[1] Pengenalan Algoritma dan Struktur Data

Apa itu Algoritma dan Struktur Data?

- ✧ Algoritma adalah urutan atau langkah-langkah untuk penghitungan atau untuk menyelesaikan suatu masalah yang ditulis secara berurutan.
- ✧ Struktur data adalah cara menyimpan atau merepresentasikan data didalam komputer agar bisa dipakai secara efisien.

Konsep Class dalam Java

Saat kita membuat program dengan Java, kita diwajibkan menggunakan class.

- ✧ Class merepresentasikan konsep atau sesuatu dari di dunia nyata

Fungsi main() adalah fungsi utama dalam program Java. Semua kode yang kita tulis di dalamnya, akan langsung dieksekusi.

```
package com.asd.praktikum1;

class Hello {
    public static void main(String[] args){
        System.out.println("Hello World!");
    }
}
```

Package sebagai Kumpulan Class Sejenis

Ketika membuat program yang lebih kompleks, misalnya membuat aplikasi Android, kalian akan sering menggunakan package.

Untuk sekarang, cukup pahami package sebagai folder yang berisi sekumpulan program Java yang saling berkaitan.

Prosedur, Fungsi, dan Method

“Bagaimana kalau kita membuat program yang cukup besar, apakah kita masih bisa menulis semua kodenya di dalam fungsi main()?”

Bisa-bisa saja, tapi kurang efektif dan akan menghabiskan banyak tenaga untuk mengetik kodenya. Belum lagi kalau ada error… Solusinya menggunakan prosedur/fungsi.

- ✧ Prosedur/fungsi dapat memecah program menjadi sub-sub program, sehingga kita bisa membuat program lebih efisien, juga mengurangi pengetikan kode yang berulang-ulang.

Jangan bingung…karena ketiga-tiganya sama.

- ✧ **Prosedur** adalah sebutan untuk fungsi yang tidak mengembalikan nilai. Fungsi ini biasanya ditandai dengan kata kunci void.
- ✧ **Fungsi** adalah sebutan untuk fungsi yang mengembalikan nilai.
- ✧ **Method** adalah fungsi yang berada di dalam Class. Sebutan ini, biasanya digunakan pada OOP.

Untuk memudahkan, mari kita sebut semuanya **fungsi**.

Cara Membuat Fungsi	Fungsi harus dibuat atau ditulis di dalam class. Struktur dasarnya seperti ini:
	
	`static TipeDataKembalian namaFungsi(){`
	`  // statemen atau kode fungsi`
	`}`
	
	Penjelasan:
	
	✧ Kata kunci static, artinya kita membuat fungsi yang dapat dipanggil tanpa harus membuat instansiasi objek.
	✧ TipeDataKembalian adalah tipe data dari nilai yang dikembalikan setelah fungsi dieksekusi.
	✧ namaFungsi() adalah nama fungsinya. Biasanya ditulis dengan huruf kecil di awalnya. Lalu, kalau terdapat lebih dari satu suku kata, huruf awal di kata kedua ditulis kapital.

Cara Memanggil/ Eksekusi Fungsi	Setelah kita membuat fungsi, selanjutnya kita akan mengeksekusi fungsinya dengan menuliskan : ✧ nama fungsi yang ingin kamu panggil/gunakan ✧ sesuatu yang perlu diketahui fungsi, tuliskan dalam tanda kurung (...). misal: ketika menggunakan method System.out.println("Hello world!"); Fungsi dapat dipanggil dari fungsi main maupun dari fungsi yang lain.

Fungsi dengan Parameter	Parameter adalah variabel yang menampung nilai untuk diproses di dalam fungsi. Parameter berperan sebagai input untuk fungsi. Struktur dasarnya seperti ini:
	
	`static TipeData namaFungsi(TipeData namaParameter, TipeData namaParameterLain){`
	`  // kode fungsi`
	`}`
	
	Penjelasan:
	
	✧ Parameter ditulis di antara tanda kurung (...);
	✧ Parameter harus diberikan tipe data;
	✧ Bila terdapat lebih dari satu parameter, maka dipisah dengan tanda koma.

