LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK

MODUL III

PENGENALAN DART

Disusun Oleh :

Dewi Atika Muthi / 2211104042

SE-06-02

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

BAB I
PENDAHULUAN

A. DASAR TEORI

Dart adalah bahasa pemrograman modern yang dikembangkan oleh Google, bersifat open-source, dan digunakan untuk pengembangan aplikasi multiplatform seperti mobile, desktop, dan web. Dart mendukung paradigma pemrograman Object-Oriented Programming (OOP) dan Functional Programming (FP). Dart memiliki kemiripan dengan bahasa pemrograman populer lainnya seperti Java, C#, dan JavaScript, yang membuatnya mudah dipelajari oleh pengembang dengan latar belakang tersebut.

Beberapa karakteristik penting dari Dart adalah sebagai berikut:

- **Statically Typed**: Variabel memiliki tipe data tetap saat dijalankan, yang memungkinkan deteksi kesalahan sebelum runtime.

```
var name = 'Praktikum PPB';

String language = 'Dart';
```

- **Type Inference**: Dart dapat menebak tipe data berdasarkan nilai yang diberikan saat deklarasi variabel.

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

- **String Interpolation**: Memungkinkan integrasi variabel dalam string secara langsung menggunakan tanda dollar ($), mempermudah pengelolaan teks dalam program.

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

- **Multi-Paradigm**: Dart mendukung OOP dan FP, memberikan fleksibilitas dalam pendekatan pengembangan.

Dalam praktikum ini, mahasiswa mempelajari konsep dasar Dart, seperti variabel, tipe data, kontrol alur (percabangan dan perulangan), serta fungsi. Konsep-konsep ini esensial untuk memahami bagaimana bahasa Dart bekerja dalam pengembangan aplikasi Flutter, yang digunakan untuk membuat antarmuka pengguna (UI) aplikasi multiplatform.

B. MAKSUD DAN TUJUAN

Tujuan dari praktikum ini adalah:

1. Mahasiswa dapat memahami dasar-dasar pemrograman menggunakan bahasa Dart.
2. Mahasiswa mampu mengimplementasikan logika percabangan dan perulangan dalam Dart.
3. Mahasiswa dapat membuat fungsi yang efisien dan memahami penggunaan parameter dan return value.
4. Mahasiswa dapat menerapkan Dart dalam pembuatan aplikasi multiplatform.

BAB II
IMPLEMENTASI

A. PRAKTIKUM

1. Variable

Soal Studi Case

Menyapa user dengan menampilkan nama, usia, lokasi, dan kondisi cuaca saat ini disimpan dalam variabel dengan camelCase.

Source code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

  // PENAMAAN VARIABLE MENGGUNAKAN CAMELCASE
  print('=====Variable=====');
  var name = 'Tika';
  String loc = 'Bandung';
  var age = 21;
  final cuaca = 'Cerah';

  print('Hello Customer, $name.\nYou are $age');
  print('Hari ini, $cuaca di $loc');

  print('\n');
}
```

Screenshoot Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\tempCodeRunnerFile.dart"
Praktikum Perangkat Bergerak

=====Variable=====
Hello Customer, Tika.
You are 21
Hari ini, Cerah di Bandung
```

Deskripsi Program

Variabel dalam Dart digunakan untuk menyimpan data yang dapat berupa berbagai tipe, seperti `'var'`, `'String'`, dan `'final'` seperti contoh di atas yang menggunakan camelCase.

- **CamelCase** adalah gaya penulisan yang menggabungkan dua kata atau lebih, tetapi kapitalisasi hanya pada kata pertama dari setiap kata setelah kata pertama.
- `'var'` adalah cara untuk mendeklarasikan variabel **tanpa menentukan tipe** datanya secara eksplisit. Di programa berisikan nama `'Tika'` dan umur `21`
- Sedangkan tipe data dapat ditetapkan secara eksplisit seperti pada `'String'`. Dalam kode dicontohkan pada lokasi `'Bandung'`
- Variabel `'final'` digunakan untuk **nilai yang tidak bisa diubah** setelah inisialisasi.

