MODUL PERULANGAN

Modul Perulangan Bagian I

Disusun Oleh :

Johannes Alexander Putra

Perulangan

Kalian akan belajar memahami konsep dan cara kerja perulangan dalam pemrograman. Setelah menyelesaikan aktivitas ini, kalian diharapkan mampu menerapkan perulangan dan menghubungkannya dengan kejadian sehari- hari.

a. Pernahkah kalian melakukan aktivitas yang sama berulang kali? Eits... tidak perlu jauh-jauh, untuk melangkah saja, kita mengulang pergerakan kaki kanan dan kiri. Apakah kalian selalu berpikir sebelum melakukan setiap langkah? Pertanyaan terakhir tersebut dapat ditanyakan pada semua kegiatan perulangan yang kalian lakukan.
b. Membaca buku - kalian membalikkan halaman buku setiap selesai membaca halaman tersebut

Salah satu keunggulan program komputer daripada manusia ialah kemampuannya untuk mengolah data yang berukuran besar atau melaksanakan suatu aksi berulang kali dalam periode waktu yang lama tanpa merasa bosan atau lelah. Hal ini dimungkinkan dengan adanya suatu kontrol perulangan. Pernyataan perulangan atau loop merupakan struktur program untuk keperluan iterasi, yaitu memproses satu atau beberapa pernyataan secara berulang (looping) berdasarkan kondisi tertentu. Program

Perulangan (Looping) merupakan suatu bagian yang bertugas untuk melakukan kegiatan mengulang suatu proses sesuai dengan yang diinginkan. Banyak dari aplikasi perangkat lunak yang melakukan pekerjaan berulang-ulang sampai sebuah kondisi yang diinginkan, oleh karena itu pengulangan merupakan bagian penting dalam pemrograman (Sukamto, 2018)

Jenis Perulangan

Terdapat beberapa jenis perulangan:

- Perulangan For

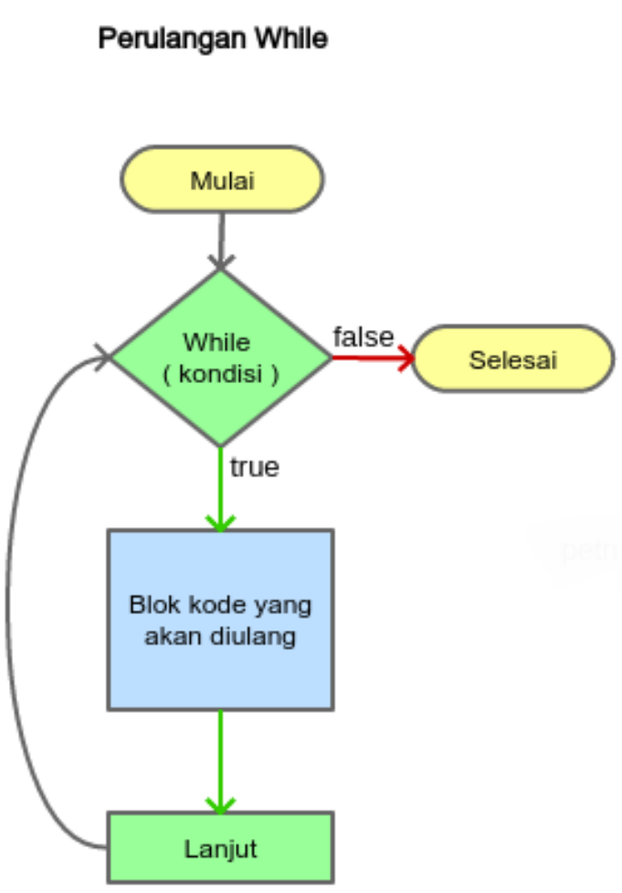
Perulangan for ini biasanya digunakan untuk perulangan yang sudah jelas perlu dilakukan berapa kali.

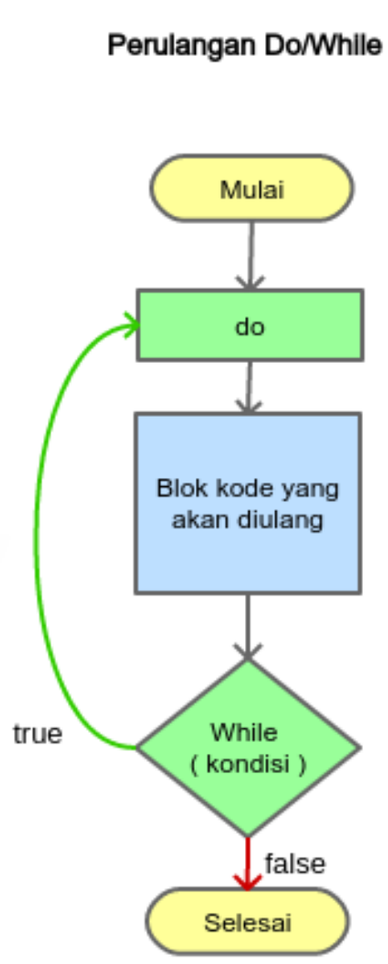
- Perulangan While

Perulangan While biasanya digunakan jika jumlah perulangan tidak diketahui atau memiliki kemungkinan untuk dapat dilakukan kurang dari batas perulangan

- Perulangan Do While

Perulangan Do While biasanya digunakan jika jumlah perulangan tidak diketahui, namun berbeda dengan while karena kondisi perulangan ada di bawah bagian bawah blok perulangan. Perulangan ini minimal dilakukan satu kali karena kondisi perulangan ada di bagian bawah





Perulangan For

Perulangan menggunakan for biasanya digunakan untuk perulangan yang sudah jelas dilakukan berapa kali. Dengan kata lain jumlah perulangan sudah jelas dan diketahui oleh pembuat program dari awal.

