**Lab Manual  
Mobile Application Development**

**Prepared by:**

**Dr. Osman Khalid**

**Associate Professor**

**LAST UPDATED: 29 Dec 2024**

Table of Contents

[DART (CLO-1) 1](#_Toc186404086)

[FLUTTER WIDGETS (CLO-2) 68](#_Toc186404087)

[NAVIGATION IN FLUTTER (CLO-2) 165](#_Toc186404088)

[DATABASE CONNECTIVITY IN FLUTTER (CLO-3) 211](#_Toc186404089)

[FLUTTER ADVANCED TOPICS (CLO-3) 223](#_Toc186404090)

# DART (CLO-1)

**Basics**

**Question P1:**

\*Print a Welcome Message\*

Write a program that prints "Welcome to Programming!" to the console.

|  |
| --- |
| void main() {  print("Welcome to Programming!");  } |

**Question A1:**

Write a program that takes the command line arguments of types string, integer and decimal value. In the case of a string, its length should be displayed, for an integer value, it should be multiplied by 100, and for a decimal value, we need to take its power of 3.

|  |
| --- |
| /\*  Question A1:  Write a program that takes as input the command line arguments of types string, integer and decimal value. In case of a string, its length should be displayed, for an integer value, it should be multiplied by 100, and for a decimal value, we need to take its power of 3.  \*/  import 'dart:math';  void main(List <String>args) {  for(var item in args) {  if(int.tryParse(item) != null) {  var num = int.parse(item);  print(num \* 100);  }  else if(double.tryParse(item) != null) {  var num = double.parse(item);  print( pow(num, 3));  }  else {  print(item.length);  }    }  } |

**Question P2:**

\*Simple Arithmetic Operations\*

Write a program that takes two numbers as input and outputs their sum, difference, product, and quotient.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking first number as input  print('Enter the first number: ');  double num1 = double.parse(stdin.readLineSync()!);  // Taking second number as input  print('Enter the second number: ');  double num2 = double.parse(stdin.readLineSync()!);  // Calculating sum, difference, product, and quotient  double sum = num1 + num2;  double difference = num1 - num2;  double product = num1 \* num2;  double quotient = num1 / num2;  // Outputting the results  print('Sum: $sum');  print('Difference: $difference');  print('Product: $product');  print('Quotient: $quotient');  } |

**Question P3:**

\*Even or Odd\*

Write a program that checks if a given number is even or odd.

|  |
| --- |
| void main() {  // Taking a number as input  int number = 23;  // Checking if the number is even or odd  if (number % 2 == 0) {  print('$number is even.');  } else {  print('$number is odd.');  }  } |

**Question P4:**

\*Find the Largest of Three Numbers\*

Write a program that takes three numbers as input and outputs the largest number.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking three numbers as input  print('Enter the first number: ');  double num1 = double.parse(stdin.readLineSync()!);  print('Enter the second number: ');  double num2 = double.parse(stdin.readLineSync()!);  print('Enter the third number: ');  double num3 = double.parse(stdin.readLineSync()!);  // Finding the largest number  double largest = num1;  if (num2 > largest) {  largest = num2;  }  if (num3 > largest) {  largest = num3;  }  // Outputting the largest number  print('The largest number is: $largest');  } |

**Question P6:**

\*Simple Interest Calculation\*

Write a program to calculate the simple interest using the formula SI = (P \* R \* T) / 100, where P is the principal, R is the rate of interest, and T is the time period.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking principal amount as input  print('Enter the principal amount (P): ');  double principal = double.parse(stdin.readLineSync()!);  // Taking rate of interest as input  print('Enter the rate of interest (R): ');  double rate = double.parse(stdin.readLineSync()!);  // Taking time period as input  print('Enter the time period in years (T): ');  double time = double.parse(stdin.readLineSync()!);  // Calculating simple interest  double simpleInterest = (principal \* rate \* time) / 100;  // Outputting the simple interest  print('The Simple Interest is: \$${simpleInterest.toStringAsFixed(2)}');  } |

**Question P7:**

\*Reverse a String\*

Write a program to reverse a string entered by the user.

|  |
| --- |
| void main() {  String str = "PAKISTAN";  print(str.split('').reversed.join());  } |

**Question P8:**

\*Check for Leap Year\*

Write a program that checks if a given year is a leap year or not.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking a year as input  print('Enter a year: ');  int year = int.parse(stdin.readLineSync()!);  // Checking if the year is a leap year  bool isLeapYear = (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);  // Outputting the result  if (isLeapYear) {  print('$year is a leap year.');  } else {  print('$year is not a leap year.');  }  } |

**Question P9:**

\*Multiplication Table\*

Write a program that generates the multiplication table for a given number.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking a number as input  print('Enter a number: ');  int number = int.parse(stdin.readLineSync()!);  // Generating the multiplication table  print('Multiplication table for $number:');  for (int i = 1; i <= 10; i++) {  print('$number x $i = ${number \* i}');  }  } |

**Question P10:**

\*Count Digits in a Number\*

Write a program that counts the number of digits in an integer entered by the user.

|  |
| --- |
| import 'dart:io';  void main() {  // Taking an integer as input  print('Enter an integer: ');  int number = int.parse(stdin.readLineSync()!);  // Counting the number of digits  int digitCount = number.abs().toString().length;  // Outputting the number of digits  print('The number of digits in $number is: $digitCount');  } |

**Records**

**Question P11:**

\*\*Task\*\*: Write a Dart program that defines a record type to store the name and age of a person. Create three records for different people and print out their details.

- Define a record `(String, int)` to hold the `name` and `age`.

- Access the fields of the record and print them.

\*\*Example Output:\*\*

```

Name: Alice, Age: 25

Name: Bob, Age: 30

Name: Charlie, Age: 22

```

|  |
| --- |
| void main() {  // Defining a record type to hold name and age  var person1 = ('Alice', 30);  var person2 = ('Bob', 25);  var person3 = ('Charlie', 35);  // Accessing and printing the fields of the records  print('Person 1: Name = ${person1.$1}, Age = ${person1.$2}');  print('Person 2: Name = ${person2.$1}, Age = ${person2.$2}');  print('Person 3: Name = ${person3.$1}, Age = ${person3.$2}');  } |

**Question P12:**

\*\*Working with Lists of Records\*\*

\*\*Task\*\*: Define a list of records to store information about students. Each student record should contain the student’s name, age, and grade. Write a Dart program that prints the details of all students in the list.

- Define a record `(String, int, double)` for `name`, `age`, and `grade`.

- Create a list of at least 5 student records.

- Loop through the list and print each student's details.

\*\*Example Output:\*\*

```

Name: John, Age: 18, Grade: 85.5

Name: Lisa, Age: 19, Grade: 90.0

...

```

|  |
| --- |
| void main() {  // Defining a record type to hold name, age, and grade  var student1 = ('Alice', 20, 85.5);  var student2 = ('Bob', 22, 90.0);  var student3 = ('Charlie', 19, 78.0);  var student4 = ('Diana', 21, 92.5);  var student5 = ('Eve', 20, 88.0);  // Creating a list of student records  var students = [student1, student2, student3, student4, student5];  // Looping through the list and printing each student's details  for (var student in students) {  print('Name: ${student.$1}, Age: ${student.$2}, Grade: ${student.$3}');  }  } |

**Question P13:**

\*\*Sorting Records by Age\*\*

\*\*Task\*\*: Create a list of person records with fields `name` and `age`. Write a program that sorts the list by age in ascending order and prints the sorted list.

- Define a record `(String, int)` for the person's name and age.

- Sort the list by age using Dart's list sorting mechanism.

- Print the sorted list of people.

\*\*Example Output (before sorting):\*\*

```

Name: Alice, Age: 25

Name: Bob, Age: 30

Name: Charlie, Age: 22

```

\*\*Example Output (after sorting):\*\*

```

Name: Charlie, Age: 22

Name: Alice, Age: 25

Name: Bob, Age: 30

```

|  |
| --- |
| //list by age in ascending order and prints the sorted list.  void main() {  var records = [  (name: "Alice", age: 25),  (name: "Bob", age: 30),  (name: "Charlie", age: 22)  ];  records.sort((a, b) => a.age.compareTo(b.age));  print(records);  } |

**Question P14:**

\*\*Filtering Records\*\*

\*\*Task\*\*: Define a list of student records where each record contains the student's `name`, `age`, and `grade`. Write a Dart program that filters out and prints only the students with grades greater than 75.

- Define a record `(String, int, double)` for the student details.

- Filter the list based on the `grade` field.

- Print the details of students who meet the criteria.

\*\*Example Output:\*\*

```

Name: John, Age: 18, Grade: 85.5

Name: Lisa, Age: 19, Grade: 90.0

|  |
| --- |
| void main() {  // Defining a record type to hold name, age, and grade  var student1 = ('Alice', 20, 85.5);  var student2 = ('Bob', 22, 70.0);  var student3 = ('Charlie', 19, 78.0);  var student4 = ('Diana', 21, 92.5);  var student5 = ('Eve', 20, 65.0);  // Creating a list of student records  var students = [student1, student2, student3, student4, student5];  // Filtering the list based on the grade field  var filteredStudents = students.where((student) => student.$3 > 75).toList();  // Printing the details of students who meet the criteria  for (var student in filteredStudents) {  print('Name: ${student.$1}, Age: ${student.$2}, Grade: ${student.$3}');  }  } |

**Question P15:**

\*\*Updating Records\*\*

\*\*Task\*\*: Write a Dart program that updates a list of product records. Each product has a `name` and `price`. The program should increase the price of all products by 10%. After updating, print the updated product list.

- Define a record `(String, double)` for product name and price.

- Use a loop or map to update the price of each product.

- Print the updated list of products.

\*\*Example Output:\*\*

```

Before Update:

Name: Laptop, Price: 1000.0

Name: Phone, Price: 600.0

After Update:

Name: Laptop, Price: 1100.0

Name: Phone, Price: 660.0

|  |
| --- |
| void main() {  // Defining a record type to hold product name and price  var product1 = ('Laptop', 1000.0);  var product2 = ('Smartphone', 500.0);  var product3 = ('Tablet', 300.0);  var product4 = ('Headphones', 100.0);  var product5 = ('Smartwatch', 200.0);  // Creating a list of product records  var products = [product1, product2, product3, product4, product5];  // Updating the price of each product by 10%  var updatedProducts = products.map((product) {  var updatedPrice = product.$2 \* 1.10;  return (product.$1, updatedPrice);  }).toList();  // Printing the updated list of products  for (var product in updatedProducts) {  print('Product: ${product.$1}, Price: \$${product.$2.toStringAsFixed(2)}');  }  } |

**Lists**

**Question P16:**

\*\*Task\*\*: Write a Dart program that creates a list of integers. Perform the following operations:

- Add an element to the list.

