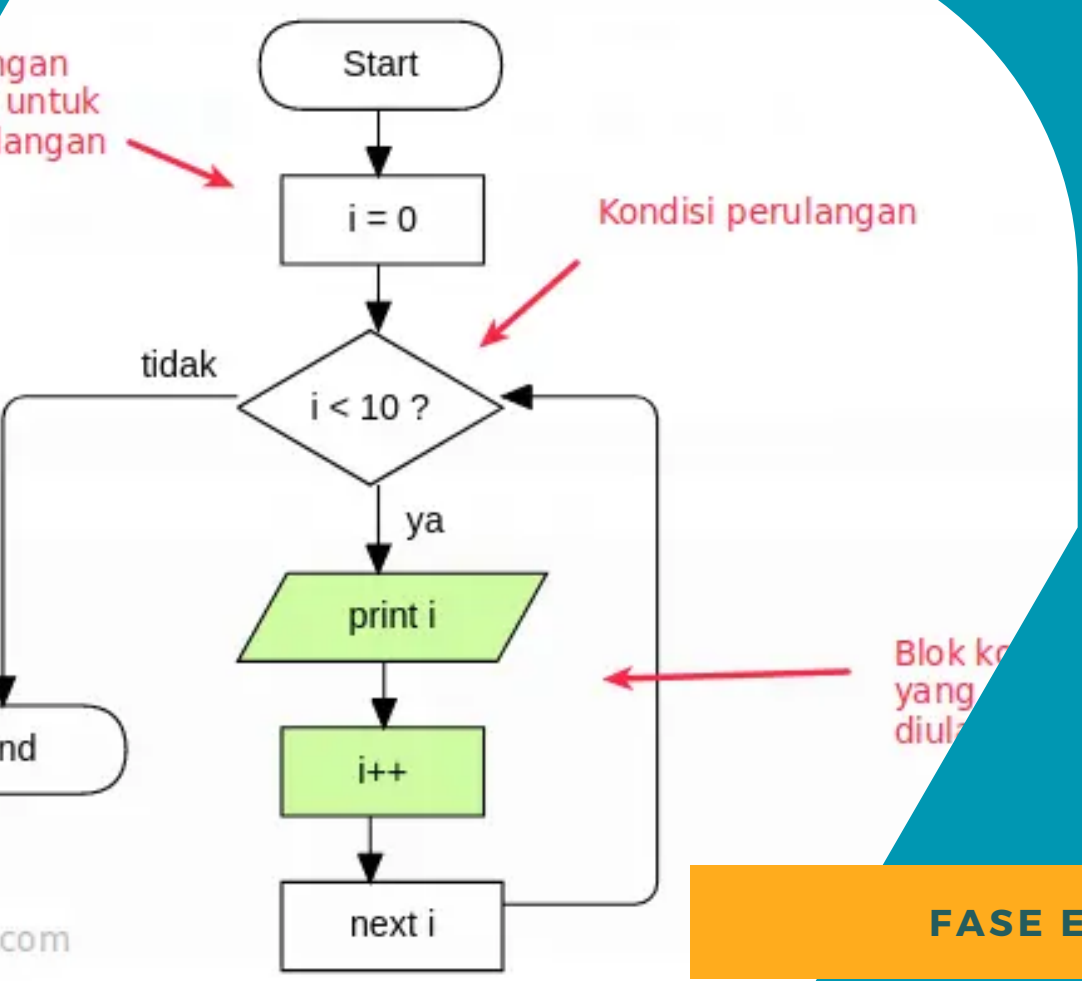


FASE E

PERULANGAN

Informatika

DISAJIKAN OLEH
Johannes Alexander Putra

DAFTAR ISI

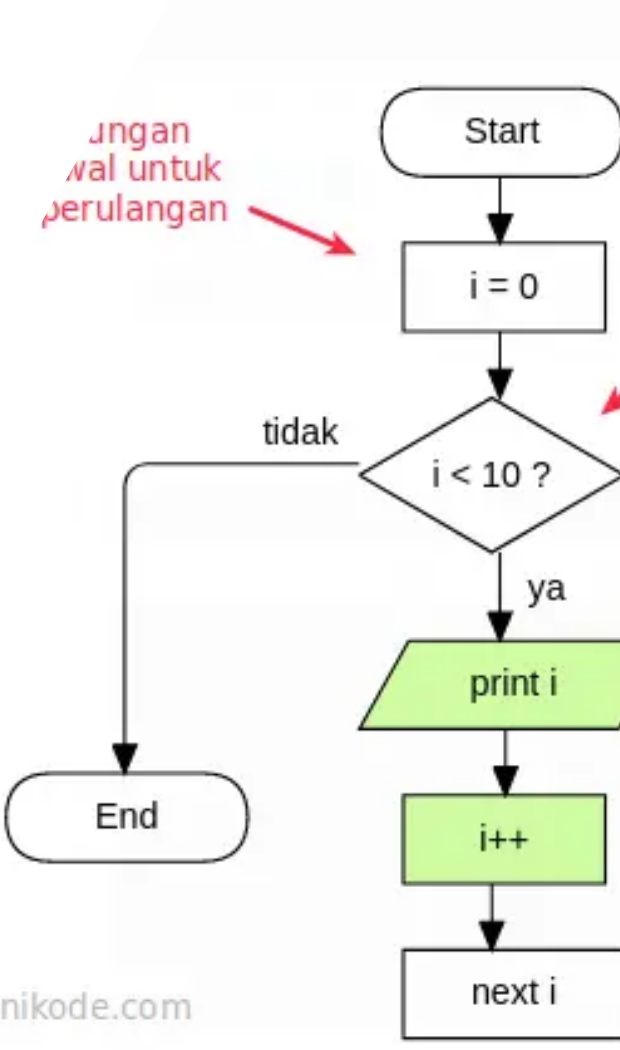
CAPAIAN PEMBELAJARAN

Pada akhir fase E, peserta didik mampu menerapkan praktik baik konsep pemrograman prosedural dalam salah satu bahasa pemrograman prosedural dan mampu mengembangkan program yang terstruktur dalam notasi algoritma atau notasi lain, berdasarkan strategi algoritmik yang tepat.

TUJUAN PEMBELAJARAN

Tujuan Pembelajaran setelah menyelesaikan pertemuan ini adalah siswa dapat:

- Memahami permasalahan yang berkaitan dengan perulangan
- Menentukan pemecahan masalah dengan perulangan
- Melakukan implementasi perulangan untuk suatu permasalahan
- Melakukan evaluasi terhadap penggunaan perulangan



PERULANGAN

Perulangan merupakan suatu bagian yang bertugas melakukan kegiatan menggulang suatu proses sesuai dengan yang diinginkan. Banyak dari aplikasi perangkat lunak yang melakukan pekerjaan yang berulang sampai sebuah kondisi yang diinginkan. Oleh karena itu, perulangan merupakan bagian yang penting dalam pemograman (Sukamto, 2018).

PERULANGAN FOR

Perulangan for merupakan struktur kontrol repetitif yang memungkinkan Anda untuk menjalankan proses dengan jumlah pengulangan tertentu (jumlah pengulangan sudah diketahui sebelumnya) (Jubilee Enterprise, 2017)