Dalam Dart, untuk menampilkan nilai variabel langsung ke dalam string menggunakan tanda `‘$’`, seperti `‘print('Hello $name')’`.

Tipe Data Primitif:

Tipe	Deskripsi	Contoh
int	Integer (bilangan bulat)	1, -2, 0
double	Bilangan desimal	3.14, 10.53, -23.23
num	Bilangan bulat dan bilangan desimal	7, 3.14, -12.00
bool	Boolean	True, False
String	Text yang terdiri dari 0 atau beberapa karakter	'Faza', 'Hasna'
List	Sekuumpulan nilai	[0,1,2,3,4], ['a', 'b', 'c']
Map	Pasangan key-value	{"a": "aku", "b": 12}
dynamic	Tipe apapun	

2. **Statement Control (if-else)**

Soal studi case:

Misal kita ingin menampilkan status operasional sebuah kafe berdasarkan waktu saat ini. Sistem ini menggunakan struktur kontrol if-else untuk menentukan apakah kafe sedang buka, tutup, atau sedang break pada jam tertentu.

Source code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// STATEMENT CONTROL PERULANGAN : if else
  print('====Statement Control: IF-ELSE====');
  var open = 8;
  var close = 23;
  var now = 12;

  print('Sekarang jam $now, status cafe:');

  if (now == 12) {
    print('Cafe kami sedang break');
  } else if (now >= open && now < close) {
    print('Cafe ini buka');
  } else {
    print('Cafe sudah tutup');
  }
}

```

Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====Statement Control: IF-ELSE====
Sekarang jam 12, status cafe:
Cafe kami sedang break
```

Deskripsi Program:

Deklarasi Variabel:

- `open` untuk menyimpan jam buka kafe (jam 08.00).

- `close` untuk menyimpan jam tutup kafe (jam 23.00).
- `now` untuk menyimpan waktu saat ini (jam 12.00 dalam program).

Penggunaan IF-ELSE Statement:

- Program akan mencetak status kafe berdasarkan waktu saat ini (now):
- `Jika jam sekarang adalah 12`, program akan `mencetak "Cafe kami sedang break"`.
- Jika jam sekarang `lebih besar atau sama dengan jam buka (08.00)` dan `kurang dari jam tutup (23.00)`, program akan `mencetak "Cafe ini buka"`.
- Jika waktu saat ini `sebelum 08.00 atau setelah 23.00`, program akan `mencetak "Cafe sudah tutup"`.

Karena dalam program dikatakan waktu sekarang adalah **jam 12**, maka output yang keluar adalah `“Café kami sedang break”`

Ada cara lain juga untuk statement control dengan if-else ini, yaitu dengan **true : false,** missal:

Soal Studi Case:
Mengecek status operasional sebuah toko berdasarkan jam buka dan waktu saat ini.

Source code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// STATEMENT CONTROL PERULANGAN : if else-true : false
  print('====IF ELSE DENGAN TRUE FALSE====');
  var nowT = 10;
  var openT = 7;
  var toko = nowT > openT ? "Toko Buka" : "Toko Tutup";
  print('Sekarang jam $nowT, status toko:');
  print(toko);
}

```

Output:

```
Connecting to VM Service at ws://127.0.0.1:50586/fVuq5dPaGAE=/ws
Connected to the VM Service.
Praktikum Perangkat Bergerak