- Remove an element from the list.

- Update an element at a specific index.

- Print the final list.

\*\*Example Input:\*\*

```

Initial List: [10, 20, 30, 40, 50]

Add 60

Remove 20

Update 30 to 35

```

\*\*Example Output:\*\*

```

Final List: [10, 35, 40, 50, 60]

```

|  |
| --- |
| void main() {  // Creating a list of integers  var numbers = [10, 20, 30, 40, 50];  // Adding an element to the list  numbers.add(60);  print('After adding 60: $numbers');  // Removing an element from the list  numbers.remove(30);  print('After removing 30: $numbers');  // Updating an element at a specific index  numbers[2] = 100; // Updating the element at index 2  print('After updating index 2 to 100: $numbers');  // Printing the final list  print('Final list: $numbers');  } |

**Question P17:**

\*\*Task\*\*: Write a Dart program that searches for a specific element in a list of strings. If the element is found, print its index. If not, print a message saying the element is not found.

- Create a list of string elements.

- Take input (or hardcode) the element to search.

- Use a loop or list method to find the index.

\*\*Example Input:\*\*

```

List: ['apple', 'banana', 'cherry', 'date']

Search for: 'cherry'

```

\*\*Example Output:\*\*

```

'cherry' found at index 2.

```

|  |
| --- |
| void main() {  // Create a list of string elements  List<String> elements = ['apple', 'banana', 'cherry', 'date', 'elderberry'];  // Element to search for  String searchElement = 'cherry';  // Find the index of the element  int index = elements.indexOf(searchElement);  // Check if the element is found  if (index != -1) {  print('Element found at index: $index');  } else {  print('Element not found');  }  } |

**Question P18:**

\*\*Task\*\*: Write a Dart program that sorts a list of integers in both ascending and descending order. Print the list after each sorting operation.

- Create a list of integers.

- Sort the list in ascending order and print it.

- Sort the list in descending order and print it.

\*\*Example Input:\*\*

```

List: [34, 12, 56, 9, 45]

```

\*\*Example Output:\*\*

```

Ascending Order: [9, 12, 34, 45, 56]

Descending Order: [56, 45, 34, 12, 9]

```

|  |
| --- |
| void main() {  // Create a list of integers  List<int> numbers = [34, 7, 23, 32, 5, 62];  // Sort the list in ascending order  numbers.sort();  print('List in ascending order: $numbers');  // Sort the list in descending order  numbers.sort((a, b) => b.compareTo(a));  print('List in descending order: $numbers');  } |

**Question P19:**

\*\*Task\*\*: Write a Dart program that filters out even and odd numbers from a list of integers. Create two new lists: one containing only even numbers and the other only odd numbers. Print both lists.

- Create a list of integers.

- Use the `where` or a loop to filter out even and odd numbers.

- Print the lists of even and odd numbers.

\*\*Example Input:\*\*

```

List: [11, 22, 33, 44, 55, 66, 77, 88]

```

\*\*Example Output:\*\*

```

Even Numbers: [22, 44, 66, 88]

Odd Numbers: [11, 33, 55, 77]

```

|  |
| --- |
| void main() {  // Create a list of integers  List<int> numbers = [10, 15, 20, 25, 30, 35, 40];  // Filter out even numbers  List<int> evenNumbers = numbers.where((number) => number.isEven).toList();  // Filter out odd numbers  List<int> oddNumbers = numbers.where((number) => number.isOdd).toList();  // Print the lists of even and odd numbers  print('Even numbers: $evenNumbers');  print('Odd numbers: $oddNumbers');  } |

**Question P20:**

\*\*Task\*\*: Write a Dart program that merges two lists of integers into one. After merging, remove any duplicate elements from the list and print the final list.

- Create two lists of integers.

- Merge the lists.

- Use a method like `.toSet()` to remove duplicates.

- Convert the result back to a list and print it.

\*\*Example Input:\*\*

```

List 1: [1, 2, 3, 4, 5]

List 2: [3, 4, 5, 6, 7]

```

\*\*Example Output:\*\*

```

Merged List: [1, 2, 3, 4, 5, 6, 7]

```

These tasks introduce key list operations in Dart such as addition, removal, updating elements, sorting, searching, filtering, and merging lists, which are essential concepts for students learning about lists.

|  |
| --- |
| void main() {  // Create two lists of integers  List<int> list1 = [1, 2, 3, 4, 5];  List<int> list2 = [4, 5, 6, 7, 8];  // Merge the lists  List<int> mergedList = [...list1, ...list2];  // Remove duplicates by converting to a set and back to a list  List<int> uniqueList = mergedList.toSet().toList();  // Print the final list  print('Merged list without duplicates: $uniqueList');  } |

**Sets**

**Question P21:**

\*\*Task\*\*: Write a Dart program that performs the following operations on a set of integers:

- Add elements to the set.

- Remove an element from the set.

- Check if a specific element exists in the set.

- Print the final set.

\*\*Example Input:\*\*

```

Initial Set: {10, 20, 30, 40}

Add: 50

Remove: 20

Check if 30 exists

```

\*\*Example Output:\*\*

```

Set after adding 50: {10, 20, 30, 40, 50}

Set after removing 20: {10, 30, 40, 50}

30 exists in the set.

```

|  |
| --- |
| void main() {  // Create a set of integers  Set<int> numbers = {1, 2, 3, 4, 5};  // Add elements to the set  numbers.add(6);  numbers.add(7);  // Remove an element from the set  numbers.remove(3);  // Check if a specific element exists in the set  int elementToCheck = 4;  bool exists = numbers.contains(elementToCheck);  print('Does the set contain $elementToCheck? $exists');  // Print the final set  print('Final set: $numbers');  } |

**Question P22:**

\*\*Task\*\*: Write a Dart program that takes two sets of integers and finds the union of both sets. The union should contain all unique elements from both sets.

- Create two sets of integers.

- Use the `.union()` method to combine them.

- Print the result.

\*\*Example Input:\*\*

```

Set 1: {1, 2, 3, 4}

Set 2: {3, 4, 5, 6}

```

\*\*Example Output:\*\*

```

Union of sets: {1, 2, 3, 4, 5, 6}

```

|  |
| --- |
| void main() {  // Create two sets of integers  Set<int> set1 = {1, 2, 3, 4, 5};  Set<int> set2 = {4, 5, 6, 7, 8};  // Find the union of both sets  Set<int> unionSet = set1.union(set2);  // Print the result  print('Union of both sets: $unionSet');  } |

**Question P23:**

\*\*Task\*\*: Write a Dart program that finds the intersection of two sets of integers. The intersection should contain only the elements that are present in both sets.

- Create two sets of integers.

- Use the `.intersection()` method to find common elements.

- Print the result.

\*\*Example Input:\*\*

```

Set 1: {5, 10, 15, 20}

Set 2: {10, 20, 25, 30}

```

\*\*Example Output:\*\*

```

Intersection of sets: {10, 20}

```

|  |
| --- |
| void main() {  // Create two sets of integers  Set<int> set1 = {1, 2, 3, 4, 5};  Set<int> set2 = {4, 5, 6, 7, 8};  // Find the intersection of both sets  Set<int> intersectionSet = set1.intersection(set2);  // Print the result  print('Intersection of both sets: $intersectionSet');  } |

**Question P24:**

\*\*Task\*\*: Write a Dart program that computes the difference between two sets of integers. The difference should contain only the elements present in the first set but not in the second.

- Create two sets of integers.

- Use the `.difference()` method to find the difference.

- Print the result.

\*\*Example Input:\*\*

```

Set 1: {1, 2, 3, 4, 5}

Set 2: {3, 4, 5, 6, 7}

```

\*\*Example Output:\*\*

```

Difference of Set 1 - Set 2: {1, 2}

```

|  |
| --- |
| void main() {  // Create two sets of integers  Set<int> set1 = {1, 2, 3, 4, 5};  Set<int> set2 = {4, 5, 6, 7, 8};  // Find the difference between the two sets  Set<int> differenceSet = set1.difference(set2);  // Print the result  print('Difference between set1 and set2: $differenceSet');  } |

**Question P25:**

\*\*Task\*\*: Write a Dart program that converts a list of integers with duplicate elements into a set to remove duplicates. Then, print both the original list and the set.

- Create a list of integers with duplicate values.

- Convert the list to a set using `toSet()`.

- Print both the original list and the set (which removes duplicates).

\*\*Example Input:\*\*

```

List: [1, 2, 2, 3, 4, 4, 5]

```

\*\*Example Output:\*\*

```

Original List: [1, 2, 2, 3, 4, 4, 5]

Set (without duplicates): {1, 2, 3, 4, 5}

```

|  |
| --- |
| void main() {  // Create a list of integers with duplicate values  List<int> numbers = [1, 2, 2, 3, 4, 4, 5, 6, 6, 7, 8, 8, 9];  // Convert the list to a set to remove duplicates  Set<int> uniqueNumbers = numbers.toSet();  // Print the original list  print('Original list: $numbers');  // Print the set (which removes duplicates)  print('Set with unique elements: $uniqueNumbers');  } |

**Map**

**Question P26:**

\*\*Task\*\*: Write a Dart program to perform the following operations on a map that stores student names as keys and their grades as values:

- Add a new key-value pair.

- Update the grade of an existing student.

- Remove a student from the map.

- Print all students and their grades.