Perulangan for dibagi menjadi dua
1. For untuk hitung naik
2. For untuk hitung turun

Perulangan For untuk Perhitungan Naik

```cpp
#include <iostream>

using namespace std;

int main() {
    for(int i = 0;i<=4;i++){
        cout<<i<<endl;
    }

}
```

jdoodle.com/ia/NeC

Keterangan:

- int i = 0 (merupakan bagian inisialisasi)
- cout<<i<<endl (merupakan bagian proses)
- i++ (merupakan Iterasi)
- i<=4 (merupakan terminasi)

Inisialisasi : Tahapan persiapan membuat kondisi awal sebelum melakukan perulangan. Misalnya mengisi variable dengan nilai awal

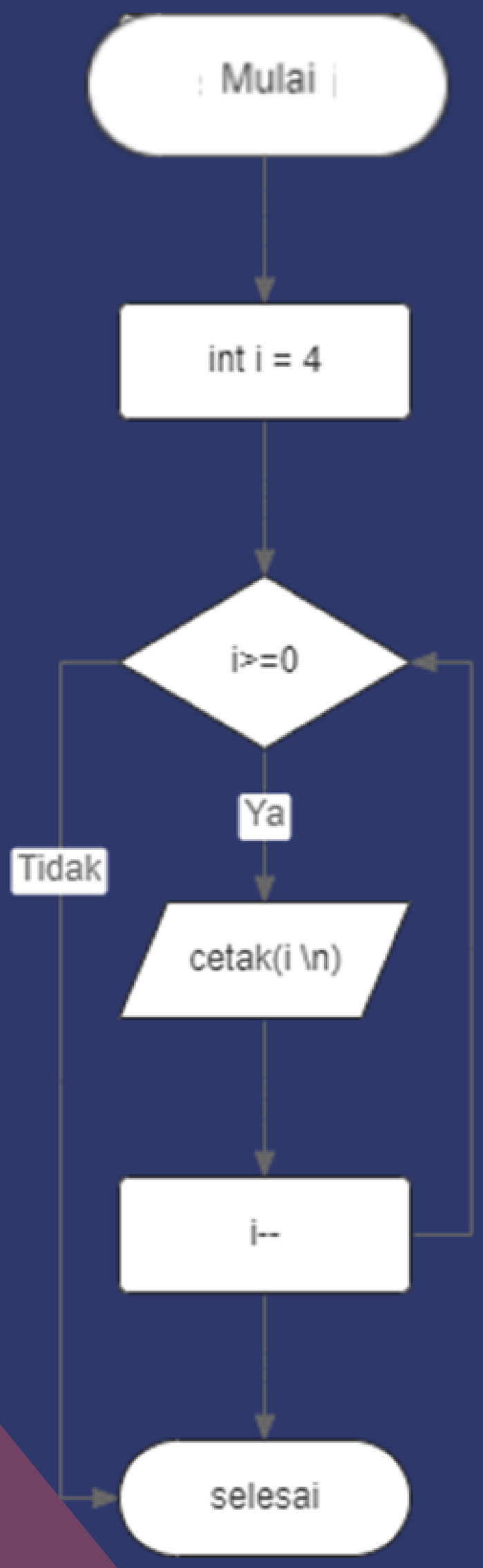
Proses: Tahapan proses terjadi di dalam bagian perulangan di mana berisi semua proses yang perlu dilakukan secara berulang

Iterasi: Kondisi penambahan agar perulangan dapat terus berjalan

Terminasi: Kondisi berhenti dari perulangan. Kondisi berhenti ini penting agar tidak terjadi perulangan yang tanpa henti. (Sukamto, 2018)

Maka output yang akan dicetak adalah
0
1
2
3
4

Pengulangan For untuk perhitungan turun



```cpp
#include <iostream>

using namespace std;

int main() {
    for(int i = 4;i>=0;i--){
        cout<<i<<endl;
    }
}
```

