Mobile Applications and Web Development - IS4904(Practical)		
Student Name:	**Student ID:**	**Section:**
Assignment (Exploring the Health Informatics Stack)		
Date: 5th March 2024	**Max Points:**	

Submission Guidelines:
Submit the essay and Dart program files via the designated submission platform.
Ensure code is well-commented and follows Dart style conventions.

Question 1: Introduction to Null Safety in Dart

Objective:

To reinforce understanding of the concept of null safety in Dart and its significance in writing reliable code.

Tasks:

1. Write a short essay explaining the importance of null safety in Dart programming(two lines max).
2. Create a Dart program that demonstrates the difference between nullable and non-nullable variables. Use both nullable and non-nullable variables in your program and explain how null safety affects their usage.

Question 2: Handling Null Values in Dart

Objective:

To practice working with nullable types and null-aware operators in Dart to effectively handle null values.

Tasks:

1. Write a Dart program that demonstrates the use of null-aware operators (??, ?., ??=) to handle null values. Include examples of each operator in your program and explain their purpose.
2. Write a Dart program and develop a function that takes a nullable integer parameter and returns a non-nullable integer. Handle null values appropriately within the function and explain your approach.
3. Create a Dart program that reads user input for a person's age and prints a customized message based on the input. Ensure the program handles null values and invalid input gracefully.

Question 3: Handling Null Values in Dart

Objective:

To delve deeper into advanced techniques for handling null values in Dart and practice implementing null-safe code patterns.

Tasks:

1. Write a Dart program that utilizes the **late** keyword and null assertion operator (!) to handle delayed initialization of non-nullable variables.

```
class Course {
  // Course attributes
  String courseId;
  String courseName;
  String courseDepartment;
  String reference;
  String coordinator;
  // Default constructor
  Course() {
    // Initialize attributes with default values
    courseId = '';
    courseName = '';
    courseDepartment = '';
    reference = '';
    coordinator = '';
  }
  // Constructor with parameters
  Course.withValues(
      this.courseId, this.courseName, this.courseDepartment, this.reference, this.coordinator);
```

```
// Setter methods
void setCourseId(String id) {
  courseId = id;
}
void setCourseName(String name) {
  courseName = name;
}

void setCourseDepartment(String department) {
  courseDepartment = department;
}

void setReference(String ref) {
  reference = ref;
}

void setCoordinator(String coord) {
  coordinator = coord;
}

// Getter methods
String getCourseId() {
  return courseId;
}

String getCourseName() {
  return courseName;
}

String getCourseDepartment() {
  return courseDepartment;
}

String getReference() {
  return reference;
}
```

```
  String getCoordinator() {
    return coordinator;
  }
}

void main() {
  // Create an empty course object using the default constructor
  Course emptyCourse = Course();

  // Create a course object with default values using the constructor with parameters
  Course defaultCourse = Course.withValues(
      'C001', 'Introduction to Programming', 'Computer Science', 'REF001', 'John Doe');

  // Set course information using setter methods
  emptyCourse.setCourseId('C002');
  emptyCourse.setCourseName('Data Structures');
  emptyCourse.setCourseDepartment('Computer Science');
  emptyCourse.setReference('REF002');
  emptyCourse.setCoordinator('Jane Smith');

  // Display course data using getter methods
  print('Course ID: ${defaultCourse.getCourseId()}');
  print('Course Name: ${defaultCourse.getCourseName()}');
  print('Course Department: ${defaultCourse.getCourseDepartment()}');
  print('Reference: ${defaultCourse.getReference()}');
  print('Coordinator: ${defaultCourse.getCoordinator()}');

  print('---');

  print('Course ID: ${emptyCourse.getCourseId()}');
  print('Course Name: ${emptyCourse.getCourseName()}');
  print('Course Department: ${emptyCourse.getCourseDepartment()}');
  print('Reference: ${emptyCourse.getReference()}');
  print('Coordinator: ${emptyCourse.getCoordinator()}');
}
```