====IF ELSE DENGAN TRUE FALSE====
Sekarang jam 10, status toko:
Toko Buka
```

Deskripsi program:

dengan menggunakan operator ternary (if-else true-false) untuk menentukan apakah toko sedang buka atau tutup, dan menampilkan status tersebut. Program ini lebih ringkas daripada penggunaan statement if-else biasa.

Deklarasi Variabel:

- `nowT` untuk menyimpan waktu sekarang (jam 10.00).
- `openT` untuk menyimpan jam buka toko (pukul 07.00).

Penggunaan Operator Ternary:

- Operator ternary digunakan untuk memeriksa apakah waktu saat ini (nowT) lebih besar dari jam buka (openT): `(nowT > openT)`
- Jika waktu saat ini lebih besar dari jam buka `(true)`, maka variabel toko akan diisi dengan string `"Toko Buka"`.
- Jika tidak `(false)`, variabel toko akan diisi dengan string `"Toko Tutup"`.

Isi jam sekarnag yang dimasukkan di program adalah `jam 10.00` maka ditampilkan status `“Toko Buka”`

3. **Statement Control (Switch Case)**
   Contoh Studi Case:
   Membuat program mengevaluasi nilai seorang siswa berdasarkan nilai grade)yang dimasukkanSetiap nilai grade memiliki evaluasi yang berbeda, mulai dari sangat bagus hingga lumayan.

   Source Code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// SWITCH CASE : NILAI
  print('====SWITCH CASE====');
  var grade = 'a';

  print('Grade kamu: $grade');

  // SWITCH CASE untuk grade
  switch (grade) {
    case 'a':
      print("Nilai kamu sangat bagus");
      break;
    case 'b':
      print("Nilai kamu baik");
      break;
    case 'c':
      print("Nilai sudah cukup");
      break;
    case 'd':
      print("Nilai kamu lumayan");
      break;
  }
  print('\n');
}

```

   Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====SWITCH CASE====
Grade kamu: a
Nilai kamu sangat bagus
```

   Deskripsi Program:

Program menggunakan statement kontrol Switch-Case untuk mencetak evaluasi atau komentar terkait grade yang diterima.

Deklarasi Variabel:

- Variabel grade yang menyimpan grade siswa (pada program `grade = 'a'`)

Penggunaan Switch-Case:

- Program memeriksa nilai grade yang tersimpan di variabel grade:
- Jika nilai adalah `'a'`, program mencetak `"Nilai kamu sangat bagus"`.
- Jika nilai adalah `'b'`, program mencetak `"Nilai kamu baik"`.
- Jika nilai adalah `'c'`, program mencetak `"Nilai sudah cukup"`.
- Jika nilai adalah `'d'`, program mencetak `"Nilai kamu lumayan"`.
- Break digunakan setelah setiap case untuk menghentikan eksekusi lebih lanjut setelah case yang sesuai ditemukan.

Pada program diisikan nilai `'a'`, maka progam menampilkan output `"Nilai kamu sangat bagus"`

4. **Looping (For Loop dan While Loop)**

Contoh Studi Case:

Menampilkan deretan angka dari 1 hingga 5 menggunakan for loop

Source Code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

  // For loop untuk mencetak angka 1 sampai 5
  print('====FOR LOOP====');
  for (int i = 1; i <= 5; i++) {
    print(i);
  }
}

```

Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====FOR LOOP====
1
2
3
4
```

Deskripsi program:

Deklarasi dan Penggunaan For Loop:

- Menginisialisasi variabel i = 1 sebagai angka awal.
- Perulangan akan terus berjalan selama kondisi `i <= 5` terpenuhi.
- Setiap kali perulangan terjadi, nilai i akan bertambah `1 (i++)`.

Mencetak Nilai:

- Tiap iterasi, nilai i akan dicetak ke layar menggunakan print(i), sehingga menghasilkan deretan angka dari 1 hingga 5.

Ada pula perulangan While Loop, Berbeda dengan for loop, while loop akan terus berjalan selama kondisi tertentu masih terpenuhi.

Contoh Studi Case While Loop:
Menampilkan deretan angka dari 1 hingga 5 menggunakan while loop.

Source Code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// While loop
  print('====WHILE LOOP====');
  int i = 1;
  while (i <= 5) {
    print('Angka: $i');
    i++;
  }
}

