PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK

MODUL III PENGENALAN DART

Disusun Oleh :

Nadia Putri Rahmaniar / 2211104013

S1 SE-06-01

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 SOFTWARE ENGINEERING FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO 2024

GUIDED

Apa itu Dart?

Dart adalah bahasa pemrograman yang dibuat oleh Google untuk mengembangkan aplikasi lintas platform, termasuk aplikasi mobile, desktop, dan web. Bahasa ini menawarkan fitur-fitur modern dan memiliki sintaks yang mirip dengan beberapa bahasa pemrograman populer seperti Java, C#, JavaScript, Swift, dan Kotlin.

Contoh kode:

```
void main() {
  print('Praktikum Perangkat Bergerak');
}
```

Karakteristik Dart:

- **Statically typed**

```
var name = 'Praktikum PPB';
 String language = 'Dart';
```

- **Type inference**

```
Hello Praktikum PPB. Welcome to Dart!
```

- **String interpolation**

```
print('Hello $name. Welcome to $language!');
```

- **Multi-paraidgm**
    1. Dart mendukung paradigma Pemrograman Berorientasi Objek (OOP), memungkinkan pembuatan program yang terstruktur dengan konsep seperti objek, kelas, properti, konstruktor, dan lain sebagainya.
    2. Mendukung paradigma Pemrograman Fungsional (FP), sehingga memungkinkan penerapan fitur-fitur seperti higher-order functions, rekursi, first-class citizens, dan lainnya.

A. Variable

- **Variabel dengan var**

  Var digunakan untuk mendeklarasikan variabel tanpa perlu menentukan tipe data secara eksplisit. Dart akan secara otomatis menginfer tipe data variabel tersebut berdasarkan nilai yang diinisialisasi.

  - Contoh Code

```dart
void main() {
  // VARIBLE
  // Menggunakan var
  var name = "Alice"; // Tipe data String
  var age = 25; // Tipe data Integer
  print("Nama: $name, Usia: $age");
}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
Nama: Alice, Usia: 25
```

- **Type Annotation**

  Type annotation digunakan untuk mendeklarasikan variabel dengan secara eksplisit menyebutkan tipe datanya. Hal ini memungkinkan penentuan tipe variabel secara langsung saat deklarasi.

  - Contoh code

```dart
void main(){
  // Type annotation
  String nama = "Bob"; // Tipe data String
  int usia = 30; // Tipe data Integer
  print("Nama: $nama, Usia: $usia");

}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
Nama: Bob, Usia: 30
```

- **Multiple var**

  Dart memungkinkan deklarasi beberapa variabel sekaligus dengan tipe data yang sama dalam satu baris

  - Contoh Code

```dart
void main(){
// Multiple variable
String firstName, lastName; // Tipe data String
firstName = "Charlie";
lastName = "Brown";
print("Nama Lengkap: $firstName $lastName");
}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
Nama Lengkap: Charlie Brown
```

B. Statement Control

- **IF-ELSE Statement**

  Untuk menangani suatu kondisi dalam program, kita dapat menggunakan pernyataan if. Apabila kondisi tersebut bernilai benar (true), blok kode di dalamnya akan dijalankan. Sebaliknya, jika kondisinya salah (false), kita dapat memanfaatkan pernyataan else untuk menjalankan blok kode lain sebagai alternatif.

  - Contoh Code

```dart
void main(){
  // IF ELSE
  // if-else statement
  var openHours = 8;
  var closedHours = 21;
  var now = 17;

  if (now > openHours && now < closedHours) {
    print("Hello, we're open");
  } else {
    print("Sorry, we've closed");
  }

  //Bentuk singkat if else
  var shopstatus = now >= openHours && now <= closedHours
      ? "Hello, we're open"
      : "Sorry, we've close";
  print(shopstatus);
}
```

- Output

```
PS D:\School\PPB\cobacode> dart run main.dart
Hello, we're open
Hello, we're open
```

- **Switch-case Statement**

  Struktur kontrol yang memungkinkan pemilihan blok kode berdasarkan nilai suatu ekspresi. Ini memudahkan pengelompokan berbagai kondisi dan lebih terstruktur dibandingkan dengan pernyataan if-else. Setiap kasus mewakili nilai tertentu, dan jika ada yang cocok, blok kode terkait akan dijalankan. Jika tidak ada yang cocok, pernyataan default dapat digunakan.

