UNIVERSITY OF RWANDA

COLLEGE OF SCIENCE AND TECHNOLOGY

SCHOOL OF ICT

COMPUTER ENGINEERING

YEAR 3

GAKO CAMPUS

ON 2nd FEB 2026

GROUP 1 MEMBERS:

1.DAVID RUBIMBURA 221014566

2.REOBOTH IRADUKUNDA 223026686

3.VALENTIN MUHIRE 223026676

MODULE: MOBILE APPLICATION SYSTEM AND DESIGN

TOPIC: **SCREENSHOTS OF THE OUTPUTS OF EACH QUESTION**

SCREENSHOTS OF THE OUTPUTS OF EACH QUESTION

Q1: Write a Dart function named `welcomeMessage` that prints a welcome message for the school system.

```
==================================================
QUESTION 1: Welcome Message Function
==================================================

==========================================
    WELCOME TO OUR SCHOOL SYSTEM
==========================================
We are delighted to have you here.
Our mission is to provide quality education
and foster a positive learning environment.
==========================================
```

Q2: Write a function named `createStudent` that uses named parameters (`name` and `age`) and prints the student details.

```
==================================================
QUESTION 2: createStudent with Named Parameters
==================================================

--- Student Record ---
Name: Alice Johnson
Age: 16
Student ID: STU-ALI16
Status: Enrolled

--- Student Record ---
Name: Bob Smith
Age: 14
Student ID: STU-BOB14
Status: Enrolled
```

Q3: Write a function named `createTeacher` with one required parameter `name` and one optional parameter `subject`. If subject is not provided, print 'Subject not assigned'.

```
==================================================
QUESTION 3: createTeacher with Optional Parameter
==================================================

--- Teacher Record ---
Name: Mr. Davis
Subject: Mathematics
Teacher ID: TCH-MD
Status: Active

--- Teacher Record ---
Name: Ms. Rodriguez
Subject: Subject not assigned
Teacher ID: TCH-MR
Status: Active
```

Q4: Create a class named `Student` with `name` and `age`, and a constructor to initialize these values.

```
==================================================
QUESTION 4: Student Class with Constructor
==================================================
Student created: John Smith, 17 years old
```

Q5: Create an object of `Student` and print the student's name and age.

```
==================================================
QUESTION 5: Create and Print Student Object
==================================================
Name: Emily Wilson
Age: 16
```

Q6: Create a class `Person` with a variable `name` and a function `introduce()` that prints the name.

```
==================================================
QUESTION 6: Person Class with introduce() Method
==================================================
Hello, my name is Michael Chen.
```

Q7: Make `Student` inherit from `Person` and call `introduce()` from a `Student` object.

```
==================================================
QUESTION 7: Inheritance - Student extends Person
==================================================
Hello, my name is Sarah Johnson.
```

Q8: Create an abstract class `Registrable` with a function `registerCourse(String courseName)`.

```
==================================================
QUESTION 8: Abstract Class Registrable
==================================================
abstract class Registrable {
  void registerCourse(String courseName);
}
```

Q9: Make `Student` implement `Registrable` and implement `registerCourse` to print the student name and course.

```
==================================================
QUESTION 9: Student implements Registrable Interface
==================================================
David Lee registered for: Mathematics
Notification: David Lee registered for Mathematics
David Lee registered for: Computer Science
Notification: David Lee registered for Computer Science
David Lee registered for: Physics
Notification: David Lee registered for Physics
Courses: Mathematics, Computer Science, Physics
```

Q10: Create a mixin `AttendanceMixin` that stores an attendance counter and has a function `markAttendance()` to increase attendance.

```
==================================================
QUESTION 10: AttendanceMixin
==================================================
--- Testing AttendanceMixin with temporary test class ---
Attendance marked. Total: 1
Attendance marked. Total: 2
Total attendance: 2
```

Q11: Apply `AttendanceMixin` to `Student`. Mark attendance 3 times and print the attendance.

```
==================================================
QUESTION 11: Student with AttendanceMixin
==================================================
Student: Olivia Brown
Attendance marked. Total: 1
Attendance marked. Total: 2
Attendance marked. Total: 3
Olivia Brown total attendance: 3 days

--------------------------------------------------
```