```

Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====WHILE LOOP====
Angka: 1
Angka: 2
Angka: 3
Angka: 4
Angka: 5
```

Deskripsi program:
Dalam hal ini, perulangan akan menampilkan angka berturut-turut sampai mencapai angka 5.

Deklarasi Variabel:

- Program mendeklarasikan variabel `i` dengan `nilai awal 1`, yang akan digunakan sebagai angka awal dalam perulangan.

Penggunaan While Loop:

- While loop akan terus berjalan selama kondisi `i <= 5` bernilai `true`.
- Nilai `i` akan dicetak ke layar dengan menggunakan `print('Angka: $i')`.

- Setelah mencetak, nilai i akan ditingkatkan 1 dengan perintah `i++` agar perulangan bergerak ke angka berikutnya.

Dalam program nilai `var i = 1` dan kondisinya adalah `I <= 5`.
Loop akan terus berjalan **hingga i mencapai nilai 6**, pada saat itu kondisi `i <= 5` **tidak lagi terpenuhi (False),** sehingga loop **berhenti pada angka 5.**

5. List

Contoh Studi Case:

Buat program list untuk menampung data. Ada dua jenis list yang digunakan di sini: **Fixed-length List** dan **Growable List**.

- **Fixed-length List** adalah list dengan panjang tetap yang tidak dapat diubah setelah dibuat.
- **Growable List** adalah list yang panjangnya dapat berubah dinamis, memungkinkan penambahan dan penghapusan elemen.

Source Code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// LIST
  print('====LIST====');
  List<int> fixedList = List.filled(3, 0);

  fixedList[0] = 10;
  fixedList[1] = 20;
  fixedList[2] = 30;

  print('fixed list : $fixedList');

  print('Membuat list baru:');
  List<int> growableList = [];

  growableList.add(11);
  growableList.add(22);
  growableList.add(33);

  print(growableList);

  print('Menambahkan elemen baru dengan ADD:');
  growableList.add(40);
  growableList.add(50);
  growableList.add(60);

  print(growableList);

  print('Menghapus elemen(22) yang ada dengan REMOVE:');
  growableList.remove(22);

  print(growableList);

  print('\n');
}
```

Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====LIST====
fixed list : [10, 20, 30]
Membuat list baru:
[11, 22, 33]
Menambahkan elemen baru dengan ADD:
[11, 22, 33, 40, 50, 60]
Menghapus elemen(22) yang ada dengan REMOVE:
[11, 33, 40, 50, 60]
```

Deskripsi Program:

Fixed-length List:
List dibuat menggunakan `List.filled(3, 0)` yang menginisialisasi list dengan **panjang 3** dan semua elemen diisi dengan **nilai awal 0.**

Nilai elemen kemudian diubah dengan menugaskan nilai baru pada setiap indeks:

```
fixedList[0] = 10;
fixedList[1] = 20;
fixedList[2] = 30;
```

Program menampilkan list tersebut dengan `print('fixed list : $fixedList')`.

```
fixed list : [10, 20, 30]
```

Growable List:
List yang dapat berubah panjang dibuat menggunakan
`List<int> growableList = [];,`
dimulai sebagai list kosong.

Program **menambahkan elemen-elemen baru** menggunakan metode add:

```
growableList.add(10);
growableList.add(20);
growableList.add(30);
```

Hasilnya kemudian ditampilkan.

```
Membuat list baru:
[10, 20, 30]
```

Penambahan Elemen:
Setelah itu, beberapa elemen baru ditambahkan lagi ke list menggunakan add:

```
growableList.add(40);
growableList.add(50);
growableList.add(60);
```

List diperbarui kemudian dicetak Kembali.

```
Menambahkan elemen baru dengan ADD:
[10, 20, 30, 40, 50, 60]
```

Penghapusan Elemen:
Program menghapus elemen bernilai 20 menggunakan metode remove:

```
growableList.remove(20);
```

Setelah penghapusan, list ditampilkan Kembali.