\*\*Example Input:\*\*

```

Initial Map: {'Alice': 85, 'Bob': 90, 'Charlie': 88}

Add: {'David': 92}

Update Bob's grade to 95

Remove: 'Charlie'

```

\*\*Example Output:\*\*

```

Updated Map: {'Alice': 85, 'Bob': 95, 'David': 92}

```

|  |
| --- |
| void main() {  // Create a map with student names as keys and their grades as values  Map<String, String> studentGrades = {  'Alice': 'A',  'Bob': 'B',  'Charlie': 'C'  };  // Add a new key-value pair  studentGrades['David'] = 'B+';  // Update the grade of an existing student  studentGrades['Alice'] = 'A+';  // Remove a student from the map  studentGrades.remove('Charlie');  // Print all students and their grades  studentGrades.forEach((student, grade) {  print('$student: $grade');  });  } |

**Question P27:**

\*\*Task\*\*: Write a Dart program that searches for a specific student in a map where the keys are student names and the values are their grades. If the student exists, print their grade. If the student does not exist, print a message saying that the student was not found.

- Create a map of student names and grades.

- Search for a student by name.

- Use `containsKey()` to check if the student exists.

\*\*Example Input:\*\*

```

Map: {'Alice': 85, 'Bob': 90, 'Charlie': 88}

Search for: 'Bob'

```

\*\*Example Output:\*\*

```

Bob's grade: 90

```

|  |
| --- |
| void main() {  // Create a map of student names and grades  Map<String, String> studentGrades = {  'Alice': 'A',  'Bob': 'B',  'Charlie': 'C'  };  // Student name to search for  String searchName = 'Bob';  // Check if the student exists in the map  if (studentGrades.containsKey(searchName)) {  print('$searchName\'s grade: ${studentGrades[searchName]}');  } else {  print('Student $searchName was not found.');  }  } |

**Question P28:**

\*\*Task\*\*: Write a Dart program that sorts a map of city names and populations by the city names (keys) in alphabetical order and prints the sorted map.

- Create a map of cities and their populations.

- Extract the entries, sort them by the city names (keys), and convert them back into a map.

- Print the sorted map.

\*\*Example Input:\*\*

```

Map: {'London': 9000000, 'Paris': 2140000, 'Berlin': 3700000}

```

\*\*Example Output:\*\*

```

Sorted Map: {'Berlin': 3700000, 'London': 9000000, 'Paris': 2140000}

|  |
| --- |
| void main() {  // Create a map of cities and their populations  Map<String, int> cityPopulations = {  'Tokyo': 37435191,  'Delhi': 29399141,  'Shanghai': 26317104,  'São Paulo': 21846507,  'Mumbai': 20411000  };  // Extract the entries and sort them by city names (keys)  var sortedEntries = cityPopulations.entries.toList()  ..sort((a, b) => a.key.compareTo(b.key));  // Convert the sorted entries back into a map  Map<String, int> sortedCityPopulations = {  for (var entry in sortedEntries) entry.key: entry.value  };  // Print the sorted map  print('Sorted city populations: $sortedCityPopulations');  } |

**Question P29:**

\*\*Task\*\*: Write a Dart program that sorts a map of product names and their prices by the prices (values) in ascending order and prints the sorted list of products.

- Create a map of products and their prices.

- Sort the map entries by values (prices) and print the sorted entries.

\*\*Example Input:\*\*

```

Map: {'Laptop': 1200, 'Phone': 800, 'Tablet': 600}

```

\*\*Example Output:\*\*

```

Sorted by Prices: {'Tablet': 600, 'Phone': 800, 'Laptop': 1200}

```

**Question P30:**

\*\*Task\*\*: Write a Dart program that manages a list of maps where each map contains information about a product (`name`, `price`, `quantity`). The program should:

- Add a new product to the list.

- Find a product by its name.

- Sort the products by price.

- Create a list of product maps, each containing `name`, `price`, and `quantity`.

- Perform operations such as searching for a product and sorting the list by price.

\*\*Example Input:\*\*

```

Product List: [

{'name': 'Laptop', 'price': 1200, 'quantity': 5},

{'name': 'Phone', 'price': 800, 'quantity': 10},

{'name': 'Tablet', 'price': 600, 'quantity': 8}

]

Add: {'name': 'Headphones', 'price': 150, 'quantity': 15}

Search for: 'Phone'

```

\*\*Example Output:\*\*

```

Added Product: {'name': 'Headphones', 'price': 150, 'quantity': 15}

Found Product: {'name': 'Phone', 'price': 800, 'quantity': 10}

Sorted Products by Price: [

{'name': 'Headphones', 'price': 150, 'quantity': 15},

{'name': 'Tablet', 'price': 600, 'quantity': 8},

{'name': 'Phone', 'price': 800, 'quantity': 10},

{'name': 'Laptop', 'price': 1200, 'quantity': 5}

]

|  |
| --- |
| void main() {  // Create a list of product maps  List<Map<String, dynamic>> products = [  {'name': 'Laptop', 'price': 999.99, 'quantity': 10},  {'name': 'Smartphone', 'price': 699.99, 'quantity': 20},  {'name': 'Tablet', 'price': 399.99, 'quantity': 15},  {'name': 'Smartwatch', 'price': 199.99, 'quantity': 30},  {'name': 'Headphones', 'price': 149.99, 'quantity': 25},  ];  // Function to add a new product to the list  void addProduct(String name, double price, int quantity) {  products.add({'name': name, 'price': price, 'quantity': quantity});  }  // Function to find a product by its name  Map<String, dynamic>? findProductByName(String name) {  return products.firstWhere((product) => product['name'] == name,  orElse: () => {});  }  // Function to sort products by price  void sortProductsByPrice() {  products.sort((a, b) => a['price'].compareTo(b['price']));  }  // Add a new product  addProduct('Keyboard', 49.99, 50);  // Find a product by its name  String searchName = 'Smartphone';  Map<String, dynamic>? foundProduct = findProductByName(searchName);  if (foundProduct != null && foundProduct.isNotEmpty) {  print('Product found: $foundProduct');  } else {  print('Product $searchName not found.');  }  // Sort products by price  sortProductsByPrice();  // Print the sorted list of products  print('Products sorted by price:');  for (var product in products) {  print(  'Name: ${product['name']}, Price: \$${product['price']}, Quantity: ${product['quantity']}');  }  } |

**List of Map**

**Question P31:**

Write a Dart program that creates a map of country names as keys and their capitals as values. Perform the following tasks:

- Add at least 5 country-capital pairs to the map.

- Access and print the capital of "Germany".

- Check if the map contains the key "India", and print the result.

|  |
| --- |
| void main() {  // Create a map of country names and their capitals  Map<String, String> countryCapitals = {  'Germany': 'Berlin',  'France': 'Paris',  'Italy': 'Rome',  'Japan': 'Tokyo',  'Canada': 'Ottawa'  };  // Access and print the capital of "Germany"  String germanyCapital = countryCapitals['Germany'] ?? 'Not found';  print('The capital of Germany is: $germanyCapital');  // Check if the map contains the key "India"  bool containsIndia = countryCapitals.containsKey('India');  print('Does the map contain the key "India"? $containsIndia');  } |

**Question P32:**

Create a map of students' names (keys) and their corresponding grades (values). Write a Dart function to iterate over the map and print each student's name along with their grade.

|  |
| --- |
| void main() {  // Create a map of students' names and their grades  Map<String, String> studentGrades = {  'Alice': 'A',  'Bob': 'B',  'Charlie': 'C',  'David': 'B+',  'Eve': 'A-'  };  // Function to print each student's name along with their grade  void printStudentGrades(Map<String, String> grades) {  grades.forEach((name, grade) {  print('$name: $grade');  });  }  // Call the function to print the student grades  printStudentGrades(studentGrades);  } |

**Question P33:**

Given the following map:

```dart

Map<String, int> inventory = {

'Apples': 50,

'Oranges': 30,

'Bananas': 20

};

```

Perform the following operations:

- Update the quantity of 'Oranges' to 45.

- Add a new item 'Mangoes' with a quantity of 60.

- Remove 'Bananas' from the map.

Print the updated map after each operation.

|  |
| --- |
| //list by age in ascending order and prints the sorted list.  void main() {  Map<String, int> inventory = {'Apples': 50, 'Oranges': 30, 'Bananas': 20};  /\*\*  \*  - Update the quantity of 'Oranges' to 45.  - Add a new item 'Mangoes' with a quantity of 60.  - Remove 'Bananas' from the map.  \*/  inventory['Oranges'] = 45;  inventory['Mangoes'] = 60;  inventory.remove('Bananas');  print(inventory);  } |

**Question P34:**

Write a Dart function that takes a list of strings as input and returns a map where the keys are words and the values are the number of times each word appears in the list. Test the function with the following list:

```dart

['apple', 'banana', 'apple', 'orange', 'banana', 'apple']

```

|  |
| --- |
| Map<String, int> countWordOccurrences(List<String> words) {  // Create an empty map to store word counts  Map<String, int> wordCounts = {};  // Iterate over each word in the list  for (String word in words) {  // If the word is already in the map, increment its count  if (wordCounts.containsKey(word)) {  wordCounts[word] = wordCounts[word]! + 1;  } else {  // If the word is not in the map, add it with a count of 1  wordCounts[word] = 1;  }  }  return wordCounts;  }  void main() {  // Test the function with the provided list  List<String> words = [  'apple',  'banana',  'apple',  'orange',  'banana',  'apple'  ];  Map<String, int> wordCounts = countWordOccurrences(words);  // Print the result  print(wordCounts);  } |

**Question P35:**

Write a Dart program that defines two maps:

```dart

Map<int, String> map1 = {1: 'One', 2: 'Two', 3: 'Three'};

Map<int, String> map2 = {4: 'Four', 5: 'Five'};

```

Merge `map2` into `map1` and print the result. Also, explain what happens if `map1` and `map2` have overlapping keys.