FungsiParameter.java

```
public class FungsiParameter {
    public static void main(String[] args) {
        // eksekusi fungsi ucapSalam()
        ucapSalam();
```

```
        //eksekusi fungsi ucapin dengan parameter string
        ucapin("Halo!");
        ucapin("Selamat datang di Praktikum Algoritma dan Struktur Data");
    }

    static void ucapSalam(){
        System.out.println("Selamat Pagi");
    }

    static void ucapin(String ucapan){
        System.out.println(ucapan);
    }
}
```

Fungsi yang Mengembalikan Nilai	Kadang-kadang, setelah fungsi memproses data yang diinputkan melalui parameter, ada juga fungsi yang harus mengembalikan nilai agar dapat diolah pada proses berikutnya. Pengembalian nilai pada fungsi menggunakan kata kunci return.
Pemanggilan Fungsi di Fungsi Lain	Fungsi-fungsi dapat saling memanggil untuk memproses data. Ini merupakan kegunaan dari fungsi sendiri yaitu memecah kode menjadi bagian-bagian kecil sehingga lebih mudah diatur.

FungsiReturn.java

```
public class BangunRuang {
    public static void main(String[] args) {
        int s = 12;
        int luas = luasKubus(s);
```

```
        System.out.println(luas);

        System.out.println("Luas Persegi dengan panjang sisi 5 adalah " +
luasPersegi(5));
    }

    // membuat fungsi luasPersegi()
    static int luasPersegi(int sisi){
        return sisi * sisi;
    }

    // membuat fungsi luasKubus()
    static int luasKubus(int sisi){
        // memanggil fungsi luasPersegi
        return 6 * luasPersegi(sisi);
    }
}
```

Fungsi Static dan Non-Static	Kata kunci static akan membuat fungsi dapat dieksekusi langsung, tanpa harus membuat instansiasi objek dari class.
	Apabila kita tidak membuat objek untuk memanggil fungsi non-static, maka akan terjadi error.

FungsiStatic.java

```
public class FungsiStatic {
    // fungsi non-static
    void makan(String makanan){
        System.out.println("Hi!");
        System.out.println("Saya sedang makan " + makanan);
    }

    // fungsi static
```

```
    static void minum(String minuman){
        System.out.println("Saya sedang minum " + minuman);
    }

    // fungsi main
    public static void main(String[] args) {

        // pemanggilan fungsi static
        minum("Kopi");

        // mambuat instansiasi objek saya dari class FungsiStatic
        FungsiStatic saya = new FungsiStatic();
        // pemanggilan fungsi non-static
        saya.makan("Nasi Goreng");
    }
}
```

Percabangan

if
> hanya memiliki satu pilihan;
	`if ( suatu_kondisi ) {`
	`  //lakukan sesuatu kalau kondisi benar`
	`  //kerjakan ini juga`
	
	`}`

If-else
 	
> memiliki dua pilihan
	`if ( suatu_kondisi ) {`
	`  //lakukan sesuatu kalau kondisi benar`
	`  //kerjakan ini juga`
	
	`}`
	`else {`
	`  //kerjakan ini kalau kondisi if salah`
	`}`

If-else if,
switch case
 	
> memiliki lebih dari dua pilihan;
 	
> SWITCH/CASE adalah bentuk lain dari IF/ELSE/IF
	`if (suatu kondisi) {`
	`  // maka kerjakan ini`
	`  // kerjakan perintah ini juga`
	`  // ...`
	`} else if (kondisi lain) {`
	`  // kerjakan ini`
	`  // kerjakan ini juga`
	`  // ...`
	`} else if (kondisi yang lain lagi) {`
	`  // kerjakan perintah ini`
	`  // kerjakan ini juga`
	`  // ...`
	`} else {`
	`  // kerjakan ini kalau`
	`  // semua kondisi di atas`
	`  // tidak ada yang benar`
	`  // ...`
	`}`

	`switch(variabel){`
	`  case 1:`
	`  // kerjakan kode ini`
	`  // kode ini juga`
	`  break;`
	`  case 2:`
	`  // kerjakan kode ini`
	`  // kode ini juga`
	`  break;`
	`  case 3:`
	`  // kerjakan kode ini`
	`  // kode ini juga`
	`  break;`
	`  default:`
	`  // kerjakan kode ini`
	`  // kode ini juga`
	`  break;}`

Tambahan:

1. Percabangan bersarang adalah percabangan dalam percabangan, jenis percabangannya dapat berbeda/tidak harus sama

2. Penggunaan operator logika dalam percabangan bisa membuat percabangan menjadi lebih singkat.

Percabangan.java

```
public class Percabangan {
    public static void main(String[] args) {
            int choice = 3;
            System.out.println("IF");

            if(choice == 1){
                System.out.println("You selected 1.");
            }
            else if(choice == 2 || choice == 3){
                System.out.println("You selected 2 or 3.");
            }
            else if(choice == 4){
                System.out.println("You selected 4.");
            }
            else{
                System.out.println("Your Choice is not 1,2,3 or 4");
            }

            System.out.println("Switch-Case");
            switch(choice) {
                case 1:
                        System.out.println("You selected 1.");
                break;

                case 2:
                case 3:
                        System.out.println("You selected 2 or 3.");
                break;

                case 4:
                        System.out.println("You selected 4.");
                break;

                default:
                        System.out.println("Please enter a choice between 1-4.");
            }
    }
}
```