```
Menghapus elemen yang ada dengan REMOVE:
[10, 30, 40, 50, 60]
```

6. **Fungsi**

Contoh Studi Case:

Membuat program yang dapat mengeksekusikan arirmatika dan mengembalikan nilai pesan.

Source Code:

```dart
void main() {
  print('Praktikum Perangkat Bergerak \n');

// FUNGSI
  print('====FUNGSI====');
  void cetakPesan(String pesan) {
    print(pesan); // mendefinisikan fungsi tanpa mengembalikan nilai
  }

  int perkalian(int a, int b) {
    return a * b; // mengemmbalikan nilai dengan return
  }

  int hasil = perkalian(4, 3); // Memanggil fungsi: perkalian
  print('Hasil perkalian fungsi = $hasil'); // Memanggil fungsi
  cetakPesan(
      'Selesai, sampai bertemu lagi di pertemuan selanjutnya!'); // Memanggil fungsi
}

```

Output:

```
[Running] dart "c:\Users\mutge\prak_ppb3\bin\guided.dart"
Praktikum Perangkat Bergerak

====FUNGSI====
Hasil perkalian fungsi = 12
Selesai, sampai bertemu lagi di pertemuan selanjutnya!
```

Deskripsi program:

Fungsi dan Langkah:

- Fungsi pertama `(cetakPesan)` menerima input berupa teks dan mencetaknya.
- Fungsi kedua `(perkalian)` menerima dua angka integer dan mengembalikan hasil dari perkalian kedua angka tersebut.
- Program memanggil fungsi perkalian dengan `argumen 4 dan 3`, lalu mencetak hasil perkalian tersebut ke layar.
- Fungsi `cetakPesan` dipanggil untuk menampilkan pesan penutup, `"Selesai, sampai bertemu lagi di pertemuan selanjutnya!"`.

B. LATHIAN

1.

Tugas Percabangan (Branching)

Soal:

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

Deskripsi :

a. Jika nilai **lebih besar dari 70**, program harus mereturn **"Nilai A"**.
b. Jika nilai **lebih besar dari 40** tetapi **kurang atau sama dengan 70**, program harus mereturn **"Nilai B"**.
c. Jika nilai **lebih besar dari 0** tetapi **kurang atau sama dengan 40**, program harus mereturn **"Nilai C"**.
d. Jika nilai tidak memenuhi semua kondisi di atas, program harus mereturn teks kosong.

Jawaban:

```dart
import 'dart:io';

// Fungsi utama untuk menjalankan program
void main() {
  // Tugas 1: Menerima input nilai
  print('Masukkan nilai: ');
  String? inputNilai = stdin.readLineSync(); // Membaca input dari pengguna

  if (inputNilai != null && int.tryParse(inputNilai) != null) {
    int nilai = int.parse(inputNilai); // Mengonversi input ke integer
    String hasil;

    // Memeriksa nilai
    if (nilai > 70) {
      hasil = "Nilai A";
    } else if (nilai > 40 && nilai <= 70) {
      hasil = "Nilai B";
    } else if (nilai > 0 && nilai <= 40) {
      hasil = "Nilai C";
    } else {
      hasil = "Nilai tidak valid.";
    }
    print('$nilai merupakan $hasil'); // Menampilkan hasil
  } else {
    print('Input tidak valid, harap masukkan bilangan bulat.');
  }
}

```

Output sample 80:

```
PS C:\Users\mutge\prak_ppb3> dart bin/guided.dart
Masukkan nilai:
80
80 merupakan Nilai A
```

Output sample 50:

```
PS C:\Users\mutge\prak_ppb3> dart bin/guided.dart
Masukkan nilai:
50
50 merupakan Nilai B
```

Deskripsi:

Input Nilai: Meminta pengguna untuk memasukkan nilai menggunakan `stdin.readLineSync()`.

Validasi Input: Setelah menerima input, program memeriksa apakah input tersebut bukan null dan dapat dikonversi menjadi bilangan bulat menggunakan `int.tryParse()`.

Penentuan Kategori Nilai:

- Jika nilai **lebih besar dari 70**, maka hasilnya adalah `"Nilai A"`.
- Jika nilai berada **dalam nilai 41 hingga 70**, hasilnya adalah `"Nilai B"`.
- Jika nilai berada **dalam nilai 1 hingga 40**, hasilnya adalah `"Nilai C"`.
- Jika nilai **kurang dari atau sama dengan 0**, program akan menginformasikan bahwa `nilai tidak valid`.

Dalam program, inputan ada 2 yaitu `80` dan `50`, maka program akan menampilkan hasil **`"Nilai A"`** untuk **nilai 80**, dan **`"Nilai B"`** untuk **nilai 50**.

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