|  |
| --- |
| void main() {  // Define two maps with overlapping keys  Map<int, String> map1 = {1: 'One', 2: 'Two', 3: 'Three'};  Map<int, String> map2 = {2: 'Deux', 4: 'Four'};  // Merge map2 into map1  map1.addAll(map2);  // Print the result  print('Merged map with overlapping keys: $map1');  } |

**Dart Spread Operator**

**Question P36:**

Write a Dart program that combines two lists using the spread operator. Given:

```dart

List<int> list1 = [1, 2, 3];

List<int> list2 = [4, 5, 6];

```

Create a new list that combines both lists using the spread operator and prints the result.

|  |
| --- |
| void main() {  List<int> A = [12, 10, 8];  List<int> B = [11, 6, 8, 7];  var newlist = [...A, ...B];  print(newlist);  } |

**Question P37:**

Write a Dart function that accepts two nullable lists of integers and returns a new list combining both. Use the null-aware spread operator (`...?`) to handle cases where one or both lists might be `null`. For example:

```dart

List<int>? list1 = [1, 2, 3];

List<int>? list2 = null;

```

Test the function with various combinations of `null` and non-`null` lists.

|  |
| --- |
| List<int> combineLists(List<int>? list1, List<int>? list2) {  return [...?list1, ...?list2];  }  void main() {  // Test cases  List<int>? list1 = [1, 2, 3];  List<int>? list2 = [4, 5, 6];  List<int>? list3 = null;  List<int>? list4 = [7, 8, 9];  // Combining different combinations of lists  print('Combining list1 and list2: ${combineLists(list1, list2)}');  print('Combining list1 and list3: ${combineLists(list1, list3)}');  print('Combining list3 and list4: ${combineLists(list3, list4)}');  print('Combining list3 and list3: ${combineLists(list3, list3)}');  print('Combining list2 and list4: ${combineLists(list2, list4)}');  } |

**Question P38:**

Write a Dart program that merges two maps using the spread operator. Given:

```dart

Map<String, String> map1 = {'name': 'Alice', 'age': '25'};

Map<String, String> map2 = {'city': 'New York', 'country': 'USA'};

```

Create a new map that combines both `map1` and `map2` and prints the result.

|  |
| --- |
| void main() {  // Define the two maps  Map<String, String> map1 = {'name': 'Alice', 'age': '25'};  Map<String, String> map2 = {'city': 'New York', 'country': 'USA'};  // Combine the maps using the spread operator  Map<String, String> combinedMap = {...map1, ...map2};  // Print the result  print('Combined map: $combinedMap');  } |

**Question P39:**

Write a Dart program that uses the spread operator to build a list from multiple sources, including other lists and individual elements. For example:

```dart

List<int> evens = [2, 4, 6];

List<int> odds = [1, 3, 5];

```

Create a new list that starts with the number 0, followed by all the elements of `evens` and `odds`, and finally ends with the number 7.

|  |
| --- |
| void main() {  // Define the lists  List<int> evens = [2, 4, 6];  List<int> odds = [1, 3, 5];  // Create a new list using the spread operator  List<int> combinedList = [0, ...evens, ...odds, 7];  // Print the result  print('Combined list: $combinedList');  } |

**Question P40:**

Write a Dart function that takes a list of lists (nested lists) and flattens it into a single list using the spread operator. For example, given:

```dart

List<List<int>> nestedList = [[1, 2], [3, 4], [5, 6]];

```

Return a single list: `[1, 2, 3, 4, 5, 6]`.

|  |
| --- |
| List<int> flattenList(List<List<int>> nestedList) {  // Use the spread operator to flatten the nested list  return [for (var sublist in nestedList) ...sublist];  }  void main() {  // Example nested list  List<List<int>> nestedList = [  [1, 2],  [3, 4],  [5, 6]  ];  // Flatten the nested list  List<int> flatList = flattenList(nestedList);  // Print the result  print('Flattened list: $flatList');  } |

**Dart’s Collection if, Collection for**

Here are five practice problems for students to practice Dart's \*\*collection `if`\*\* and \*\*collection `for`\*\* in list, map, and set literals:

**Question P41:**

Write a Dart program that creates a list representing a shopping cart. Use the \*\*collection `if`\*\* to conditionally add an item "Coupon Discount" to the list only if a boolean variable `discountApplied` is `true`. The initial cart should contain `"Apples"`, `"Bananas"`, and `"Oranges"`. Print the final list based on whether the discount is applied or not.

```dart

bool discountApplied = true; // or false

```

|  |
| --- |
| void main() {  // Initial items in the shopping cart  List<String> shoppingCart = ['Apples', 'Bananas', 'Oranges'];  // Boolean variable to indicate if discount is applied  bool discountApplied = true; // Change to false to test the other condition  // Use collection if to conditionally add "Coupon Discount"  List<String> finalCart = [  ...shoppingCart,  if (discountApplied) 'Coupon Discount'  ];  // Print the final list  print('Final shopping cart: $finalCart');  } |

**Question P42:**

Write a Dart program that generates a list of numbers from 1 to 10 using the \*\*collection `for`\*\*. Then, use the \*\*collection `if`\*\* to include only even numbers in the final list. Print the list of even numbers.

```dart

List<int> numbers = [for (var i = 1; i <= 10; i++) if (i % 2 == 0) i];

```

|  |
| --- |
| void main() {  // Generate a list of numbers from 1 to 10 and include only even numbers  List<int> evenNumbers = [  for (var i = 1; i <= 10; i++)  if (i % 2 == 0) i  ];  // Print the list of even numbers  print('List of even numbers: $evenNumbers');  } |

**Question P43:**

Create a Dart program that uses a \*\*map literal\*\* to store product names as keys and their prices as values. Use the \*\*collection `if`\*\* to include an entry `"Discount"` only if a boolean `applyDiscount` is `true`. If the discount is applied, the value should be a 10% deduction of the total price. Otherwise, no discount should appear in the map.

```dart

bool applyDiscount = false; // or true

```

|  |
| --- |
| void main() {  // Define the product prices  Map<String, double> productPrices = {  'Laptop': 1000.0,  'Smartphone': 800.0,  'Tablet': 500.0  };  // Boolean variable to indicate if discount is applied  bool applyDiscount = true; // Change to false to test the other condition  // Calculate the total price  double totalPrice = productPrices.values.reduce((a, b) => a + b);  // Use collection if to conditionally add "Discount"  Map<String, double> finalPrices = {  ...productPrices,  if (applyDiscount) 'Discount': totalPrice \* 0.10  };  // Print the final map  print('Final prices: $finalPrices');  } |

**Question P44:**

Write a Dart program that creates a \*\*set\*\* of favorite colors. The set should initially contain `'Blue'`, `'Green'`, and `'Red'`. Use the \*\*collection `if`\*\* to include `'Purple'` only if a boolean variable `likesPurple` is `true`. Print the final set of favorite colors.

```dart

bool likesPurple = true; // or false

```

|  |
| --- |
| void main() {  // Initial set of favorite colors  Set<String> favoriteColors = {'Blue', 'Green', 'Red'};  // Boolean variable to indicate if 'Purple' is liked  bool likesPurple = true; // Change to false to test the other condition  // Use collection if to conditionally add 'Purple'  Set<String> finalColors = {...favoriteColors, if (likesPurple) 'Purple'};  // Print the final set of favorite colors  print('Final set of favorite colors: $finalColors');  } |

**Question P45:**

Write a Dart program that generates a list of the first 10 Fibonacci numbers using the \*\*collection `for`\*\*. Use the \*\*collection `if`\*\* to include only numbers greater than 10 in the final list. Print the result.

```dart

List<int> fibonacci = [0, 1, for (int i = 2; i < 10; i++) fibonacci[i - 1] + fibonacci[i - 2]];

```

|  |
| --- |
| void main() {  // Generate the first 10 Fibonacci numbers  List<int> fibonacci = [0, 1];  for (int i = 2; i < 10; i++) {  fibonacci.add(fibonacci[i - 1] + fibonacci[i - 2]);  }  // Use collection if to include only numbers greater than 10  List<int> filteredFibonacci = [  for (var num in fibonacci)  if (num > 10) num  ];  // Print the result  print('First 10 Fibonacci numbers: $fibonacci');  print('Fibonacci numbers greater than 10: $filteredFibonacci');  } |

**Control flow operators**

Here are five practice problems for students to practice Dart's control flow operators (`if`, `else`, `else if`, `switch`, `for`, `while`, etc.):

**Question P46:**

Write a Dart function that takes an integer as input and uses an `if-else` statement to determine whether the number is even or odd. The function should print "Even" if the number is even, and "Odd" if it’s odd. Test the function with different integer inputs.

|  |
| --- |
| void checkEvenOrOdd(int number) {  if (number % 2 == 0) {  print('Even');  } else {  print('Odd');  }  }  void main() {  // Test the function with different integer inputs  checkEvenOrOdd(4); // Even  checkEvenOrOdd(7); // Odd  checkEvenOrOdd(10); // Even  checkEvenOrOdd(15); // Odd  checkEvenOrOdd(0); // Even  } |

**Question P47:**

Create a Dart program that takes a student's score (an integer between 0 and 100) and uses `if-else if-else` statements to determine their grade. The grading system should be as follows:

- Score >= 90: Grade A

- Score >= 80: Grade B

- Score >= 70: Grade C

- Score >= 60: Grade D

- Score < 60: Grade F

Print the grade based on the score.

|  |
| --- |
| void determineGrade(int score) {  if (score >= 90) {  print('Grade A');  } else if (score >= 80) {  print('Grade B');  } else if (score >= 70) {  print('Grade C');  } else if (score >= 60) {  print('Grade D');  } else {  print('Grade F');  }  }  void main() {  // Test the function with different scores  determineGrade(95); // Grade A  determineGrade(85); // Grade B  determineGrade(75); // Grade C  determineGrade(65); // Grade D  determineGrade(55); // Grade F  } |

**Question P48:**

Write a Dart function that takes an integer between 1 and 7 as input, where each number corresponds to a day of the week (1 for Monday, 2 for Tuesday, etc.). Use a `switch` statement to print the name of the corresponding weekday. If the number is outside the range 1–7, print "Invalid day".