```
1. for (initialization; termination; increment) {
2.     statement(s)
3. }
```

Contoh:

```
#include <iostream>
using namespace std;

int main()
{
    for (int i = 1; i <= 10; i++) {
        cout << "Angka: " << i << endl;
    }
    return 0;
}
```

Bedah Code Perulangan For
Kita membuat variabel dengan tipe data integer yang kita sebut sebagai initialization. Dalam hal ini kita menginisialisasi angka dimulai dari 1.
1. int i=1;
Pada tahap ini kita menentukan batasan nilai akhir suatu perulangan atau disebut dengan termination. Contoh di atas adalah angka lebih kecil atau sama dengan 10.
1. i<=10;
Kita menentukan aksi terhadap perulangan, aksi tersebut bisa menaikkan (increment) atau menurunkan (decrement). Apabila kita ingin melakukan aksi increment maka nilai awal (initialization) harus lebih kecil daripada nilai akhir (termination). Begitupun sebaliknya, jika melakukan aksi menurun maka nilai awal (initialization) harus lebih besar daripada nilai akhir (termination). Seperti contoh di atas kita melakukan aksi increment.
1. i++;

PERULANGAN WHILE

Pengulangan while adalah struktur kontrol yang memungkinkan Anda mengulangi suatu proses dengan jumlah pengulangan tertentu.