```dart
import 'dart:io';

// Fungsi utama untuk menjalankan program
void main() {
  // Tugas 2: Menampilkan piramida bintang
  print('Masukkan tinggi piramida: ');
  String? inputTinggi = stdin.readLineSync();

  if (inputTinggi != null && int.tryParse(inputTinggi) != null) {
    int tinggi = int.parse(inputTinggi);

    // Mencetak piramida
    for (int i = 1; i <= tinggi; i++) {
      String line = ''; // Variabel untuk menyimpan baris

      // Menambahkan spasi di depan
      for (int j = tinggi; j > i; j--) {
        line += ' '; // Menambahkan spasi ke string
      }

      // Menambahkan bintang ke baris
      for (int k = 1; k <= (2 * i - 1); k++) {
        line += '*'; // Menambahkan bintang ke string
      }

      print(line); // Mencetak seluruh baris
    }
  } else {
    print('Input tidak valid, harap masukkan bilangan bulat.');
  }
}

```

Output tinggi piramida (4):

```
PS C:\Users\mutge\prak_ppb3> dart bin/guided.dart
Masukkan tinggi piramida:
4
   *
  ***
 *****
*******
```

Deskripsi:

- **Input Tinggi Piramida**: Meminta pengguna untuk memasukkan tinggi piramida menggunakan `stdin.readLineSync()`.
- **Validasi Input:** Setelah menerima input, program memeriksa apakah input tersebut bukan null dan dapat dikonversi menjadi bilangan bulat menggunakan `int.tryParse()`.

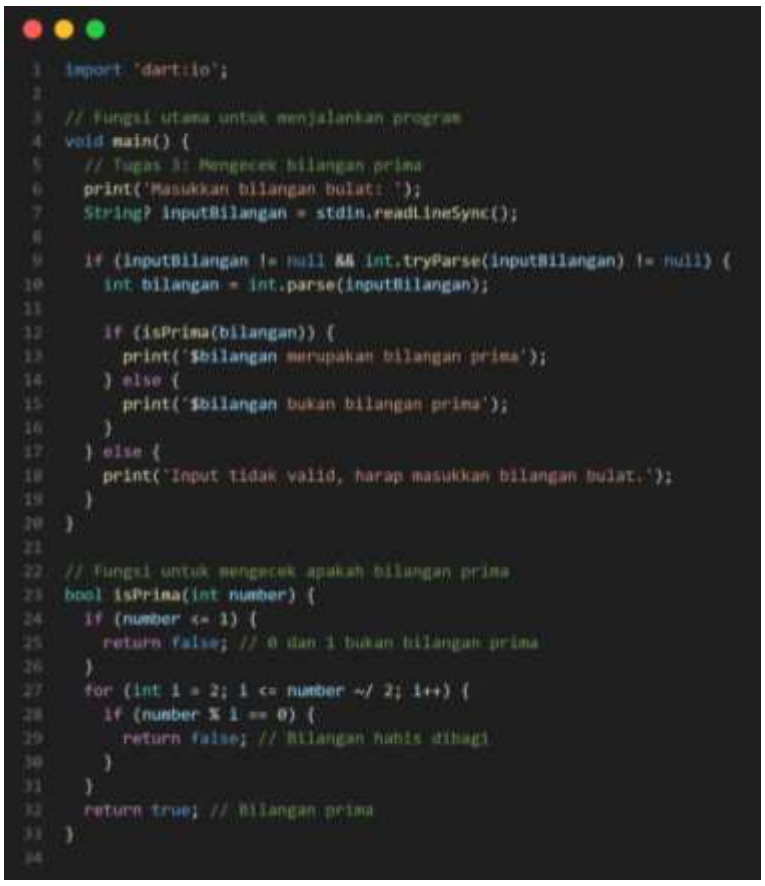
Mencetak Piramida:

- Program menggunakan dua loop bersarang:
- Loop Pertama `(i)`: Mengiterasi dari 1 hingga tinggi piramida yang dimasukkan.
- Loop Kedua `(j)`: Menambahkan spasi di depan setiap baris untuk menyesuaikan bentuk piramida.
- Loop Ketiga `(k)`: Menambahkan bintang ke setiap baris sesuai dengan jumlah yang dihitung menggunakan `rumus 2 * i - 1`, di mana i adalah nomor baris saat ini.

Input banyak baris yang diinputkan pada program di atas adalah 4, maka outputnya pyramid terbuat dengan 4 baris.

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.

Jawaban:



```dart
import 'dart:io';

// Fungsi utama untuk menjalankan program
void main() {
  // Tugas 3: Mengecek bilangan prima
  print('Masukkan bilangan bulat: ');
  String? inputBilangan = stdin.readLineSync();

  if (inputBilangan != null && int.tryParse(inputBilangan) != null) {
    int bilangan = int.parse(inputBilangan);

    if (isPrima(bilangan)) {
      print('$bilangan merupakan bilangan prima');
    } else {
      print('$bilangan bukan bilangan prima');
    }
  } else {
    print('Input tidak valid, harap masukkan bilangan bulat.');
  }
}

// Fungsi untuk mengecek apakah bilangan prima
bool isPrima(int number) {
  if (number <= 1) {
    return false; // 0 dan 1 bukan bilangan prima
  }
  for (int i = 2; i <= number ~/ 2; i++) {
    if (number % i == 0) {
      return false; // Bilangan habis dibagi
    }
  }
  return true; // Bilangan prima
}
```

Output inputan 23:

```
PS C:\Users\mutge\prak_ppb3> dart bin/guided.dart
Masukkan bilangan bulat:
23
23 merupakan bilangan prima
```

Output inputan 12:

```
PS C:\Users\mutge\prak_ppb3> dart bin/guided.dart
Masukkan bilangan bulat:
12
12 bukan bilangan prima
```

Deskripsi:

- **Input Bilangan:** Meminta pengguna untuk memasukkan bilangan bulat menggunakan `stdin.readLineSync()`.
- **Validasi Input**: Dari inputan, program memeriksa apakah input tersebut bukan null dan dapat dikonversi menjadi bilangan bulat menggunakan `int.tryParse()`.

Pemeriksaan Bilangan Prima:

- Program memanggil fungsi `isPrima(bilangan)` untuk menentukan apakah bilangan yang dimasukkan adalah bilangan prima.

Fungsi isPrima:

- Jika bilangan kurang dari atau sama dengan 1, fungsi mengembalikan false (karena 0 dan 1 bukan bilangan prima).
- Menggunakan loop untuk memeriksa apakah bilangan dapat dibagi tanpa sisa oleh angka lain dari 2 hingga setengah dari bilangan tersebut. Jika ada angka yang dapat membagi habis, fungsi mengembalikan false.
- Jika tidak ditemukan pembagi, fungsi mengembalikan true, menunjukkan bahwa bilangan tersebut adalah bilangan prima.

Inputan dalam contoh program adalah 23 dan 12.
23 adalah bilangan prima karena hanya dapat dibagi oleh 1 dan 23.
12 bukan bilangan prima karena memiliki banyak faktor, yaitu 1, 2, 3, 4, 6, dan 12.

Maka output untuk input 23 adalah `“23 merupakan bilangan prima”`, dan input 12 adalah `“12 bukan bilangan prima”`

BAB III
KESIMPULAN DAN SARAN

A. KESIMPULAN

Pada modul ini, telah dipelajari dasar-dasar pemrograman dengan bahasa Dart. Mulai dari penggunaan struktur kontrol seperti if-else dan perulangan yang sangat membantu dalam mengatur alur program. Selain itu, konsep fungsi dipahami sebagai cara efektif untuk membuat kode lebih terstruktur. Penggunaan list juga sangat berguna dalam menyimpan dan mengelola data.

B. REFERENSI

Dart Programming Language Documentation. (n.d.). Retrieved from https://dart.dev/guides

R. (2021). "Programming in Dart." O'Reilly Media.