|  |
| --- |
| void printWeekday(int day) {  switch (day) {  case 1:  print('Monday');  break;  case 2:  print('Tuesday');  break;  case 3:  print('Wednesday');  break;  case 4:  print('Thursday');  break;  case 5:  print('Friday');  break;  case 6:  print('Saturday');  break;  case 7:  print('Sunday');  break;  default:  print('Invalid day');  }  }  void main() {  // Test the function with different inputs  printWeekday(1); // Monday  printWeekday(4); // Thursday  printWeekday(7); // Sunday  printWeekday(0); // Invalid day  printWeekday(8); // Invalid day  } |

**Question P49:**

Given a list of integers:

```dart

List<int> numbers = [10, 20, 30, 40, 50];

```

Write a Dart program that uses a `for` loop to iterate over the list and prints the square of each number. For example, for the number 10, the program should print `100`.

|  |
| --- |
| void main() {  // List of integers  List<int> numbers = [10, 20, 30, 40, 50];  // Iterate over the list and print the square of each number  for (int number in numbers) {  print('The square of $number is ${number \* number}');  }  } |

**Question P50:**

Write a Dart function that takes a positive integer as input and uses a `while` loop to calculate the sum of its digits. For example, if the input is `123`, the output should be `6` (1 + 2 + 3). Print the result.

|  |
| --- |
| void sumOfDigits(int number) {  int sum = 0;  int temp = number;  while (temp > 0) {  sum += temp % 10;  temp ~/= 10;  }  print('The sum of the digits of $number is $sum');  }  void main() {  // Test the function with different inputs  sumOfDigits(123); // The sum of the digits of 123 is 6  sumOfDigits(456); // The sum of the digits of 456 is 15  sumOfDigits(789); // The sum of the digits of 789 is 24  } |

**Dart’s Patterns**

Here are five practice problems for students to practice patterns in Dart, which may include destructuring, matching, and working with Dart's pattern syntax in various contexts:

**Question P51:**

Given a list of integers:

```dart

List<int> numbers = [10, 20, 30, 40, 50];

```

Use Dart's destructuring syntax to extract the first two elements from the list into variables and print them. Also, assign the remaining elements to another list and print that as well.

|  |
| --- |
| void main() {  // Given list of integers  List<int> numbers = [10, 20, 30, 40, 50];  // Extract the first two elements  int first = numbers[0];  int second = numbers[1];  // Assign the remaining elements to another list  List<int> remaining = numbers.sublist(2);  // Print the extracted elements  print('First element: $first');  print('Second element: $second');  // Print the remaining elements  print('Remaining elements: $remaining');  } |

**Question P52:**

Write a Dart program that defines a class `Person` with fields `name` and `age`. Create a function that takes a `Person` object and uses a `switch` statement with object patterns to print different messages based on the person's age group:

- If the person is younger than 18, print "Minor".

- If the person is between 18 and 60, print "Adult".

- If the person is over 60, print "Senior".

|  |
| --- |
| class Person {  String name;  int age;  Person(this.name, this.age);  }  void printAgeGroup(Person person) {  switch (person.age) {  case var age when age < 18:  print('${person.name} is a Minor');  break;  case var age when age >= 18 && age <= 60:  print('${person.name} is an Adult');  break;  case var age when age > 60:  print('${person.name} is a Senior');  break;  default:  print('Invalid age');  }  }  void main() {  // Test the function with different Person objects  Person person1 = Person('Alice', 17);  Person person2 = Person('Bob', 25);  Person person3 = Person('Charlie', 65);  printAgeGroup(person1); // Alice is a Minor  printAgeGroup(person2); // Bob is an Adult  printAgeGroup(person3); // Charlie is a Senior  } |

**Question P53:**

Write a Dart function that takes a `Map<String, dynamic>` representing a product with fields `name`, `price`, and an optional `discount`. Use pattern matching to extract the `price` and `discount` (if available), then calculate and print the final price. If no discount is present, print the original price.

|  |
| --- |
| void calculateFinalPrice(Map<String, dynamic> product) {  // Extract the price and discount using pattern matching  double price = product['price'];  double discount = product.containsKey('discount') ? product['discount'] : 0.0;  // Calculate the final price  double finalPrice = price - (price \* discount / 100);  // Print the final price  if (discount > 0) {  print(  'The final price after a discount of $discount% is: \$${finalPrice.toStringAsFixed(2)}');  } else {  print('The original price is: \$${price.toStringAsFixed(2)}');  }  }  void main() {  // Test the function with different products  Map<String, dynamic> product1 = {  'name': 'Laptop',  'price': 1000.0,  'discount': 10.0  };  Map<String, dynamic> product2 = {'name': 'Smartphone', 'price': 800.0};  calculateFinalPrice(  product1); // The final price after a discount of 10.0% is: $900.00  calculateFinalPrice(product2); // The original price is: $800.00  } |

**Question P54:**

Given the following list of lists:

```dart

List<List<int>> nestedList = [

[1, 2],

[3, 4],

[5, 6, 7]

];

```

Write a Dart program that uses pattern matching to find and print the inner list that contains exactly 3 elements. If no such list exists, print "No match found".

|  |
| --- |
| void main() {  // Given list of lists  List<List<int>> nestedList = [  [1, 2],  [3, 4],  [5, 6, 7]  ];  // Variable to track if a match is found  bool matchFound = false;  // Iterate over the nested list  for (var innerList in nestedList) {  if (innerList.length == 3) {  print('Inner list with exactly 3 elements: $innerList');  matchFound = true;  break;  }  }  // If no match is found, print "No match found"  if (!matchFound) {  print('No match found');  }  } |

**Question P55:**

Write a Dart function that takes a tuple (a two-element list) as an argument, where the first element is a string representing a person's name, and the second element is an integer representing their age. Use Dart's pattern matching in the function parameter to destructure the tuple, then print a message like "John is 25 years old". Test the function with different tuples.

|  |
| --- |
| void printPersonInfo(List<dynamic> person) {  // Destructure the tuple using pattern matching  var [name, age] = person;  // Print the message  print('$name is $age years old');  }  void main() {  // Test the function with different tuples  printPersonInfo(['John', 25]); // John is 25 years old  printPersonInfo(['Alice', 30]); // Alice is 30 years old  printPersonInfo(['Bob', 22]); // Bob is 22 years old  } |

**Dart’s Switch Statement**

**Question P56:**

Write a Dart program that takes an integer between 1 and 7 as input, where each number corresponds to a day of the week (1 for Monday, 2 for Tuesday, etc.). Use a `switch` statement to print the name of the corresponding day. If the input number is outside the range, print "Invalid day".

|  |
| --- |
| void main() {  int day = 3;  // Use a switch statement to print the corresponding day  switch (day) {  case 1:  print('Monday');  break;  case 2:  print('Tuesday');  break;  case 3:  print('Wednesday');  break;  case 4:  print('Thursday');  break;  case 5:  print('Friday');  break;  case 6:  print('Saturday');  break;  case 7:  print('Sunday');  break;  default:  print('Invalid day');  }  } |

**Question P57:**

Create a Dart program that takes two numbers and an operator (`+`, `-`, `\*`, or `/`) as input. Use a `switch` statement to perform the appropriate arithmetic operation based on the input operator and print the result. If an invalid operator is entered, print "Invalid operator".

|  |
| --- |
| import 'dart:io';  void main() {  // Prompt the user to enter the first number  print('Enter the first number:');  double? num1 = double.tryParse(stdin.readLineSync()!);  // Prompt the user to enter the second number  print('Enter the second number:');  double? num2 = double.tryParse(stdin.readLineSync()!);  // Prompt the user to enter an operator  print('Enter an operator (+, -, \*, /):');  String? operator = stdin.readLineSync();  // Check if the numbers are valid  if (num1 == null || num2 == null || operator == null) {  print('Invalid input');  return;  }  // Use a switch statement to perform the appropriate operation  switch (operator) {  case '+':  print('Result: ${num1 + num2}');  break;  case '-':  print('Result: ${num1 - num2}');  break;  case '\*':  print('Result: ${num1 \* num2}');  break;  case '/':  if (num2 != 0) {  print('Result: ${num1 / num2}');  } else {  print('Error: Division by zero');  }  break;  default:  print('Invalid operator');  }  } |

**Question P58:**

Write a Dart program that simulates a traffic light system. The program should take a string as input (`"red"`, `"yellow"`, or `"green"`) and use a `switch` statement to print the following:

- If the input is `"red"`, print "Stop".

- If the input is `"yellow"`, print "Slow down".

- If the input is `"green"`, print "Go".