Selama kondisi bernilai benar maka statemen dalam tubuh while akan terus dieksekusi

```
#include <iostream>
using namespace std;

int main()
{
    int value = 1;
    while(value <=10){
        cout<<"Angka : "<<value<<endl;
        value++;
    }
    return 0;
}

```

Bedah Code Perulangan While
Kita mendeklarasikan variabel nilai dengan angka 1.
1. int value = 1;
Pengecekan suatu kondisi pada variabel. Apabila nilai lebih kecil atau sama dengan 10.
1. while (value <= 10)
Melakukan sebuah perintah apabila kondisi terpenuhi.
1. cout<< "Angka : <<value << endl
<< digunakan untuk menghubungkan sesuatu
<<endl untuk berpindah baris
2. value++;

PERULANGAN DO WHILE

Perulangan yang mempunyai fungsi yang sama dengan While, tetapi pengecekan kondisinya dilakukan di akhir. Pada perulangan ini minimal melaksanakan perintah sekali, kemudian mengecek kondisi.

```
#include <iostream>
using namespace std;

int main()
{
    int value = 1;
    do{
        cout<<"Angka : " <<value<<endl;
        value++;

    }while(value <=10);
    return 0;
}
```

Angka : 1
Angka : 2
Angka : 3
Angka : 4
Angka : 5
Angka : 6
Angka : 7
Angka : 8
Angka : 9
Angka : 10

Bedah Code Perulangan Do-While

Bagaimana bisa menghasilkan output seperti di atas? Mari kita bahas bersama-sama:

1. Kita mendeklarasikan variabel nilai dengan angka 1.

1. int value = 1;

2. Melakukan sebuah perintah terlebih dahulu.

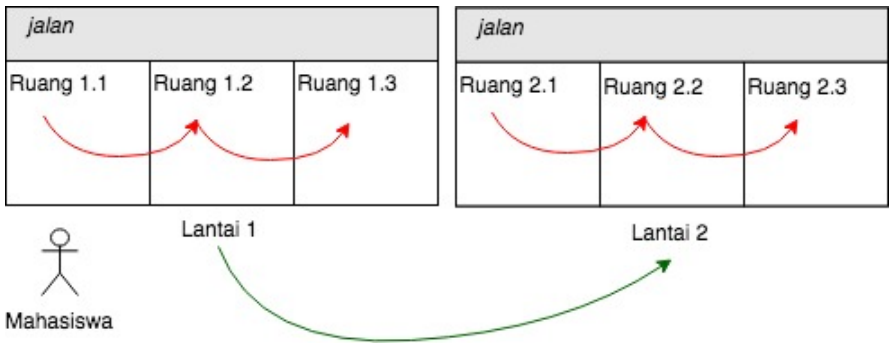
1. System.out.println("Angka : " + value);
2. value++;

