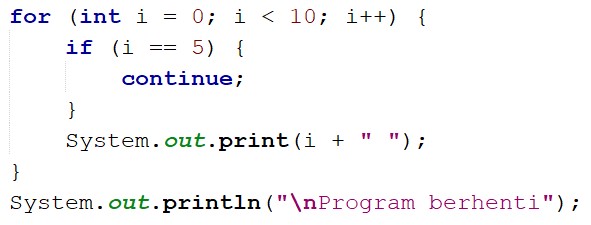
true

false

int

i

continue



**Skip**

**perulangan**

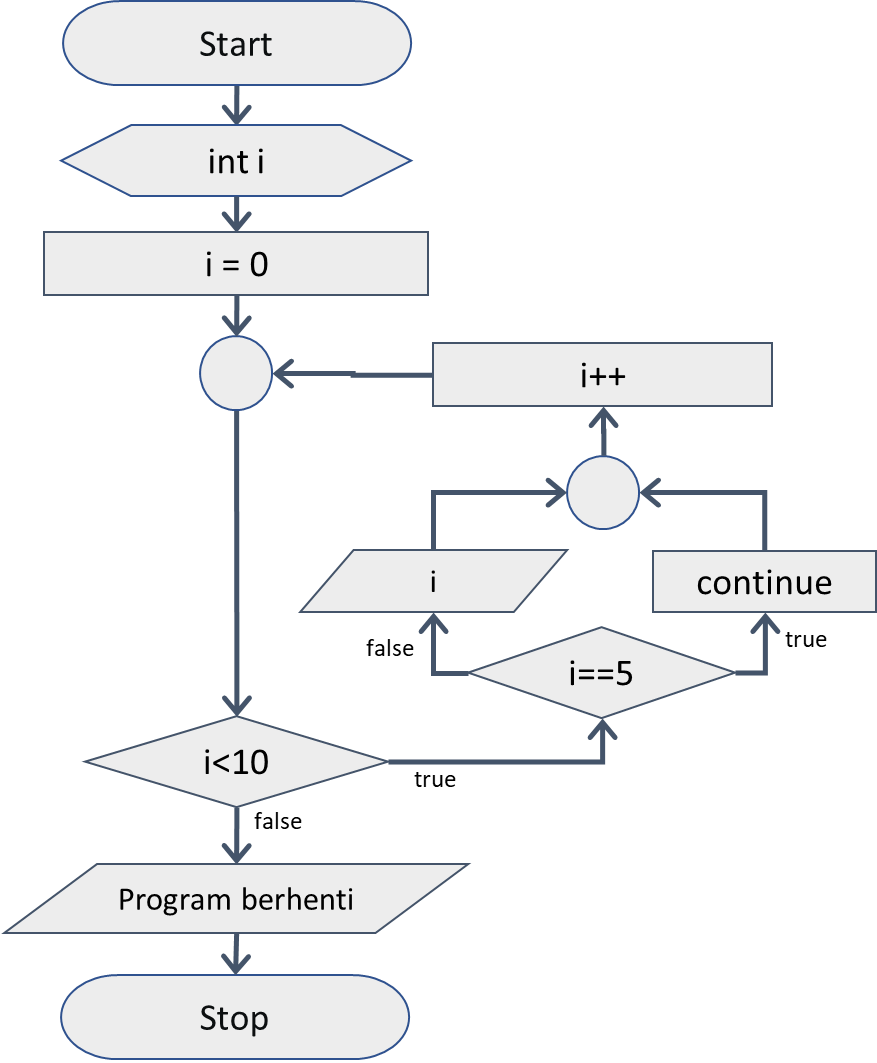
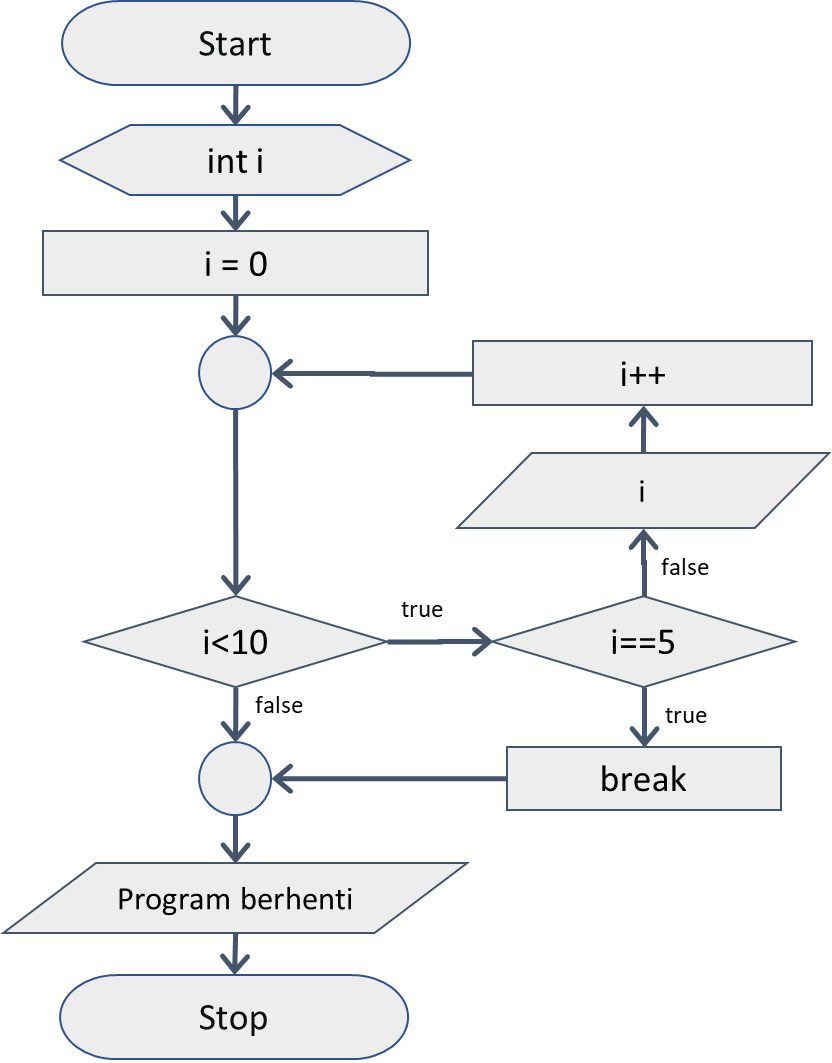
**sesuai**

**kondisi**

Kode Program

Output

# Perbedaan BREAK dan CONTINUE pada Flowchart



Setelah

**break**

tidak ada

update

nilai (

increment

/

decrement

)

Setelah

**continue**

masih ada

update

nilai (

increment

/

decrement

)

# Latihan Mandiri

Buatlah flowchart dari pernyataan berikut dengan menggunakan FOR, WHILE, atau DO-WHILE:

1. Pengguna memasukkan nama dan jenis kelamin dari 30 mahasiswa di suatu kelas. Nama-nama mahasiswa yang ditampilkan hanya yang berjenis kelamin perempuan
2. Menampilkan hasil penjumlahan deret bilangan 25 sampai dengan 1
3. Menampilkan deret bilangan 1 sampai 50, kecuali bilangan kelipatan 3 (1 2 4 5 7 8 10 … 47 49 50)

# Tugas Kelompok

1. Identifikasi sesuai project masing-masing fitur apa saja yang membutuhkan konsep perulangan
2. Tentukan bentuk perulangan yang sesuai (FOR, DO-WHILE, WHILE-DO)
3. Gunakan sentinel/BREAK/CONTINUE jika diperlukan
4. Buatlah algoritma dalam bentuk flowchart