- For any other input, print "Invalid signal".

|  |
| --- |
| import 'dart:io';  void main() {  // Prompt the user to enter a traffic light signal  print('Enter a traffic light signal (red, yellow, green):');  String? signal = stdin.readLineSync()?.toLowerCase();  // Use a switch statement to print the appropriate message  switch (signal) {  case 'red':  print('Stop');  break;  case 'yellow':  print('Slow down');  break;  case 'green':  print('Go');  break;  default:  print('Invalid signal');  }  } |

**Question P59:**

Write a Dart function that takes the name of a month as a string (e.g., `"January"`, `"February"`) and uses a `switch` statement to determine which season the month falls into. Print one of the following:

- "Winter" for December, January, and February

- "Spring" for March, April, and May

- "Summer" for June, July, and August

- "Autumn" for September, October, and November

If the input is not a valid month name, print "Invalid month".

|  |
| --- |
| void determineSeason(String month) {  switch (month.toLowerCase()) {  case 'december':  case 'january':  case 'february':  print('Winter');  break;  case 'march':  case 'april':  case 'may':  print('Spring');  break;  case 'june':  case 'july':  case 'august':  print('Summer');  break;  case 'september':  case 'october':  case 'november':  print('Autumn');  break;  default:  print('Invalid month');  }  }  void main() {  // Test the function with different month names  determineSeason('January'); // Winter  determineSeason('April'); // Spring  determineSeason('July'); // Summer  determineSeason('October'); // Autumn  determineSeason('Invalid'); // Invalid month  } |

**Question P60:**

Write a Dart program that simulates a restaurant menu. The program should display the following options to the user:

1. Pizza

2. Burger

3. Pasta

4. Salad

Use a `switch` statement to print the price of the selected item based on the user's input. If the user selects an invalid option, print "Invalid choice".

|  |
| --- |
| import 'dart:io';  void main() {  // Display the menu options  print('Restaurant Menu:');  print('1. Pizza');  print('2. Burger');  print('3. Pasta');  print('4. Salad');  // Prompt the user to enter their choice  print('Enter the number of your choice:');  int? choice = int.tryParse(stdin.readLineSync()!);  // Use a switch statement to print the price of the selected item  switch (choice) {  case 1:  print('Pizza: \$12.00');  break;  case 2:  print('Burger: \$8.00');  break;  case 3:  print('Pasta: \$10.00');  break;  case 4:  print('Salad: \$6.00');  break;  default:  print('Invalid choice');  }  } |

**Question A2:**

Initialize a list of records, consisting of name and age values. Sort list with respect to name and then with age.

|  |
| --- |
| /\*  Create a list of records, consisting of name and age values. Sort list with respect to name and age.  \*/  void main(List <String>args) {  var records = [  (name: 'Ali', age: 45),  (name: 'Javed', age: 54),  (name: 'Salman', age: 36),  (name: 'Ben', age: 36),  (name: 'Javed', age: 45),  ];  print("Before sorting: ");  print(records);  records.sort((a, b) {  var result = a.name.compareTo(b.name);  if( result == 0)  {  return a.age.compareTo((b.age));  }  else  {  return result;  }      });  print("After sorting: ");  print(records);  } |

**Question A3:**

Initialize a list of integers, 10, 20, 30, 40.

Create another list using the first list, such that at its initialization, the new list is initialized like this:

Item 1: 10, Item 2: 20, Item 3: 30, Item 4: 40.

|  |
| --- |
| /\*  Create a list of integers, 10, 20, 30, 40.  Create another list using the first list, such that at its initialization, the new list is initialized like this:  Item 1: 10, Item 2: 20, Item 3: 30, Item 4: 40.  \*/  void main(List <String>args) {  var items = [10, 20, 30, 40];  var newlist = [for( int i in items) 'Item ${ (i/10).round()}: $i'];  //Item 1: 10, Item 2: 20, Item 3: 30, Item 4: 40  print(newlist);  } |

.

**Question A4:**

Suppose we have initialized a list of 4 integers. You need to sum the elements of the list without using any loops or calling list elements through their indexes.

|  |
| --- |
| /\*  Suppose we have a list of 4 integers. You need to sum the elements of the list without using any loops or calling list elements through their indexes.  \*/  void main(List <String>args) {  var items = [10, 20, 30, 40];  int sum = items.reduce((a, b) => a + b);  print(sum);  } |

**Question A5:**

Suppose we have two numbers a=10 and b=20. You need to swap the numbers without using any third temporary variable, or any arithmetic or logical operators.

|  |
| --- |
| /\*  Suppose we have two numbers a=10 and b=20. You need to swap the numbers without using any third temporary variable, or any arithmetic or logical operators.  \*/  void main(List <String>args) {  int a = 10;  int b = 20;  (a, b) = (b, a );  print(a);  print(b);  } |

**Question A6:**

Suppose you have a range of numbers, and their respective grades:

10 – 30, grade E

31 – 50, grade D

51 – 70, grade C

71 – 90, grade B

91 – 100, grade A

Write a switch – case statement, that takes the marks and show the grade.

|  |
| --- |
| /\*  Suppose you have a range of numbers, and their respective grades:  10 – 30, grade E  31 – 50, grade D  51 – 70, grade C  71 – 90, grade B  91 – 100, grade A  Write a switch – case statement, that takes the marks and show the grade.  \*/  void main(List <String>args) {  int num = 45;  switch(num)  {  case >= 10 && <= 30:  print("grade E");  break;  case >= 31 && <= 50:  print("grade D");  break;  case >= 51 && <= 70:  print("grade C");  break;  case >= 71 && <= 90:  print("grade B");  break;  case >= 91 && <= 100:  print("grade A");  break;  }  } |

**Question B7:**

Initialize a list of Map with the following items:

{"position": 10, "name": "Jawad"},

{"position": 33, "name": "Faisal"},

{"position": 4, "name": "Zahid"},

{"position": 6, "name": "Ali"},

{"position": 9, "name": "Noman"},

{"position": 4, "name": "Ben"},

Sort the list with first with respect to position, and then with respect to name in case the positions are same.

|  |
| --- |
| void main() {  List<Map<String, dynamic>> students = [  {"position": 10, "name": "Jawad"},  {"position": 33, "name": "Faisal"},  {"position": 4, "name": "Zahid"},  {"position": 6, "name": "Ali"},  {"position": 9, "name": "Noman"},  {"position": 4, "name": "Ben"},  ];  // select \* from students order by position, name;  students.sort((a, b) {  var test = a["position"].compareTo(b["position"]);  if( test == 0 )  return a["name"].compareTo(b["name"]);  else  return test;  });  print(students);  } |

**Question B8:**

Initialize a list of Map with the following items:

{"name":"Ali", "age":45, "marks":32 },

{"name":"Noman", "age":32, "marks":23 },

{"name":"Faisal", "age":41, "marks":43 },

{"name":"Noman", "age":11, "marks":43 },

{"name":"Faisal", "age":8, "marks":43 },

Print those records whose age is greater than 30 and whose name is either Noman or Faisal

|  |
| --- |
| void main() {  List<Map<String, dynamic>> mylist = [  {"name":"Ali", "age":45, "marks":32 },  {"name":"Noman", "age":32, "marks":23 },  {"name":"Faisal", "age":41, "marks":43 },  {"name":"Noman", "age":11, "marks":43 },  {"name":"Faisal", "age":8, "marks":43 },  ];  mylist.where( (item) => item['age'] > 30 && (item['name']=='Noman'||item['name']=='Faisal') ).forEach((item) {  print("${item['name']} ${item['age']} ${item['marks']}");  }  );  } |

**Question A8:**

Write an example of function definition and function call with named parameters.

|  |
| --- |
| /\*  Write an example of function definition and function call with named parameters.  \*/  void main() {  myfunction(first: 10, second: 20);  }  void myfunction({int? first, int? second}) {  int ans = (first ?? 0) + (second ?? 0);  print(ans);  } |

**Question B2:**

Write arrow functions for the following equations:

|  |
| --- |
| /\*  Write arrow functions for the following equations:  \*/  void main() {  print( Z(2,3) );  }  var A = (int a, int b) => a\*a + b\*b\*b\*b;  var Z = (int p, int t) => p\*p + 5\*t + A(1,2); |

**Question A9:**

Write arrow functions for the following equations:

Z = a 2 + 4.B2 – 8b + 2a

|  |
| --- |
| /\*  Write arrow functions for the following equations:  \*/  import 'dart:math';  void main() {  print( A(1, 2, 3) );  }  var A = (int x, int y, int p) => x\*x + 2\*x\*y + p\*Z(2,3);  var Z = (int a, int b) => a\*a + 4 \* pow( B(1,2), 2) - 8\*b + 2\*a;  var B = (int n, int q) => n\*n + q\*n + 1; |

**Question A10:**

Suppose the equation is:

Z = x2 + 4y2 – 8N2

Where N is represented by a separate equation:

N = p2 + q2

Solve ‘Z’ with arrow function, such that you need to define the arrow function N within the body of Z.

|  |
| --- |
| import 'dart:math';  void main() {  print(Z(4, 5)(3, 4));  }  var Z = (int x, int y) =>  (int p, int q) => x \* x + 4 \* y \* y - 8 \* pow(p \* p + q \* q, 2); |

**Question A13:**

Given the following list: ['apples', 'bananas', 'oranges'];

Append a string with each element of the list and capitalize each element of list. Use a combination of map and forEach function.

|  |
| --- |
| /\*  Question A13:  Given the following list: ['apples', 'bananas', 'oranges'];  Append a string with each element of the list and capitalize each element of list. Use a combination of map and forEach function.  \*/  void main() {  var list = ['apples', 'bananas', 'oranges'];  list.map( (item) => '$item hello').forEach( (item) => print(item.toUpperCase()));  } |

**Question A14:**

Create a small calculator application using typedef functions performing these operations, add, subtract, multiply, and divide.

|  |
| --- |
| /\*  Question A14:  Create a small calculator application using typedef functions performing these operations, add, subtract, multiply, and divide.  \*/  typedef double Calculator(double a, double b);  void main() {  Calculator myfunction;  myfunction = addition;  print( myfunction(5, 6));  myfunction = subtraction;  print( myfunction(7, 3));  myfunction = multiplication;  print( myfunction(2, 3));  myfunction = division;  print( myfunction(9, 3));  }  double addition(double a, double b) {  return a+b;  }  double subtraction(double a, double b) {  return a-b;  }  double multiplication(double a, double b) {  return a\*b;  }  double division(double a, double b) {  return a/b;  } |

**Question A15:**

Suppose you have the following array,

List<Map<String, String>> myArray = [

{'name': 'ali', 'age': '45'},

{'name': 'noman', 'age': '34'},

];

Display the key and value of array elements.