  - Contoh code

```dart
void main(){
 var day = 3; // Misalkan 1 = Senin, 2 = Selasa, dst.
  switch (day) {

    case 1:
      print("Senin");
      break;
    case 2:
      print("Selasa");
      break;
    case 3:
      print("Rabu");
      break;
    case 4:
      print("Kamis");
      break;
    case 5:
      print("Jumat");
      break;
    case 6:
      print("Sabtu");
      break;
    case 7:
      print("Minggu");
      break;
    default:
      print("Hari tidak valid");
  }
}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
Rabu
```

C. Looping

- **For Loops**

  Gunakan perulangan for ketika jumlah iterasi sudah diketahui secara pasti, misalnya untuk melakukan looping sebanyak 10 kali dengan kenaikan 1 pada setiap iterasi.

  - Contoh code

```dart
void main (){
 // LOOPING
  for (int i = 1; i <= 5; i++) {
    print(i);
  }
}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
1
2
3
4
5
```

- **While Loops**

  Gunakan perulangan while ketika tidak diketahui secara pasti kapan perulangan akan berhenti, misalnya dengan meminta pengguna untuk terus memasukkan angka hingga mereka menginput tanda " ".

  - Contoh Code

```dart
// While loops
  int i = 1; // Deklarasi variabel
  while (i <= 5) {
    print(i);
    i++; // Tambahkan 1 ke i setelah setiap iterasi
  }
```

- Output

```
PS D:\School\PPB\cobacode> dart run main.dart
1
2
3
4
5
```

D. List

List adalah tipe data yang digunakan untuk menyimpan kumpulan nilai secara berurutan. List bisa berisi elemen dengan tipe data yang sama atau berbeda. Anda dapat menginisialisasi List dengan nilai-nilai tertentu, atau membuatnya kosong dan mengisi elemen-elemen di kemudian hari.

- **Fixed length list**

  Fixed length list adalah jenis list yang ukurannya telah ditentukan saat inisialisasi dan tidak dapat diubah. Setelah dibuat, jumlah elemen dalam list ini tetap, sehingga tidak bisa menambah atau menghapus elemen, tetapi nilainya masih bisa diubah.

  - Contoh Code

```dart
void main(){
  // LIST
  // Membuat fixed-length list dengan panjang 3
  List<int> fixedList =
      List.filled(3, 0); // List dengan 3 elemen,diisi dengan 0

  // Mengubah elemen dalam list
  fixedList[0] = 10;
  fixedList[1] = 20;
  fixedList[2] = 30;

  print(fixedList); // Output: [10, 20, 30]
}
```

  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
[10, 20, 30]
```

- **Growable List**

  Gunakan growable list apabila memiliki banyak object yang tidak menentu atau banyaknya object yang terus bertambah.

  - Contoh code

```dart
void main(){

 // Membuat growable list (panjangnya bisa berubah)
  List<int> growableList = [];

  // Menambahkan elemen ke dalam list
  growableList.add(10);
  growableList.add(20);
  growableList.add(30);

  print(growableList); // Output: [10, 20, 30]

  // Menambahkan lebih banyak elemen
  growableList.add(40);
  growableList.add(50);

  print(growableList); // Output: [10, 20, 30, 40, 50]

  // Menghapus elemen dari list
  growableList.remove(20);

  print(growableList); // Output: [10, 30, 40, 50]
}
```

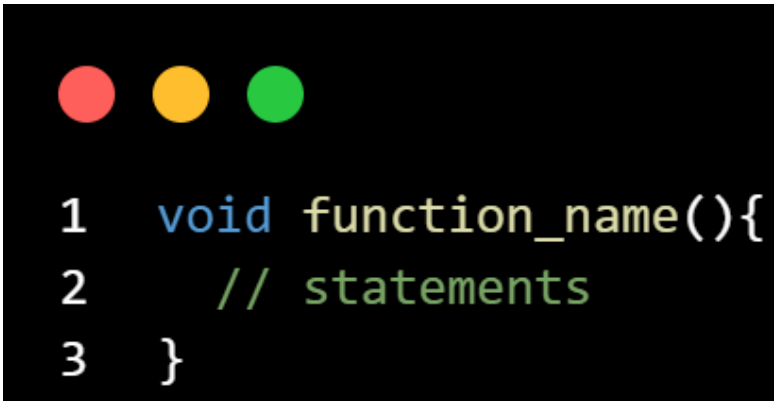
  - Output

