PRACTICAL 1

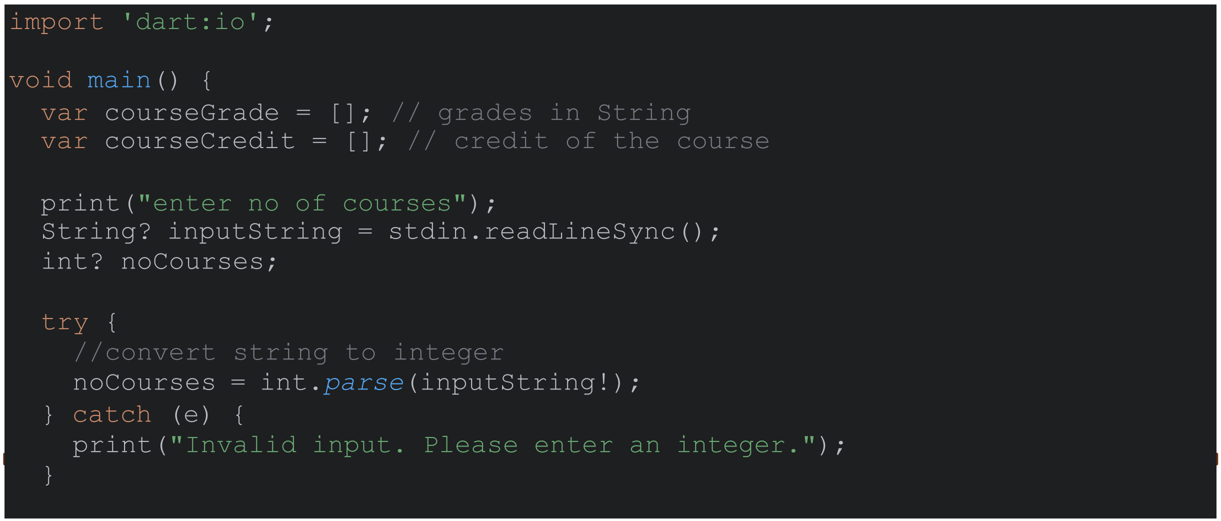
AIM:

Write a program using Dart to calculate student results from grade point average which includes the concept of data types, functions, control flow, error handling, and dart package.

THEORY:

- **Data types**
  - String
  - Var
  - Let
  - Int
  - Double
- **Functions**
  - Main():Entry point of the program controls all over execution
  - courseDetails(): Takes the grade scored in the course and the credits of the course from the user
  - convertGrade(): converts the grade into the double datatype
  - result(): gives the allover gpa as output
- **Control Flow**
  - while loop: Iterates for user input until no of courses are entered.
  - if-else statement: Handles potential invalid input formats.
  - Switch statement: Handles the conversion of the grade into specific double
- **Error Handling**
  - try-catch block: Encloses code that might throw exceptions (e.g., FormatException for invalid input).
  - throw statement: Used to raise a FormatException for invalid input.
- **Dart Package**
  - Dart.io : It is a core Dart library that provides fundamental input/output (VO) functionality for Dart applications.