Display the values of the array

|  |
| --- |
| /\*  Question A15:  Suppose you have the following array,  List<Map<String, String>> myArray = [  {'name': 'ali', 'age': '45'},  {'name': 'noman', 'age': '34'},  ];  Display the key and value of array elements.  Display the values of the array  \*/  void main() {  List<Map<String, String>> myArray = [  {'name': 'ali', 'age': '45'},  {'name': 'noman', 'age': '34'},  ];  // To display the key and values:  print("\n\n Printing key and values");  for (var map in myArray) {  map.forEach((key, value) {  print('$key: $value');  });  }  print("\n\n Using for loop");  for (var mapObject in myArray) {  // Access elements using key  print('Name: ${mapObject['name']}');  print('Age: ${mapObject['age']}');  print(''); // Add an empty line for better readability  }  } |

**Question A16:**

Suppose we have the following arrays:

var myArray1 = [3, 4, 5]

var myArray2 = [6, 7, 8]

Write code to append the myArray2 into myArray1.

|  |
| --- |
| /\*  Question A16:  Suppose we have the following arrays:  var myArray1 = [3, 4, 5]  var myArray2 = [6, 7, 8]  Write code to append the myArray2 into myArray1.  \*/  void main() {  var myArray1 = [3, 4, 5];  var myArray2 = [6, 7, 8];  var appended = [...myArray1, ...myArray2];  print(appended);  } |

**Question A17:**

Suppose we have an Dart object { 'name': 'Devin', 'hairColor': 'brown' }

Write code to change value of hairColor using spread syntax (…) three dots.

|  |
| --- |
| /\*  Question A17:  Suppose we have an Dart object { name: 'Devin', hairColor: 'brown' }  Write code to change value of hairColor using spread syntax (…) three dots.  \*/  void main() {  var object = {'name': 'Devin', 'hairColor': 'brown'};  print("Before change");  print(object);  object = {...object, 'hairColor':'red'};  print("After change");  print(object);  } |

**Question A18:**

Write an example of defining an arrow function within another arrow function.

|  |
| --- |
| import 'dart:math';  final outerFunction = (int x) => (int y) => x + pow(y, 3);  void main() {  // Outer arrow function  // Usage  final result = outerFunction(5)(3);  print('Result: $result'); // Output: Result: 8  } |

**Question A19:**

Create a class Person with attributes: id, name, age.

Derive two classes from person, named Student and Teacher.

The extra attributes of Student are cgpa, currently enrolled semester (e.g., FA22 or SP22, etc), admission date.

The extra attributes of Teacher are salary, designation (Lecturer, Assistant Professor, Professor, etc), department, and joining date.

Populate a list of at least 3 records in each class using class objects.

A user should be able to search a student or teacher with the provided ID. You should store objects of Teacher and Student in a list.

Print list of students whose cgpa is greater than 3.7.

|  |
| --- |
| /\*  Question A19:  Create a class Person with attributes: id, name, age.  Derive two classes from person, named Student and Teacher.  The extra attributes of Student are cgpa, currently enrolled semester (e.g., FA22 or SP22, etc), admission date.  The extra attributes of Teacher are salary, designation (Lecturer, Assistant Professor, Professor, etc), department, and joining date.  Populate a list of at least 3 records in each class using class objects.  A user should be able to search a student or teacher with the provided ID. You should store objects of Teacher and Student in a list.  Print list of students whose cgpa is greater than 3.7.  \*/  class Person {  String? id;  String? name;  int? age;  Person(this.id, this.name, this.age);  }  class Student extends Person {  double? cgpa;  String? semester;  String? admissionDate;  Student(super.id, super.name, super.age, this.cgpa, this.semester, this.admissionDate);  }  class Teacher extends Person {  double? salary;  String? designation;  String? department;  String? joiningDate;  Teacher(super.id, super.name, super.age, this.salary, this.designation, this.department, this.joiningDate);  }  void main() {    List<Student> students = [  Student('123', 'John Doe', 21, 3.1, 'Fa23', '2022-09-01'),  Student('456', 'Shahid Gul', 12, 2.8, 'Fa22', '2023-09-01'),  Student('432', 'Muneeza Malik', 21, 3.9, 'Sp22', '2023-10-01'),  Student('789', 'Javed Henry', 30, 2.9, 'Sp23', '2025-09-01'),  Student('728', 'Neelam Khan', 30, 3.7, 'Sp23', '2025-09-01'),  ];  List<Teacher> teachers = [  Teacher('444', 'Noman Ali', 21, 31000, 'AP', 'CS', '2022-09-01'),  Teacher('666', 'Ali Shahid', 12, 28000, 'Lecturer', 'EE', '2023-09-01'),  Teacher('777', 'Qasim Khan', 30, 29000, 'AssocProf', 'MS', '2025-09-01'),  ];    String studentId='456';  for(Student s in students) {  if(s.id == studentId) {  print("Student found");  break;  }  }  String teacherId='777';  for(Teacher t in teachers) {  if(t.id == teacherId) {  print("Teacher found");  break;  }  }  students.where((student) => student.cgpa! >= 3.7).forEach((student)=>print("${student.id} ${student.name}"));  } |

**Question A20:**

Given the following list of objects (name, age, marks), you need to write myObjects.where().forEach() function, so that the name, age, and marks of those students are printed on screen whose age is greater than 25 and marks are greater than equal to 50, and name is Alice or Bob

myObjects.add(Student(name: 'Alice', age: 25, marks: 55));

myObjects.add(Student(name: 'Bob', age: 30, marks: 50));

myObjects.add(Student(name: 'Alice', age: 27, marks: 40));

myObjects.add(Student(name: 'Charlie', age: 22, marks: 45));

|  |
| --- |
| /\*  Question A20:  Given the following list of objects, you need to write myObjects.where().forEach() function, so that the name, age, and marks of those students are printed on screen whose age is greater than 25 and marks are greater than equal to 50, and name is Alice or Bob  myObjects.add(Student(name: 'Alice', age: 25, marks: 55));  myObjects.add(Student(name: 'Bob', age: 30, marks: 50));  myObjects.add(Student(name: 'Alice', age: 27, marks: 40));  myObjects.add(Student(name: 'Charlie', age: 22, marks: 45));  \*/  class Student {  String? name;  int? age;  double? marks;    Student({this.name, this.age, this.marks});  }  void main() {  List<Student> myObjects = [];  myObjects.add(Student(name: 'Alice', age: 25, marks: 55));  myObjects.add(Student(name: 'Bob', age: 30, marks: 50));  myObjects.add(Student(name: 'Alice', age: 27, marks: 40));  myObjects.add(Student(name: 'Charlie', age: 22, marks: 45));  myObjects.where((student) => student.age! > 25 && student.marks! >= 50 && (student.name == "Alice" || student.name == "Bob")).forEach((student)=>print("${student.name} ${student.age} ${student.marks}"));  } |

**Question B1:**

Given the following list of objects, you need to write myObjects.where().forEach() function, so that the name, age, and marks of those students are printed on screen whose age is greater than 30 and name is Noman or Faisal.

Student(name:"Ali", age:45, marks:32 ),

Student(name:"Faisal", age:41, marks:43 ),

Student(name:"Noman", age:11, marks: 43),

Student(name:"Faisal", age:8, marks:43)

|  |
| --- |
| class Student{  String? name;  int? age;  int? marks;  Student({this.name, this.age, this.marks});  }  void main() {  var mylist = [  Student(name:"Ali", age:45, marks:32 ),  Student(name:"Faisal", age:41, marks:43 ),  Student(name:"Noman", age:11, marks: 43),  Student(name:"Faisal", age:8, marks:43)  ];  mylist.where( (item) => item.age! > 30 && (item.name=='Noman'||item.name=='Faisal') ).forEach((item) {  print("${item.name} ${item.age} ${item.marks}");  }  );  } |

# FLUTTER WIDGETS (CLO-2)

**Question C1:**

Make an app in Flutter that shows the following on screen:

A screenshot of a phone

Description automatically generated

To load an image from the assets folder in Flutter, you need to follow a few steps. Let’s make sure everything is set up correctly:

flutter:

  # The following line ensures that the Material Icons font is

  # included with your application, so that you can use the icons in

  # the material Icons class.

  uses-material-design: true

  # To add assets to your application, add an assets section, like this:

  assets:

    - images/

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {    runApp(const MyApp());  }  class MyApp extends StatelessWidget {    const MyApp({super.key});    @override    Widget build(BuildContext context) {      return MaterialApp(        home: Scaffold(          appBar: AppBar(            title: const Text('Fruit App'),          ),          body: const Center(            child: Column(              mainAxisAlignment: MainAxisAlignment.center,              children: [                FruitCard(imagePath: 'images/orange.jpg', fruitName: 'Banana'),                SizedBox(height: 20),                FruitCard(imagePath: 'images/ratti\_gali\_lake.jpg', fruitName: 'Orange'),              ],            ),          ),        ),      );    }  }  class FruitCard extends StatelessWidget {    final String imagePath;    final String fruitName;    const FruitCard({super.key, required this.imagePath, required this.fruitName});    @override    Widget build(BuildContext context) {      return Container(        width: 200,        height: 200,        decoration: BoxDecoration(          border: Border.all(color: Colors.black),          borderRadius: BorderRadius.circular(10),        ),        child: Column(          mainAxisAlignment: MainAxisAlignment.center,          children: [            Image.asset(imagePath, width: 100, height: 100),            const SizedBox(height: 10),            Text(              fruitName,              style: const TextStyle(fontWeight: FontWeight.bold),            ),          ],        ),      );    }  } |

**Question C2:**

Write a flutter app to show the grade of a student for the given marks. The marks are passed as an argument to the constructor of the widget class computing the grade. The computed grade is shown in the Text widget. Here is the grade distribution for different marks ranges:

< 50 --- F

>= 50 and < 60 --- E

>= 60 and < 70 --- D

>= 70 and < 80 --- C

>= 80 and < 90 --- B

>= 90 --- A

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {    const MyApp({super.key});  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(  title: const Text('Grade Calculator'),  ),  body: const Center(  child: GradeCalculator(marks: 65), // Example: Pass the marks here  ),  ),  );  }  }  class GradeCalculator extends StatelessWidget {  final int marks;  const GradeCalculator({required this.marks, super.key});  String calculateGrade() {  if (marks < 50) {  return 'F';  } else if (marks >= 50 && marks < 60) {  return 'E';  } else if (marks >= 60 && marks < 70) {  return 'D';  } else if (marks >= 70 && marks < 80) {  return 'C';  } else if (marks >= 80 && marks < 90) {  return 'B';  } else {  return 'A';  }  }  @override  Widget build(BuildContext context) {  final grade = calculateGrade();  return Text(  'Grade: $grade',  style: const TextStyle(fontSize: 24, fontWeight: FontWeight.bold),  );  }  } |

**Question C3:**

Use constructor arguments to pass names of students from a Name() widget to an Attendance() widget. The following should be the output by Attendance() widget, where Present or Absent status is randomly generated.