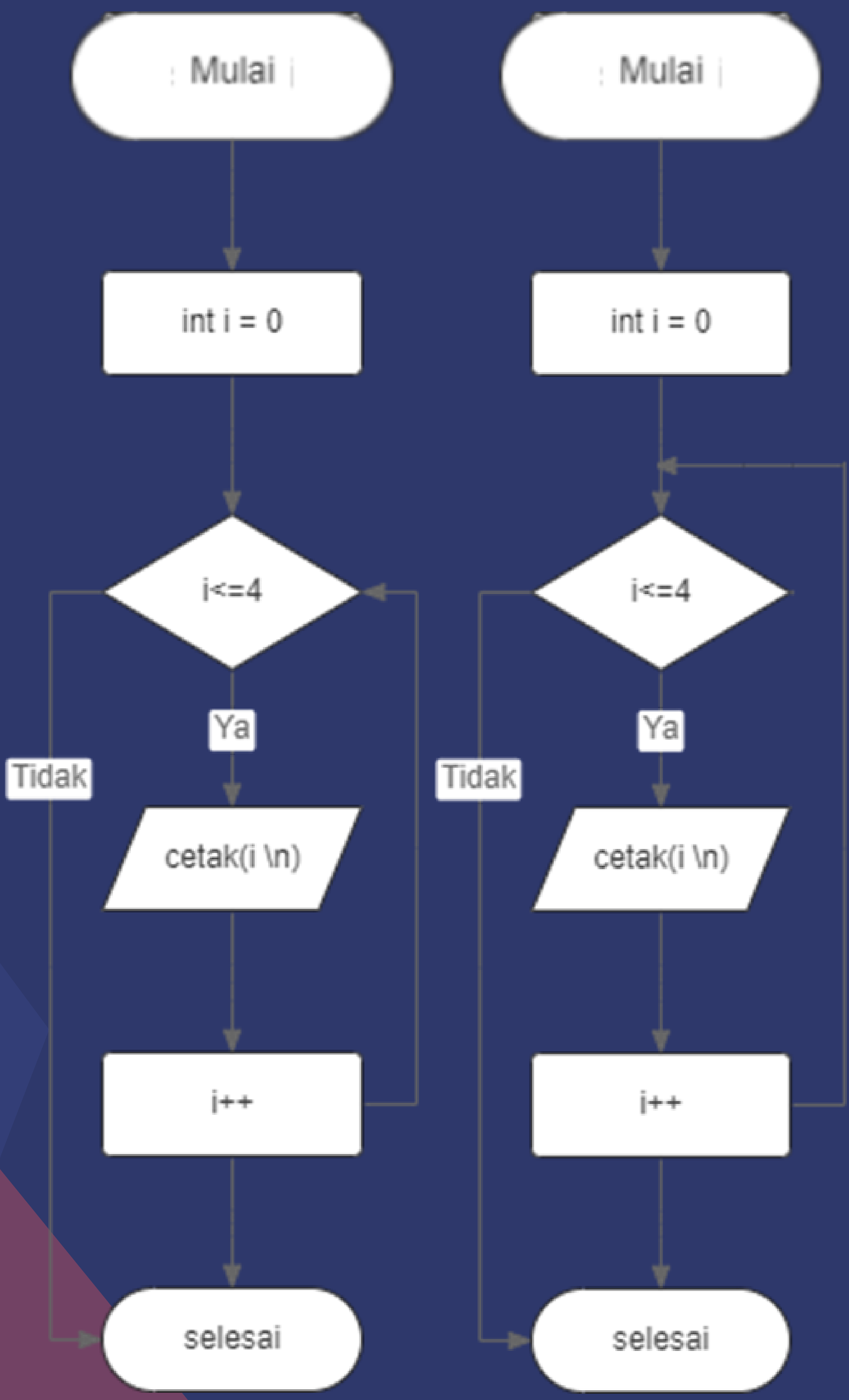
i = 4 (merupakan inisialisasi/nilai awal)
i>=0 (merupakan terminasi/batas atas)
i-- (merupakan iterasi)
cout<<i<<endl (merupakan proses)

jdoodle.com/ia/NeD

Output:
4
3
2
1
0

Contoh Soal dan Penyelesaian
Miftah ingin membuat sebuah aplikasi perulangan dalam bahasa pemrograman C++ untuk menuliskan angka-angka secara berurutan. Diketahui angka yang akan dicetak adalah bilangan bulat dari 0 sampai 4 dengan new line. Diketahui bahwa batas awal dan akhir sudah diinisialisasi pada variable i. Perulangan apakah yang paling cocok dan buatkanlah programnya dengan algoritma yang paling sesuai dalam bahasa C++?
Ada 4 Tahapan Penyelesaian masalah menurut (Polya, 2014) Memahami Masalah, Merencanakan Penyelesaian Masalah, Melaksanakan Rencana Penyelesaian Masalah, dan Memeriksa Kembali
Tahap 1 (Memahami Masalah):
1. Apakah informasi yang diberikan sudah cukup untuk menjawab pertanyaan?
Jawab:
Data yang diberikan sudah dapat menjawab pertanyaan. Dalam soal sudah diberikan batas awal dan batas akhir dari perulangan.
2. Apakah saja data yang diketahui?
Jawab:
- Miftah ingin membuat sebuah aplikasi perulangan yang menuliskan angka-angka secara berurutan
- Miftah mengetahui bahwa angka yang dituliskan adalah dari angka 0 sampai angka 4 tanpa new line.
- Batas awal dan akhir sudah diinisialisasi dengan variable i
- Bahasa Pemmrograman yang diguakan adalah C++

Tahap 2 (Membuat Rencana Penyelesaian Masalah):
1. Kira-kira kondisi apakah yang mirip dengan kasus tersebut?
Jawab: For, Dikarenakan tadi telah disebutkan bahwa terdapat batas awal dan batas akhir yang jelas yaitu menggulang dari angka 0 sampai 4
2. Kira-kira bagaimana bentuk flowchartnya
Jawab: Anda dapat menggunakan salah satu dari kedua flowchart ini yang penting adalah arahnya mengarah kepada pengecekan kondisi.