```
PS D:\School\PPB\cobacode> dart run main.dart
[10, 20, 30]
PS D:\School\PPB\cobacode> dart run main.dart
[10, 20, 30]
[10, 20, 30, 40, 50]
[10, 30, 40, 50]
```

E. Fungsi

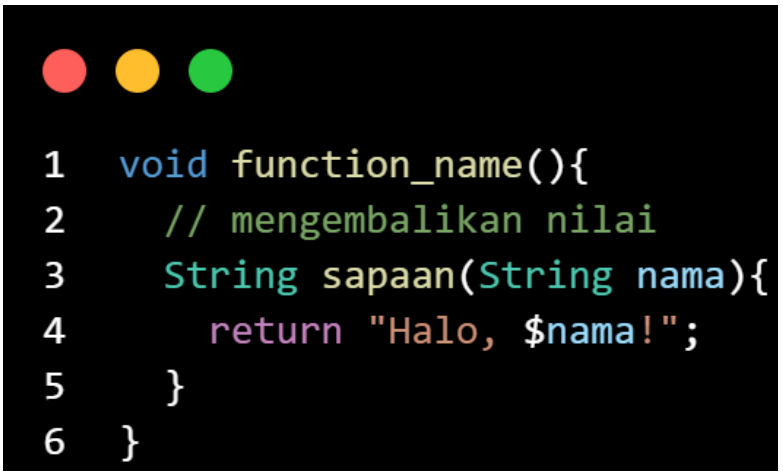
Dalam bahasa pemrograman yang mendukung Pemrograman Berorientasi Objek, fungsi atau prosedur memegang peranan yang krusial. Untuk mencapai kualitas kode yang tinggi, programmer dapat menerapkan berbagai prinsip pemrograman yang umum, seperti SOLID, KISS, dan YAGNI. Semua prinsip tersebut mengedepankan pemisahan tanggung jawab (separation of concerns), yang berarti setiap bagian kode memiliki tanggung jawab masing-masing dan meminimalkan penggunaan boilerplate code.

- **Mendefinisikan Fungsi**



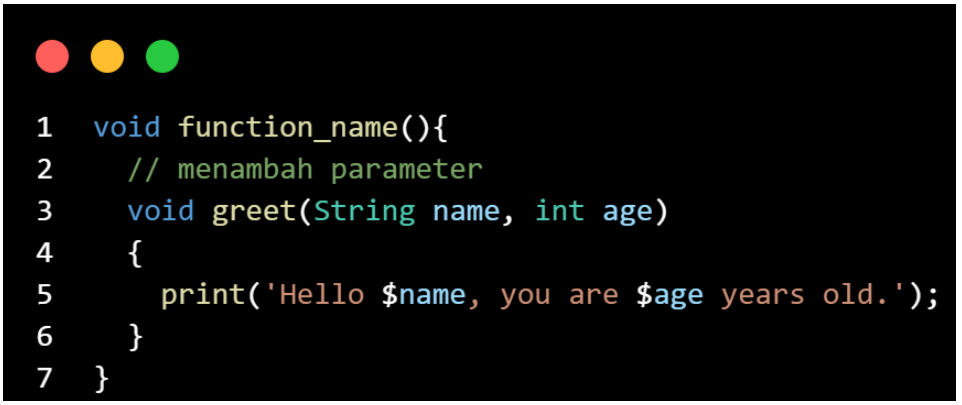
```
void function_name(){
    // statements
}
```

- **Memanggil Mengembalikan Nilai**