Q12: Create a List storing multiple `Student` objects. Add 3 students.

```
==================================================
QUESTION 12: List of Student Objects
==================================================
Students in list: 3
1. Emma Wilson, Age: 15
2. James Miller, Age: 16
3. Sophia Davis, Age: 17
```

Q13: Create a Map where the key is student ID and value is a `Student`. Print all student names.

```
==================================================
QUESTION 13: Map with Student Objects
==================================================
Student Map Entries: 3
STU-ALE16 -> Alex Turner
STU-MAY15 -> Maya Patel
STU-RYA17 -> Ryan Kim
```

Q14: Use an anonymous function to print all student names from the list.

```
==================================================
QUESTION 14: Anonymous Function
==================================================
Student names:
  - Emma Wilson
  - James Miller
  - Sophia Davis
```

Q15: Write an arrow function that takes a student name and prints a greeting message.

```
QUESTION 15: Arrow Function
================================================
Welcome to class, Michael!
Welcome to class, Sarah!
Welcome to class, David!

================================================
```

Q16: Write an async function `loadStudents()` that waits 2 seconds and returns the list of students.

```
================================================
QUESTION 16: Async Function loadStudents()
================================================
Loading students from database...
First student: Emma Johnson
Total loaded: 5 students
```

Q17: In main(), call `loadStudents()`, use await, and print the number of students loaded.

```
================================================
QUESTION 17: Using await with loadStudents()
================================================
Loading students from database...
Total students loaded: 5
  - Emma Johnson (Age: 16)
  - Noah Williams (Age: 17)
  - Olivia Brown (Age: 15)
  - Liam Davis (Age: 16)
  - Ava Miller (Age: 17)
```

Q18: Explain in your own words: why mixins are useful, and how inheritance and mixins are different.

WHY MIXINS ARE USEFUL:

1. Code Reuse Across Hierarchies: Share functionality between unrelated classes

2. Avoid Diamond Problem: No ambiguity when same method exists in multiple parents

3. Flexible Composition: Add behaviors like building blocks, not rigid hierarchies

4. Single Responsibility: Keep classes focused, mixins handle cross-cutting concerns

5. Testability: Mixins can be tested independently

DIFFERENCES BETWEEN INHERITANCE AND MIXINS:

INHERITANCE	MIXINS
"is-a" relationship	"has-a" capability
Single parent only	Multiple mixins
Deep class hierarchy	Flat composition
Tight coupling	Loose coupling
Compile-time resolution	Linearized order
Hard to change later	Easy to add/remove

PRACTICAL EXAMPLE IN OUR CODE:

Inheritance: Student IS-A Person (gets basic identity)

Mixin 1: Student HAS Attendance tracking capability

Mixin 2: Student HAS Notification capability

Result: Flexible, reusable behaviors without complex hierarchy

Q19: Create a new mixin `NotificationMixin` that prints a message when a student is registered to a course. Apply it to `Student`.

```
QUESTION 19: NotificationMixin
==============================================
Daniel White registered for: Advanced Mathematics
Notification: Daniel White registered for Advanced Mathematics
Daniel White registered for: Data Structures
Notification: Daniel White registered for Data Structures
```

Q20: Write a short paragraph explaining how learning Dart helps you understand Flutter.

Dart is the foundation of Flutter development. By learning Dart, we gain:

1. WIDGET UNDERSTANDING:

- Flutter widgets are Dart classes
- State management uses Dart's reactive programming

2. ASYNC OPERATIONS:

- Flutter apps heavily use async/await for API calls

- Future and Stream are core Dart concepts

3. OBJECT-ORIENTED STRUCTURE:

- Flutter's widget tree is built using Dart OOP
- Inheritance and mixins create reusable UI components

4. TYPE SAFETY:

- Dart's strong typing prevents runtime errors in Flutter
- Null safety ensures robust mobile applications

5. HOT RELOAD:

- Dart's JIT compiler enables Flutter's fast development cycle
- Quick iteration is possible because of Dart's architecture

Learning Dart first makes Flutter development intuitive because every Flutter app is essentially a well-structured Dart program with a UI layer.