Artinya proses looping dilakukan mulai dari 0 sampai dengan 4 karena batas terminasinya i<=4 (I lebih kecil atau sama dengan 4) yang artinya 4 ikut termasuk yang dicetak.

Tahap 3 (Melaksanakan Rencana Penyelesaian Masalah):
1. Bagaimana implementasi masalah tersebut dalam pemrograman C++?

```cpp
#include <iostream>
using namespace std;
int main(){
    for(int i = 0;i<=4;i++){
        cout<<i<<endl;
    }
    return 0;
}
```

Tahap 4 (Memeriksa Kembali):
1. Apakah kode tersebut sudah benar dan menghasilkan output dengan benar. Misal kodenya begini ?

```cpp
#include <iostream>
using namespace std;
int main(){
    for(int i = 0;i<=4;i++){
        cout<<i<<endl;
    }
    return 0;
}
```

Jawab: Kode tersebut tidak menghasilkan error dan hasilnya pun tepat

```
0
1
2
3
4
```

2. Bagaimana agar kode tersebut menghasilkan output yang berbeda misalnya

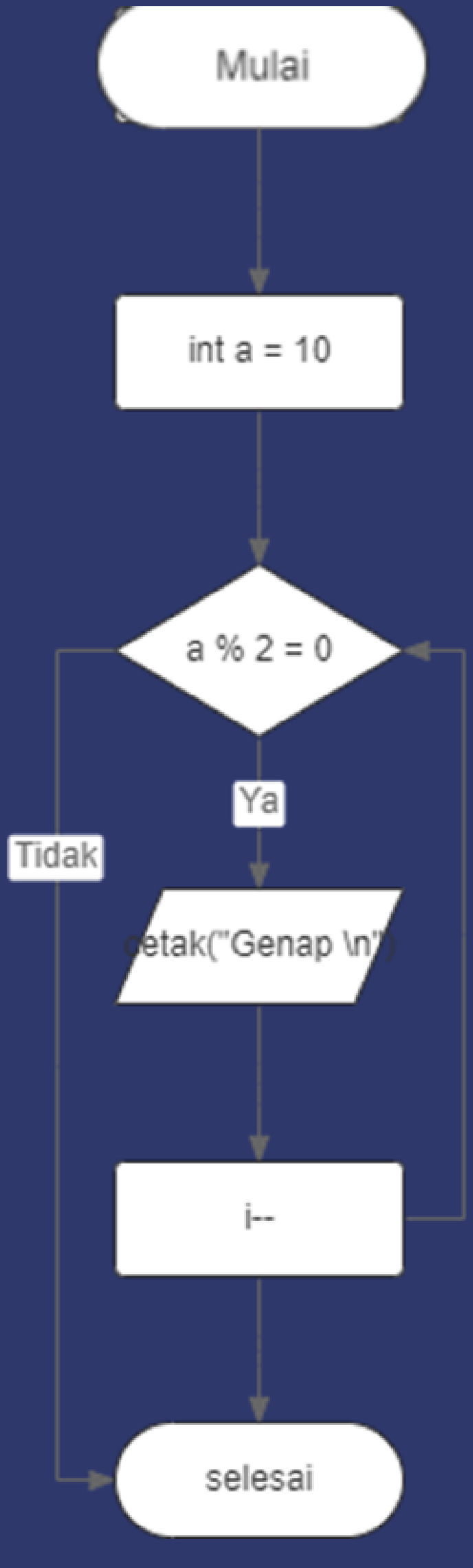
4
3
2
1
0

Jawab: Bisa tinggal ubah for menaik menjadi for menurun

```
#include <iostream>
using namespace std;
int main() {
    for(int i = 4;i>=0;i--){
        cout<<i<<endl;

    }
    return 0;
}
```