3. Pengecekan suatu kondisi terhadap variabel. Apabila terpenuhi maka perintah dilanjutkan.

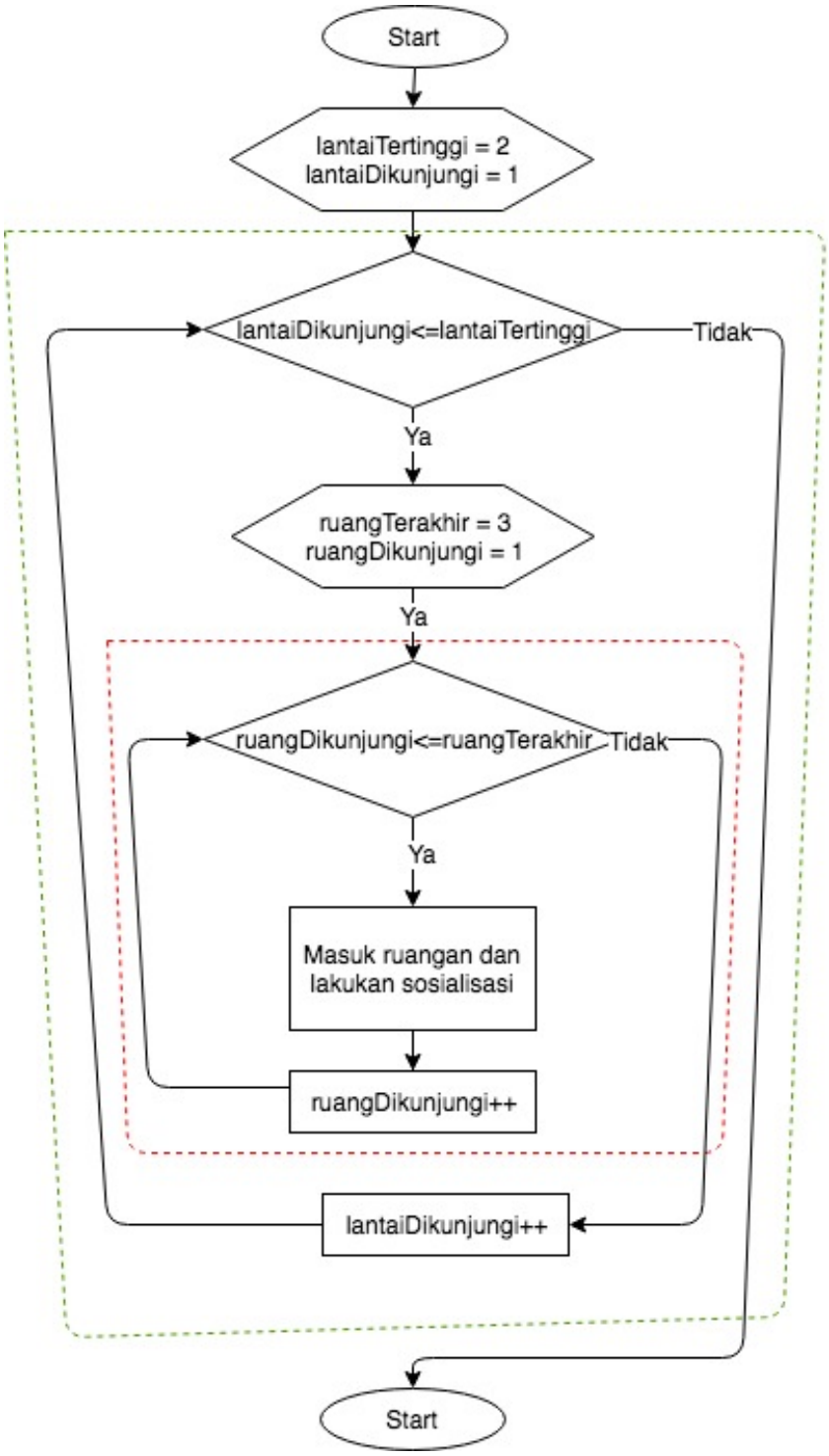
1. while (value <= 10)

PERULANGAN BERSARANG

Perulangan bersarang (nested loop) adalah struktur perulangan yang berada di dalam perulangan lainnya. Pada umumnya, struktur perulangan yang berada di dalam perulangan lainnya tersebut memiliki hubungan yang saling terkait dalam menyesuaikan sebuah kasus. Pada dasarnya tidak ada batasan dalam jumlah perulangan bersarang. Tetapi sebaiknya tidak terlalu dalam, untuk menghindari kompleksitas yang tinggi serta alur program menjadi lebih sukar untuk dipahami.



Jika digambarkan dalam flowchart maka ilustrasi di atas akan tampak seperti Gambar di bawah ini . Area yang bergaris merah disebut dengan inner loop, sedangkan area yang bergaris hijau adalah outer loop.


Start
lantaiTertinggi = 2
lantaiDikunjungi = 1
lantaiDikunjungi<=lantaiTertinggi
Tidak
Ya
ruangTerakhir = 3
ruangDikunjungi = 1
Ya
ruangDikunjungi<=ruangTerakhir
Tidak
Ya
Masuk ruangan dan
lakukan sosialisasi
ruangDikunjungi++
lantaiDikunjungi++
Start

Dan apa dituliskan dalam pseudocode yang sangat sederhana, maka bentuk dasar dari perulangan bersarang bisa dituliskan sebagai berikut:

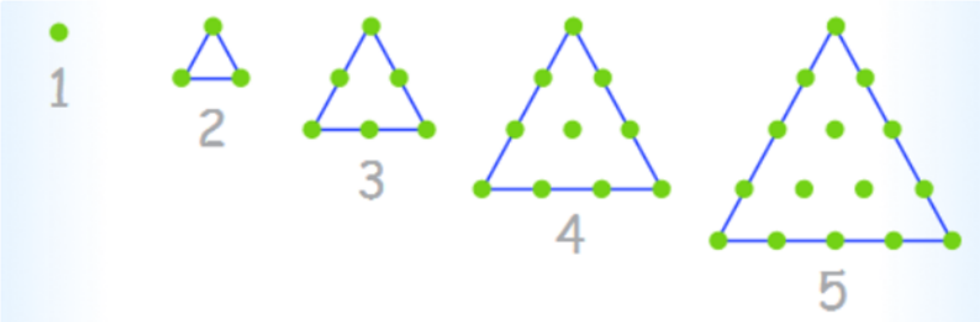
```
loop-1 {
    loop-2 {
    // another statement
    loop-n {
    // statement
    }
  }
}
```

