**BÁO CÁO TUẦN 1**

**Họ và tên:** Nguyễn Kỳ Anh  
**MSSV:** 20225793  
 **GR1**  
**Giáo viên hướng dẫn:** Nguyễn Đức Toàn

**1. Cài đặt Flutter trên VSCode**

Trong tuần đầu tiên,em đã tìm hiểu về cách cài đặt môi trường phát triển Flutter trên Visual Studio Code (VSCode). Các bước thực hiện gồm:

* Tải và cài đặt Flutter SDK từ trang chính thức của Flutter.
* Cấu hình biến môi trường để sử dụng Flutter trên terminal.
* Cài đặt và cấu hình trình giả lập Android hoặc kết nối với thiết bị thật.
* Cài đặt plugin Flutter và Dart trên VSCode.
* Kiểm tra cài đặt bằng lệnh flutter doctor để đảm bảo môi trường hoạt động đúng cách.

**2. Các kiểu khai báo cơ bản trong Dart**

Dart hỗ trợ nhiều kiểu dữ liệu cơ bản, bao gồm:

* int: Số nguyên.
* double: Số thực.
* bool: Kiểu luận lý với hai giá trị true và false.
* String: Chuỗi ký tự.
* var: Biến có thể nhận bất kỳ kiểu dữ liệu nào.
* dynamic: Tương tự như var, nhưng có thể thay đổi kiểu dữ liệu trong quá trình chạy chương trình.

**3. Các kiểu toán tử trong Dart**

Dart cung cấp nhiều loại toán tử, bao gồm:

* Toán tử số học (+, -, \*, /, %)
* Toán tử quan hệ (==, !=, >, <, >=, <=)
* Toán tử logic (&&, ||, !)
* Toán tử gán (=, +=, -=, \*=, /=)
* Toán tử điều kiện (condition ? expr1 : expr2)

**4. Câu lệnh điều kiện và vòng lặp**

* Câu lệnh điều kiện: if, else if, else
* Vòng lặp:
  + for: Lặp với số lần xác định trước.
  + while: Lặp khi điều kiện còn đúng.
  + do-while: Chạy ít nhất một lần trước khi kiểm tra điều kiện.

**5. Hàm cơ bản trong Dart**

Hàm trong Dart có thể có kiểu trả về hoặc không trả về giá trị.

* Hàm có kiểu trả về: int, double, bool, String.
* Hàm không có kiểu trả về: void.
* Hàm có thể có tham số và giá trị mặc định.

Ví dụ:

int sum(int a, int b) {

return a + b;

}

**6. Enum và Iterable trong Dart**

* enum: Định nghĩa danh sách các giá trị có thể có của một kiểu dữ liệu.
* Iterable: Làm việc với danh sách dữ liệu có thể lặp.

**7. Cấu trúc dữ liệu List trong Dart**

* List: Danh sách có thể chứa nhiều phần tử với các phương thức như add(), remove(), length.

Ví dụ:

List<int> numbers = [1, 2, 3, 4, 5];

numbers.add(6);

numbers.remove(3);

print(numbers);

**8. Map trong Dart**

* Map: Dữ liệu dạng key-value.

Ví dụ:

Map<String, int> ages = {

"An": 25,

"Bình": 30

};

**9. Set trong Dart**

* Set: Tập hợp không chứa phần tử trùng lặp.

Ví dụ:

Set<int> uniqueNumbers = {1, 2, 3, 3, 4};

print(uniqueNumbers); // {1, 2, 3, 4}

**10. Truyền tham số cho hàm**

Dart hỗ trợ truyền tham số theo 3 cách:

* Tham số vị trí (positional parameters)
* Tham số đặt tên (named parameters)
* Tham số mặc định (default parameters)

**11. Cơ bản về Class trong Dart**

Lập trình hướng đối tượng trong Dart dựa trên class và object. Một class có thể có thuộc tính và phương thức.

Ví dụ:

class Person {

String name;

int age;

Person(this.name, this.age);

}

**12. Get và Set trong Dart**

* getter và setter giúp kiểm soát truy cập thuộc tính của class.

Ví dụ:

class Car {

String \_model;

Car(this.\_model);

String get model => \_model;

set model(String newModel) => \_model = newModel;

}

**13. Initializer List**

* Dùng để khởi tạo giá trị trước khi constructor chạy.

Ví dụ:

class Point {

final int x;

final int y;

Point(this.x, this.y);

}

**14. Named Constructors**

Dart hỗ trợ constructors có tên giúp tạo đối tượng linh hoạt hơn.

Ví dụ:

class Animal {

String name;

Animal(this.name);

Animal.fromJson(Map<String, String> json) : name = json['name'] ?? '';

}