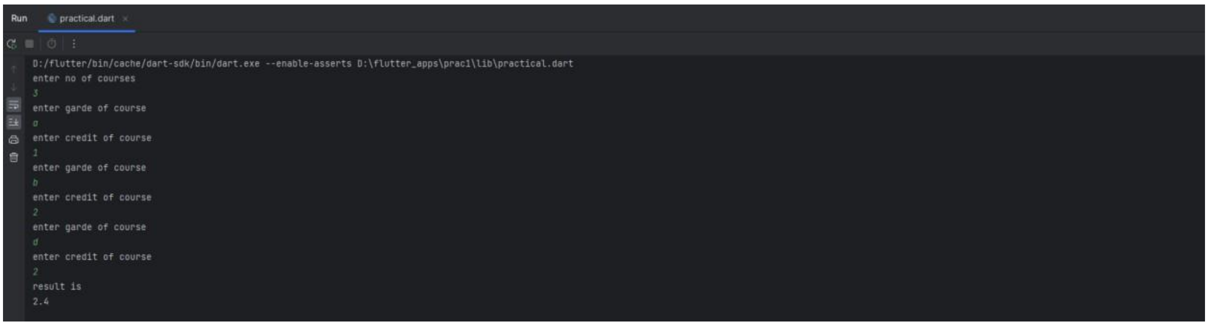
CODE:



```
import 'dart:io';

void main() {
  var courseGrade = []; // grades in String
  var courseCredit = []; // credit of the course

  print("enter no of courses");
  String? inputString = stdin.readLineSync();
  int? noCourses;

  try {
    //convert string to integer
    noCourses = int.parse(inputString!);
  } catch (e) {
    print("Invalid input. Please enter an integer.");
  }
  courseDetails(courseGrade, courseCredit, noCourses);

  result(courseGrade , courseCredit , noCourses);
}

//takes input for the grades of the courses and their credits
courseDetails(var grades, var credits, var noCourses) {
  var i = 0;
  while (i < noCourses!) {
    print("enter garde of course ");
    var newGrade = stdin.readLineSync()!.toUpperCase();
    grades.add(convertGrade(newGrade));
    print("enter credit of course ");
    var credit = stdin.readLineSync();
    var newCredit = int.parse(credit!);
    credits.add(newCredit);
    i++;
  }
}

//convert grades into the numbers
convertGrade(var grade) {
  switch (grade) {
    case 'A':
      return 4.0;
    case 'B':
      return 3.0;
    case 'C':
      return 2.0;
    case 'D':
      return 1.0;
    case 'F':
      return 0.0;
    default:
      throw Exception('Invalid grade');
  }
}

//converting grades into gpa
result(var grades , var credits , var n){
  var gpa = 0.0;
  var totCredits = 0;
  for(int i=0 ; i<n ; i++){
    totCredits += (credits[i] as int);
    gpa += (grades[i] as double)*(credits[i] as int);
  }
  gpa = gpa/totCredits;
  print( "result is ");
  print(gpa);
}
```

OUTPUT:

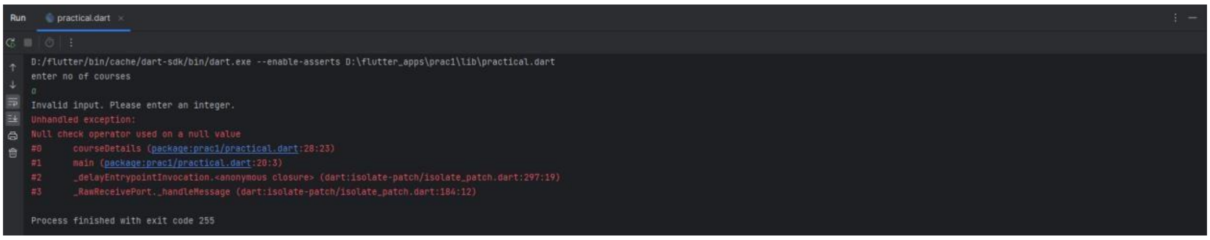
Here, we enter all values correctly and we get the GPA as the answer accordingly.



```
D:/flutter/bin/cache/dart-sdk/bin/dart.exe --enable-asserts D:\flutter_apps\prac1\lib\practical.dart
enter no of courses
3
enter garde of course
a
enter credit of course
1
enter garde of course
b
enter credit of course
2
enter garde of course
d
enter credit of course
2
result is
2.4
```

Figure 1 entering the appropriate values

Now, we enter the values which are inappropriate and the program shows the error.



```
D:/flutter/bin/cache/dart-sdk/bin/dart.exe --enable-asserts D:\flutter_apps\prac1\lib\practical.dart
enter no of courses
a
Invalid input. Please enter an integer.
Unhandled exception:
Null check operator used on a null value
#0      courseDetails (package:prac1/practical.dart:28:23)
#1      main (package:prac1/practical.dart:20:3)
#2      _delayEntrypointInvocation.<anonymous closure> (dart:isolate-patch/isolate_patch.dart:297:19)
#3      _RawReceivePort._handleMessage (dart:isolate-patch/isolate_patch.dart:184:12)

Process finished with exit code 255
```

Figure 2 entering the inappropriate values

Latest Applications:

For making an app with UI/UX with functionalities of calculating CGPA/SGPA.

Learning Outcome:

- User input and validation: Gathering input from the user and validating its correctness to ensure data integrity.
- Calculations and result mapping: Performing numerical calculations (GPA calculation) and using data structures to associate values with meaningful results.
- Code readability and maintainability: Writing clear, well-structured code with meaningful variable names and comments for better understanding and future modifications.