```
void function_name(){
    // mengembalikan nilai
    String sapaan(String nama){
        return "Halo, $nama!";
    }
}
```

- **Menambahkan Parameter**



```
void function_name(){
    // menambah parameter
    void greet(String name, int age)
    {
        print('Hello $name, you are $age years old.');
    }
}
```

UNGUIDED

1. Buatlah sebuah fungsi dalam Dart yang menerima sebuah nilai dari user, lalu melakukan percabangan untuk memberikan output berdasarkan kondisi berikut:

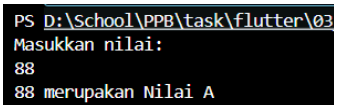
   Deskripsi :

   - Jika nilai lebih besar dari 70, program harus mereturn "Nilai A".
   - Jika nilai lebih besar dari 40 tetapi kurang atau sama dengan 70, program harus mereturn "Nilai B".
   - Jika nilai lebih besar dari 0 tetapi kurang atau sama dengan 40, program harus mereturn "Nilai C".
   - Jika nilai tidak memenuhi semua kondisi di atas, program harus mereturn teks kosong.
   - Sampel Input: 80 Sampel Output: 80 merupakan Nilai A Sampel Input: 5 Sampel Output: 50 merupakan Nilai B

   Source Code :

```dart
import 'dart:io';

void main() {
  // Meminta input dari user
  print('Masukkan nilai:');
  String? userInput = stdin.readLineSync();  // Baca input sebagai string

  // Coba parsing input ke int, jika gagal gunakan -1
  int nilai = int.tryParse(userInput ?? '') ?? -1;

  // Panggil fungsi evaluasiNilai dan simpan hasilnya
  String evaluasiResult = evaluasiNilai(nilai);

  // Cek apakah hasil evaluasi valid
  if (evaluasiResult.isNotEmpty) {
    print('$nilai merupakan $evaluasiResult');  // Tampilkan hasil evaluasi
  } else {
    print('Nilai tidak valid');  // Jika input tidak valid
  }
}

// Fungsi untuk mengevaluasi nilai
String evaluasiNilai(int nilai) {
  if (nilai > 70) {
    return 'Nilai A';  // Jika nilai lebih dari 70
  } else if (nilai > 40 && nilai <= 70) {
    return 'Nilai B';  // Jika nilai antara 41 dan 70
  } else if (nilai > 0 && nilai <= 40) {
    return 'Nilai C';  // Jika nilai antara 1 dan 40
  } else {
    return '';  // Jika nilai tidak valid
  }
}

```

Output :



Penjelasan :

Kode di atas menerapkan logika percabangan untuk menentukan kategori nilai. Pertama, program meminta pengguna untuk memasukkan sebuah nilai, yang kemudian akan ditampilkan di konsol. Setelah itu, nilai yang dimasukkan akan diubah menjadi tipe data integer. Jika proses konversi gagal, nilai akan diatur menjadi -1 sebagai nilai default.

Selanjutnya, nilai yang telah dikonversi tersebut akan dievaluasi melalui sebuah fungsi. Jika nilai tersebut lebih dari 70, hasilnya adalah 'Nilai A'; jika nilainya berada di antara 41 hingga 70, hasilnya 'Nilai B'; dan jika nilainya berada di antara 1 hingga 40, hasilnya adalah 'Nilai C'. Jika nilai yang dimasukkan tidak memenuhi semua kriteria tersebut, fungsi akan mengembalikan string kosong. Output yang ditampilkan adalah sebuah string yang menunjukkan nilai yang telah dimasukkan dan kategorinya, atau pesan yang menyatakan bahwa nilai tersebut tidak valid.

2. Buatlah sebuah program dalam Dart yang menampilkan piramida bintang dengan menggunakan for loop. Panjang piramida ditentukan oleh input dari user.

Source Code :

```dart
import 'dart:io';

void main() {
  // Inputan dari user
  stdout.write('Masukkan panjang angka piramida : ');
  int? length = int.parse(stdin.readLineSync()!);

  // Menggunakan perulangan For
   for (int i = 1; i <= length; i++) {
    // Cetak spasi
    stdout.write(' ' * (length - i));