Perulangan

Counted Loop

for	`for( int hitungan = 0; hitungan <= 10; hitungan++ ){`
	`// blok kode yang akan diulang}`
	
	✧ variabel hitungan tugasnya untuk menyimpan hitungan pengulangan.
	✧ hitungan <= 10 artinya selama nilai hitungannya lebih kecil atau sama dengan 10, maka pengulangan akan terus dilakukan. Dengan kata lain, perualangan ini akan mengulang sebanyak 10 kali.
	✧ hitungan++ fungsinya untuk menambah satu (+1) nilai hitungan peda setiap pengulangan.
	✧ Blok kode For dimulai dengan tanda '{' dan diakhiri dengan '}'.

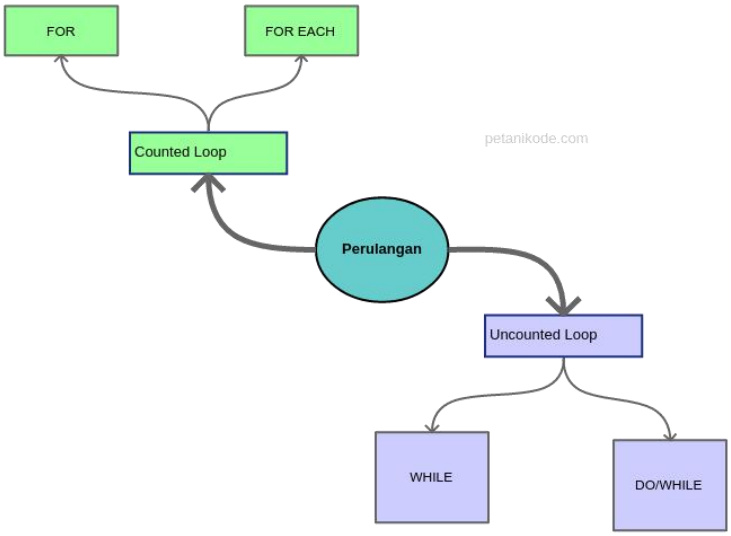
foreach
 	
> digunakan untuk menampilkan isi dari *array*.
 	
>selengkapnya, nanti bisa di pelajari pada materi Array.
	`for ( int item : dataArray ) {`
	`// blok kode yang diulang}`
	
	✧ variabel item akan menyimpan nilai dari array
	✧ Kita bisa baca seperti ini: "Untuk setiap item dalam dataArray, maka lakukan perulangan"

Uncounted Loop

while
 	
> while bisa kita artikan selama.
 	
> cara kerja perulangan ini seperti percabangan, ia akan melakukan perulangan selama
	`while ( kondisi ) {`
	`// blok kode yang akan diulang}`
	
	✧ kondisi bisa kita isi dengan perbandingan maupun variabel boolean. Kondisi ini hanya memiliki nilai true dan flase.

kondisinya bernilai true.	✧ Perulangan while akan berhenti sampai kondisi bernilai false.

do-while
 	
> cara kerja perulangan do-while se benarnya sama seperti perulangan while.
 	
> bedanya, do-while melakukan satu kali perulangan dulu. kemudian mengecek kondisinya.
	do {
	// blok kode yang akan diulang} while (kondisi);
	✧ Jadi kerjakan dulu (Do), baru di cek kondisinya while(kondisi).
	✧ Kalau kondisi bernilai ture, maka lanjutkan perulangan.



Tambahan:

- Perulangan juga dapat bersarang. Perulangan bersarang maksudnya, perulangan dalam perulangan atau disebut juga *nested loop,* sering digunakan pada array multi dimensi.
- Jenis perulangan di dalam perulangan bisa berbeda, misalnya di dalam perulangan while ada perulangan for.

Perulangan.java

```
public class Perulangan {
    public static void main(String[] args) {
            int i=0;
            // membuat array
            int angka[] = {3,1,42,24,12};


            for (i = 0; i <= 10; i = i + 2) {
                System.out.println(i);
            }
            i=0;

            while(i<=10){
                System.out.println(i);
                i=i+2;
            }

            do{
                System.out.println(i);
                i=i+2;
            }while(i<=10)

           // menggunakan perulangan For each untuk menampilkan angka
           for( int x : angka ){
                System.out.print(x + " ");
           }

    }
}
```