While

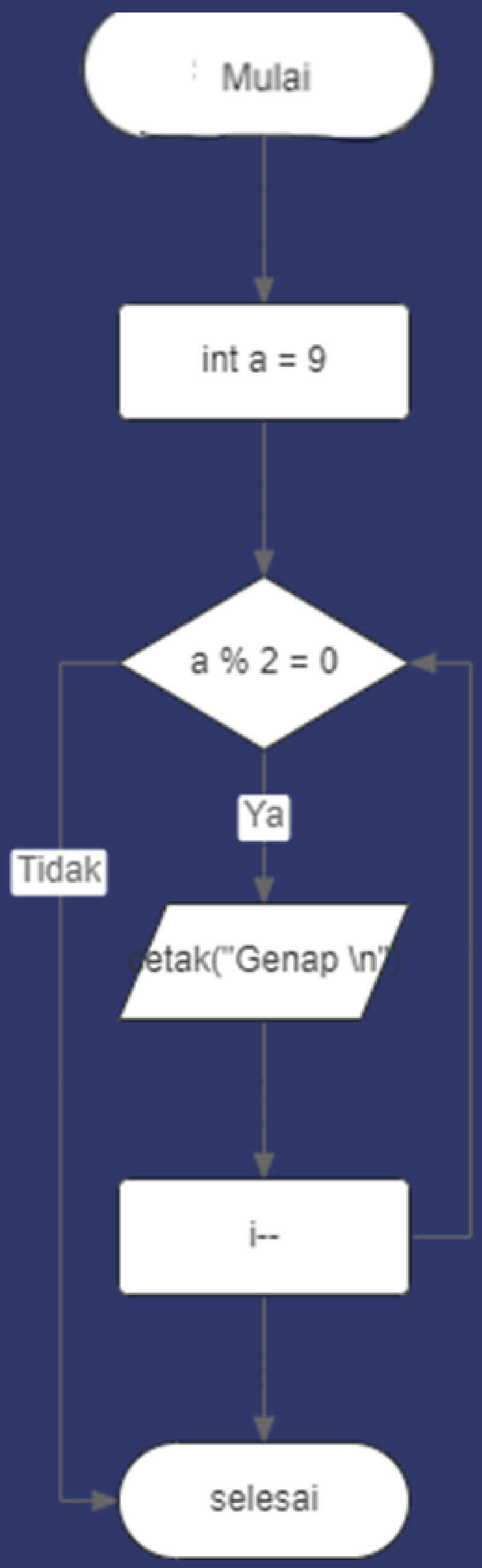
Perulangan while biasa digunakan jika jumlah perulangan tidak diketahui atau memilikin kemungkinan dapat dilakukan kurang dari batas perulangan. Perulangan while ini akan melakukan perulangan selama kondisi perulangan terpenuhi. (Sukamto, 2018)



```cpp
#include <iostream>

using namespace std;

int main() {
    int a = 10;
    while(a % 2 == 0){
        cout<<"genap"<<endl;
        a--;
    }
    return 0;

}
```

jdoodle.com/ia/NeI

Penjelasan: Sebelum memasuki badan perulangan kondisi terlebih dahulu diperiksa apakah memenuhi atau tidak, Aksi dikerjakan berulang kali selama kondisinya bernilai benar. Jika kondisi bernilai salah maka kondisi tidak dikerjakan atau berhenti dikerjakan (Munir & Lidya, 2016) Dalam kasus tersebut outputnya adalah Genap kenapa hanya satu kali diulang, karena pada perulangan kedua kondisi sudah tidak dipenuhi karena a = 9 dan 9 % 2 tidak sama dengan 0