Semua sintaks perulangan yang telah dibahas sebelumnya, seperti for, while dan do-while, semuanya bisa digunakan untuk kasus perulangan bersarang. Dan tidak ada aturan yang mengharuskan menggunakan sintaks yang sama antara perulangan luar dan perulangan yang ada di dalamnya. Misalkan, perulangan luar menggunakan for, dan perulangan yang dalam menggunakan while, hal tersebut bisa dilakukan.

```
for(int i=0; i<2; i++)  {
  for(int j=1; j<=5; j++) {
    //statement
  }
}
```

CONTOH SOAL

Para Capybara sedang belajar tentang Triangular Number Sequence dengan menggunakan kelereng. Guru mereka meminta untuk menyusun kelereng-kelereng mereka membentuk segitiga-segitiga yang merepresentasikan anggota deret dari 1 sampai N. Bentuk dari Triangular Number Sequence adalah sebagai berikut.



Untuk mendapatkan angka ke-N pada deret, jumlah titik pada segitiga sebelumnya ditambahkan dengan sebaris titik baru berjumlah N. Misalnya, untuk mendapatkan angka ke-4 pada deret, jumlah titik pada segitiga sebelumnya (6) ditambahkan dengan N (4). Jadi, angka ke-4 pada deret adalah 10. Para capybara sangat tidak bisa berhitung. Mereka tidak tahu berapa jumlah kelereng yang diperlukan untuk membentuk setiap segitiga. Bantulah mereka untuk menghitung jumlah kelereng yang dibutuhkan oleh masingmasing segitiga, dari segitiga pertama sampai segitiga ke-N!

Spesifikasi Input

Input berupa sebuah bilangan bulat N yang merupakan banyaknya segitiga yang harus dibentuk.

Spesifikasi Output

Output berupa N buah bilangan bulat (dipisahkan oleh spasi) yang merupakan jumlah kelereng untuk membentuk segitiga pertama sampai segitiga ke-N.
Contoh

Input: 6
Output: 1 3 6 10 15 21

Step 1. Memahami Masalah

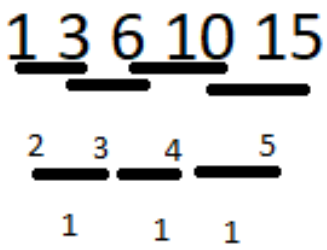
Memahami masalah: Menentukan apa yang diketahui dan ditanyakan kemudian memberikan keterangan apa yang sudah diketahui cukup untuk menjawab pertanyaan.

Soal tersebut adalah soal yang berbentuk pola jika kita memasukan angka 6 akan menghasilkan pola 1 3 6 10 15 21

Step 2- Mengidentifikasi Masalah / Merencanakan Penyelesaian Masalah

Merencanakan penyelesaian masalah : Mengidentifikasi masalah dan mencari jalan yang tepat untuk menyelesaikannya

Masalah pada soal tersebut merupakan masalah pola bilangan dan tepat menggunakan perulangan for karena batas awal dan batas akhir diketahui



Step 3- Melaksanakan Pemecahan Masalah

Penyelesaian dalam bahasa C++

```cpp
#include <iostream>

using namespace std;

int main()
{
    int i,angka;
    angka = 0;
    int masukan;
    cin>>masukan;
    for(i = 1;i<=masukan;i++)
{       angka = angka + i;
        cout<<angka;
        cout<<" ";
    }
}
```

Step 4-Memeriksa Kembali

Memeriksa kembali hasil, langkah ini dilakukan dengan memeriksa kebenaran jawaban

Pada bagian ini kita harus memerika kembali

1. Pada program tersebut memasukan satu angka
2. Hasil pada keluaran sudah tepat

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.
Wahyono, dkk. (2021). Buku Panduan Guru Informatika SMA Kelas 10. Jakarta: Pusat Kurikulum dan Perbukuan

Ada pertanyaan? Hubungi kami.

www.itsjohanes.my.id
johannesap@upi.edu
0819-3417-2542