    // Cetak bintang
    stdout.write('*' * (2 * i - 1));

    // Pindah baris baru
    print('');
  }
}
```

Output :

```
PS D:\School\PPB\task\flutter\03_pengenalan_dart\unguided\task2> dart looping.dart
Masukkan panjang piramida : 8
       *
      ***
     *****
    *******
   *********
  ***********
 *************
***************
```

Penjelasan :

Program ini dirancang untuk mencetak piramida bintang dengan tinggi yang ditentukan oleh pengguna. Program ini memanfaatkan loop for. Proses dimulai dengan meminta pengguna untuk memasukkan tinggi piramida melalui fungsi stdin.readLineSync(). Input tersebut kemudian diubah menjadi tipe data integer menggunakan int.parse(). Jika pengguna tidak memasukkan nilai yang valid, program akan mengalami error, tetapi dalam implementasi ini diasumsikan bahwa input selalu valid.

Setelah mendapatkan tinggi piramida, program menggunakan dua loop for. Loop pertama berfungsi untuk mencetak spasi sehingga bintang akan tersusun rapi dalam bentuk piramida. Loop kedua mencetak karakter bintang (*) sesuai dengan jumlah yang

ditentukan, di mana jumlah bintang pada setiap baris bertambah seiring bertambahnya baris. Setiap kali sebuah baris selesai dicetak, program berpindah ke baris baru menggunakan print(''). Hasil akhir dari program ini adalah tampilan piramida bintang yang tinggi sesuai dengan input yang diberikan oleh pengguna.

3. Buatlah program Dart yang meminta input berupa sebuah bilangan bulat dari user, kemudian program akan mengecek apakah bilangan tersebut merupakan bilangan prima atau bukan.
    - Sampel Input: 23
    - Sampel Output: bilangan prima
    - Sampel Input: 12 Sampel Output: bukan bilangan prima

Source Code :

```dart
import 'dart:io';

void main() {
  // Inputan dari user
  stdout.write('Masukkan Bilangan Bulat: ');
  int? bill = int.parse(stdin.readLineSync()!);

  // Cek inputan bilangan dari user
  bool is_prime(int bill) {
    if (bill <= 1) {
      // Bilangan prima tidak boleh sama dengan atau kurang dari 1
      return false;
    }

    for (int i = 2; i <= bill ~/ 2; i++) {
      if (bill % i == 0) {
        // Jika ada pembagi selain satu dan dirinya sendiri, maka itu bukan prima
        return false;
      }
    }
    // Jika hanya ada satu pembagi dan dirinya sendiri = bilangan prima
    return true;
  }

  // Menentukan sebuah bilangan prima atau bukan
  if (is_prime(bill)) {
    print('$bill adalah bilangan prima');
  } else {
    print('$bill bukan bilangan prima');
  }
}
```

Output :

```
PS D:\School\PPB\task\flutter\03_pengenalan_dart
Masukkan Bilangan Bulat: 22
22 bukan bilangan prima
```

Penjelasan :

Program ini dirancang untuk menerima input berupa bilangan bulat dari pengguna dan menentukan apakah bilangan tersebut merupakan bilangan prima atau tidak. Awalnya, pengguna diminta untuk

memasukkan bilangan bulat, setelah itu program akan melakukan pemeriksaan terhadap angka tersebut.

Program ini menggunakan sebuah fungsi yang mengimplementasikan loop untuk mencari apakah ada angka lain yang dapat membagi bilangan tersebut selain 1 dan dirinya sendiri. Sebuah bilangan dikatakan prima jika hanya dapat dibagi oleh 1 dan bilangan itu sendiri. Jika ditemukan angka lain yang dapat membagi bilangan tersebut, maka bilangan itu bukanlah bilangan prima, dan program akan menampilkan pesan bahwa bilangan tersebut bukan prima.

Sebaliknya, jika tidak ada pembagi lain, bilangan tersebut dianggap sebagai bilangan prima.Output dari program ini adalah pesan yang menyatakan apakah bilangan yang dimasukkan oleh pengguna adalah bilangan prima atau bukan.