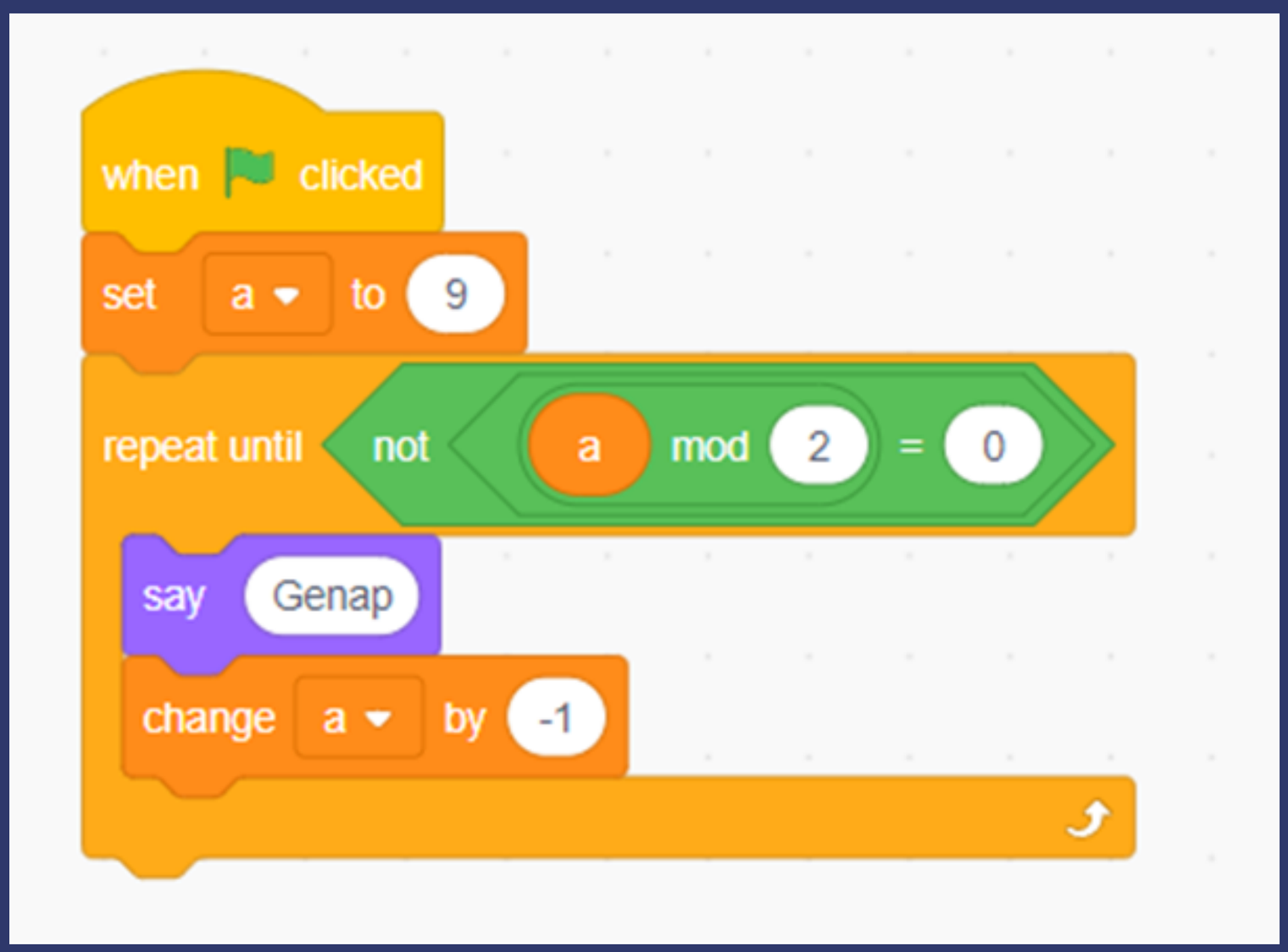

Mulai
int a = 9
a % 2 = 0
Ya
Tidak
cetak("Genap \n")
i--
selesai

```cpp
#include <iostream>

using namespace std;

int main() {
    int a = 9;
    while(a % 2 == 0){
        cout<<"genap"<<endl;
        a--;
    }
    return 0;

}
```

jdoodle.com/ia/NeL

Jika a adalah 9 maka tidak ada aksi yang dikerjakan karena tidak memenuhi pengecekan kondisi a % 2 == 0 karena 9 %2 adalah 1

Jika diubah dalam Scratch



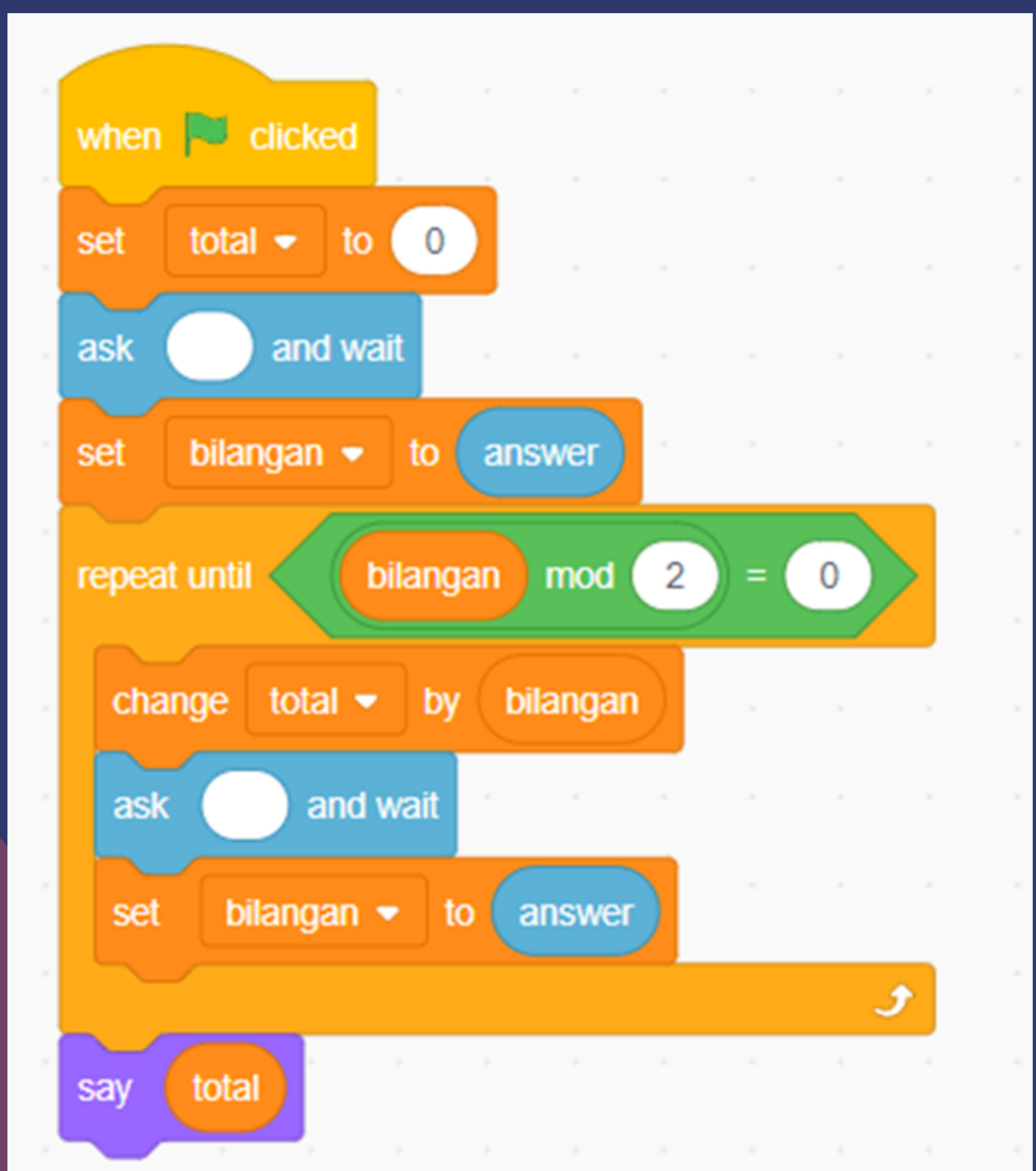
Penjelasan kode tersebut akan melakukan pengecekan di awal repeat until (dikerjakan sampai) dengan a mod 2 ! = 0. Perhatikan ini berbeda dengan while loop kalau while loop perulangan dilakukan jika kondisi terpenuhi. Sedangkan repeat until, perulangan dilakukan sampai. Dalam kasus ini Perulangan dilakukan sampai a mod 2 tidak sama dengan 0. Dalam hal ini perulangan tidak dilakukan karena pengulangan dilakukan sampai a mod 2 != 0 dan memang benar 9 mod 2 != 0.

Contoh Soal dan Penyelesaian
Andi ingin membuat program dalam bahasa C++ & Scratch untuk menampilkan jumlah dari penjumlahan bilangan ganjil yang dimasukan oleh pengguna. Andi mengetahui bahwa program tersebut akan terus meminta inputan bilangan sampai angka yang dimasukan bukan bilangan ganjil. Andi mengetahui pengecekan program dilakukan di awal sebelum masuk ke dalam blok perulangan. Program akan mencetak total bilangan ganjil yang dimasukan tanpa new line. Penamaan variable dibebaskan.
Perulangan apakah yang paling sesuai dan bagaimana bentuk kodenya?
Ada 4 Tahapan Penyelesaian masalah menurut (Polya, 2014) Memahami Masalah, Merencanakan Penyelesaian Masalah, Melaksanakan Rencana Penyelesaian Masalah, dan Memeriksa Kembali
Tahap 1 (Memahami Masalah):
1. Apakah informasi yang diberikan sudah cukup untuk menjawab pertanyaan?
Jawab:
Informasi yang diberikan sudah bisa menjawab pertanyaan, pertama diketahui bahwa program yang akan dibuat ditujukan untuk menghitung penjumlahan dari bilangan ganjil yang dimasukan. Diketahui juga program berhenti ketika angka yang dimasukan bukan ganjil dan akan mencetak totalnya.
2. Apa saja data yang diketahui?
Jawab:
- Data yang diketahui adalah program akan menghitung jumlah bilangan ganjil yang dimasukan dan akan berhenti ketika yang dimasukan bukan ganjil.
- Penamaan variable dibebaskan
- Program akan mencetak totalnya
- Bahasa Pemrograman yang digunakan adalah C++ dan Scratch

Tahap 2 (Merencanakan Pemecahan Masalah)
1. Kondisi apakah yang mirip dengan kasus tersebut
Jawab:
Perulangan While dikarenakan kita tidak mengetahui berapa kali perulangan yang dilakukan dan pengecekan dilakukan di awal sebelum masuk ke dalam blok perulangan.
2. Bagaimana flowchart diagramnya?
Jawab:
Anda dapat menggunakan salah satu dari kedua flowchart ini yang penting adalah arahnya setelah blok program dalam perulangan while dijalankan. Maka proses dilanjutkan dengan melakukan pengecekan kembali jadi arah panahnya harus mengarah kepada tempat mengecek kembali

Tahap 3 (Melaksanakan Pemecahan Masalah)
Jawab:

```cpp
#include <iostream>
using namespace std;
int main(){
    int bilangan;
    int total = 0;
    cin>>bilangan;
    while(bilangan % 2 != 0){
        total = total + bilangan;
        cin>>bilangan;
    }
    cout<<total;
}
```



Tahap 4 (Memeriksa Kembali)

1. Apakah hasil pemecahan masalah sudah benar?

Jawab:

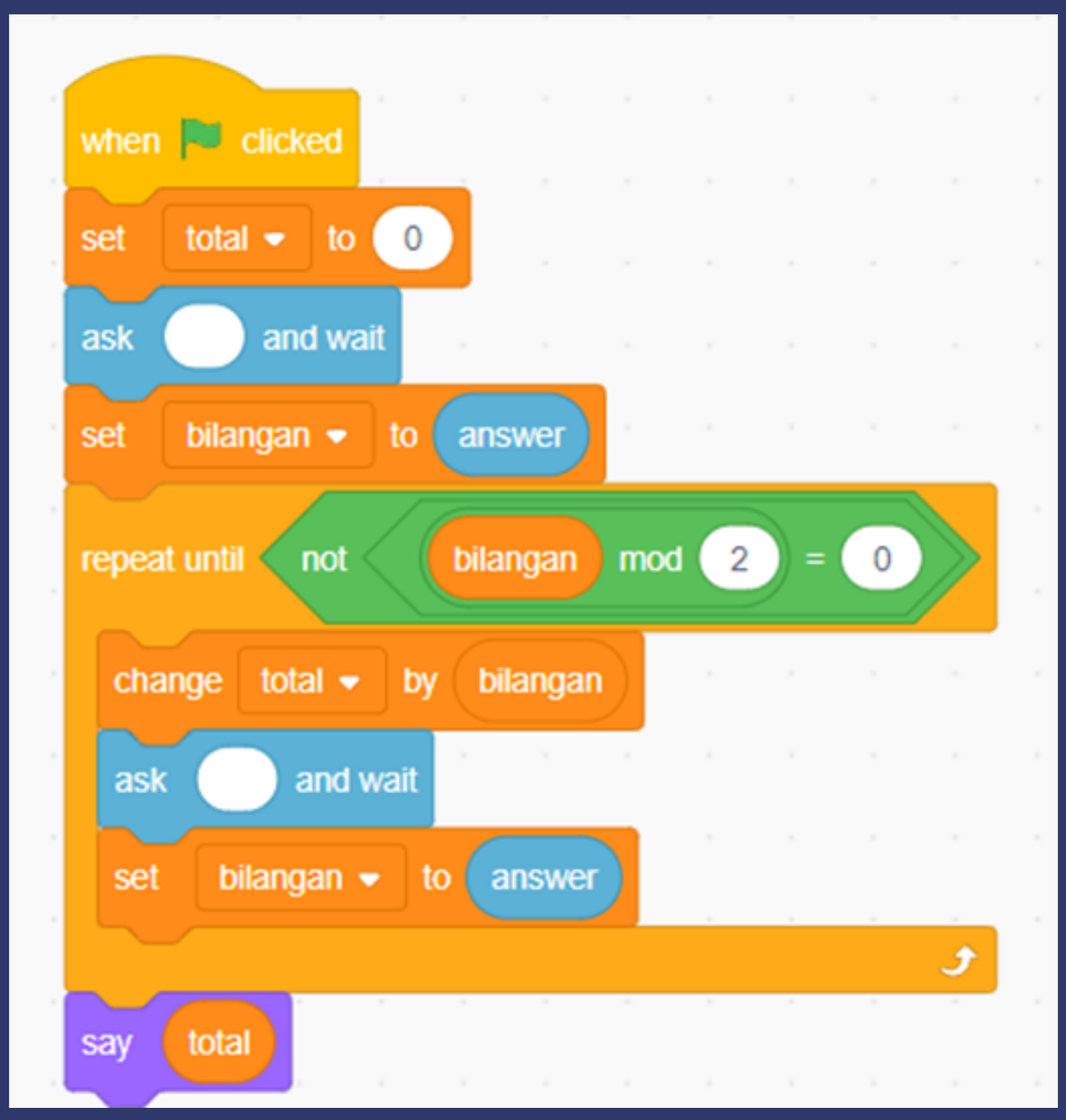
Pemecahan masalah tersebut sudah sesuai dengan yang diminta yaitu menghitung total dari bilangan ganjil dan berhenti ketika yang dimasukan adalah bilangan genap

```
1
3
5
2
9
--------------------------------
Process exited after 2.715 seconds with return value 0
Press any key to continue . . .
```

2. Bagaimana agar kode tersebut menghasilkan total dari bilangan genap dan berhenti ketika yang dimasukan adalah bilangan ganjil?

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

    int bilangan;
    int total = 0;
    cin>>bilangan;
    while(bilangan % 2 ==0){
        total = total + bilangan;
        cin>>bilangan;
    }
    cout<<total;
    return 0;
}
```

Dengan menggubah kondsi pada while yang sebelumnya bilangan % 2 != 0 menjadi bilangan % 2==0



Dengan menggubah kondisi pada repeat until menjadi not bilangan mod 2 = 0 dari sebelumnya bilangan mod 2 = 0

Daftar Pustaka

Munir, R., & Lidya, L. (2016). Algoritma Dan Pemrograman Dalam Bahasa Pascal, C, C++ Edisi Keenam. Bandung: Informatika.

Polya, G. (2014). How to Solve It: A New Aspect of Mathematical Method. New Jersey: Princeton University Press.

Sukamto, R. A. (2018). Logika Algoritma dan Pemograman Dasar. Bandung: Modula.

Daftar Pustaka

Polya, G. (2015). How to Solve It: A New Aspect of Mathematical Method. New Jersey: Princeton University Press.

Sukamto, R. A. (2018). Logika Algoritma dan Pemograman Dasar. Bandung: Modula.