Ali Khan Present

Noman Present

Faisal Absent

Javed Absent

|  |
| --- |
| import 'package:flutter/material.dart';  import 'dart:math'; // For random number generation  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(  title: const Text('Student Attendance'),  ),  body: const Center(  child: Attendance(names: ['Ali Khan', 'Noman', 'Faisal', 'Javed']),  ),  ),  );  }  }  class Attendance extends StatelessWidget {  final List<String> names;  const Attendance({super.key, required this.names});  String getRandomStatus() {  final random = Random();  return random.nextBool() ? 'Present' : 'Absent';  }  @override  Widget build(BuildContext context) {  return Column(  mainAxisAlignment: MainAxisAlignment.center,  children: [  for (var name in names)  Padding(  padding: const EdgeInsets.all(8.0),  child: Text('$name\t\t${getRandomStatus()}'),  ),  ],  );  }  } |

**Question C4:**

Write code to add a button in Flutter. The text showing in the button should be Click Here. When the button is clicked, a Snackbar should be shown with message “hello world”.

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp( MaterialApp(  home: Scaffold(    appBar: AppBar(  title: const Text('Button Example'),  ),  body: const MyApp(),  )  )  );  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return Center(  child: ElevatedButton(  onPressed: () {  // Show a Snackbar when the button is clicked  ScaffoldMessenger.of(context).showSnackBar(  const SnackBar(  content: Text('Hello, world!'),  backgroundColor: Colors.blue,  ),  );  },  child: const Text('Click Here'),  ),  );  }  } |

**Question C5:**

Write the code of TextFied() widget function.

When a user enters any text in the TextField(), it is also automatically written in another TextField() in capital letters.

|  |
| --- |
| /\*  Question C5:  Write the code of TextFied() widget function.  When a user enters any text in the TextField(), it is also automatically written in another TextField() in capital letters.  \*/  import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(  home: Scaffold(  body: Center(  child: InputDisplay(),  )  )  ));  }  class InputDisplay extends StatefulWidget {  const InputDisplay({super.key});  @override  State<InputDisplay> createState() => \_InputDisplayState();  }  class \_InputDisplayState extends State<InputDisplay> {  String message = "";  String message1="";  String userInput="";  @override  Widget build(BuildContext context) {  return Column(  mainAxisAlignment: MainAxisAlignment.center,  children: <Widget>[  TextField(  onChanged: (value) => setState(() => userInput = value  ),  decoration: const InputDecoration(  hintText: 'Enter your message'  ),  ),  TextField(  controller: TextEditingController(text: userInput.toUpperCase()),  decoration: const InputDecoration(  hintText: 'Enter your message'  ),  ),  ],  );  }  } |

**Question C6:**

Write a flutter program in which when a button is clicked, the text of first TextField() is assigned to second TextField()

|  |
| --- |
| /\*  Question C6:  Write a flutter program in which when a button is clicked, the text of first TextField() is assigned to second TextField()  \*/  import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(  home: Scaffold(  body: Center(  child: InputDisplay(),  )  )  ));  }  class InputDisplay extends StatefulWidget {  const InputDisplay({super.key});  @override  State<InputDisplay> createState() => \_InputDisplayState();  }  class \_InputDisplayState extends State<InputDisplay> {  String message = "";  String userInput="";  @override  Widget build(BuildContext context) {  return Column(  mainAxisAlignment: MainAxisAlignment.center,  children: <Widget>[  TextField(  onChanged: (value) => setState(() => userInput = value  ),  decoration: const InputDecoration(  hintText: 'Enter your message'  ),  ),  ElevatedButton(onPressed: () => setState(    () {  message = userInput;    }        ), child: Text("Press button")),    TextField(  controller: TextEditingController(text: message),  decoration: const InputDecoration(  hintText: 'Enter your message'  ),  ),  ],  );  }  } |

**Question C7:**

Write a flutter code, so that when the button is clicked, the text “hello world” should be shown in the Text() widget, and the button should be disabled.

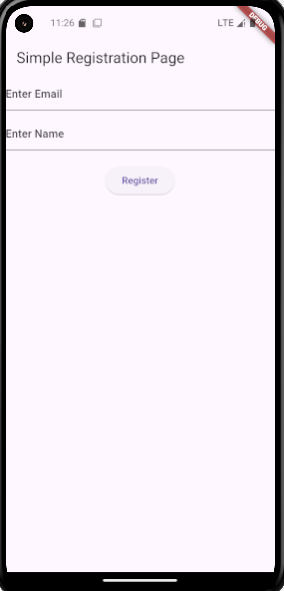
|  |
| --- |
| /\*  Question C7:  Write a flutter code, so that when the button is clicked, the text “hello world” should be shown in the Text() widget, and the button should be disabled.  \*/  import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: MyHomePage(),  );  }  }  class MyHomePage extends StatefulWidget {  const MyHomePage({super.key});  @override  \_MyHomePageState createState() => \_MyHomePageState();  }  class \_MyHomePageState extends State<MyHomePage> {  bool isButtonDisabled = false;  void \_showHelloWorld() {  setState(() {  isButtonDisabled = true;  });  // Show a Snackbar with "Hello, world!"    }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Button Example'),  ),  body: Center(  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: [  Text(  isButtonDisabled ? 'Hello, world!' : '',  style: const TextStyle(fontSize: 20),  ),  const SizedBox(height: 20),  ElevatedButton(  onPressed: isButtonDisabled ? null : \_showHelloWorld,  child: const Text('Click Here'),  ),  ],  ),  ),  );  }  } |

**Question C8:**

Create a simple registration page in Flutter asking for user’s email and name. When the user clicks on register button, the information should be shown using Text widgets. If any input is missing, snackbar message should be shown about the missing element. Use TextEditingController() class to get values of TextField().

|  |
| --- |
| /\*  Question C8:  Create a simple registration page in Flutter asking for user’s email and name. When the user clicks on register button, the information should be shown using Text widgets. If any input is missing, snackbar message should be shown about the missing element. Use TextEditingController() class to get values of TextField().  \*/  import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatefulWidget {  const MyApp({super.key});  @override  State<MyApp> createState() => \_MyAppState();  }  class \_MyAppState extends State<MyApp> {  // Controllers to store text from each box  final TextEditingController \_textController1 = TextEditingController();  final TextEditingController \_textController2 = TextEditingController();  String \_displayText = ""; // Variable to store combined text  void \_onPressed() {  // Combine text from controllers and update display text  setState(() {  \_displayText = "${\_textController1.text} - ${\_textController2.text}";  });  }  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(  title: const Text('Simple Registration Page'),  ),  body: Center(  child: Column(  mainAxisAlignment: MainAxisAlignment.start,  children: [  TextField(  controller: \_textController1,  decoration: const InputDecoration(  hintText: 'Enter Email',  ),  ),  const SizedBox(height: 10),  TextField(  controller: \_textController2,  decoration: const InputDecoration(  hintText: 'Enter Name',  ),  ),  const SizedBox(height: 20),  ElevatedButton(  onPressed: \_onPressed,  child: const Text('Register'),  ),  const SizedBox(height: 10),  Text(\_displayText),  ],  ),  ),  ),  );  }  } |

**Output:**



**Question C9:**

Suppose you have two TextField(), each containing a number, and a button to add the values of the two TextField(). When the button is clicked, the values of the TextField() are added and result should be shown in a Text() widget.

|  |
| --- |
| /\*  Question C9:  Suppose you have two TextField(), each containing a number, and a button to add the values of the two TextField(). When the button is clicked, the values of the TextField() are added and result should be shown in a Text() widget.  \*/  import 'package:flutter/material.dart';  void main() {  runApp(  const MaterialApp(  home: Scaffold(  body: Center(  child: Counter(),  ),  ),  ),  );  }  class Counter extends StatefulWidget {  const Counter({super.key});  @override  State<Counter> createState() => \_CounterState();  }  class \_CounterState extends State<Counter> {  final TextEditingController \_counter1 = TextEditingController(text: "100");  final TextEditingController \_counter2 = TextEditingController(text: "200");    int \_sum = 0;    void \_sumcounter() {  setState(() {  int c1=0;  int c2=0;  if( int.tryParse(\_counter1.text) != null ) {  c1 = int.parse(\_counter1.text);  }  if( int.tryParse(\_counter2.text) != null ) {  c2 = int.parse(\_counter2.text);  }  \_sum = c1 + c2;  });  }  @override  Widget build(BuildContext context) {  return Column( // Use Column for vertical layout  mainAxisAlignment: MainAxisAlignment.center, // Center the content vertically  children: <Widget>[    TextField(  controller: \_counter1,  ) ,// Spacing between elements  TextField(  controller: \_counter2,  ),  ElevatedButton(  onPressed: \_sumcounter,  child: const Text('Sum'),  ),  Text("$\_sum")  ],  );  }  } |

**Question C10:**

The following layout has three number buttons, a plus and equal operator, and a TextField() initialized with a zero “0”.

0

2

+

1

3

=

The user should be able to enter an expression like this: 423+35+223. When the user press the equal button, the answer should be shown in the TextField().

|  |
| --- |
| /\*  The following layout has three number buttons, a plus and equal operator, and a TextField() initialized with a zero “0”.  The user should be able to enter an expression like this: 423+35+223. When the user press the equal button, the answer should be shown in the TextField().  \*/  import 'package:flutter/material.dart';  void main() {  runApp(  MaterialApp(  home: Scaffold(  appBar: AppBar(  title: const Text('Calculator'),  ),  body: const MyApp()  ),  )  );  }  class MyApp extends StatefulWidget {  const MyApp({super.key});  @override  State<MyApp> createState() => \_MyAppState();  }  class \_MyAppState extends State<MyApp> {  // Controllers to store text from each box  final TextEditingController \_mainscreen = TextEditingController();  String mainscreentext = "";  String previous="";  void callAlert(String msg) {  AlertDialog alert = AlertDialog(  title: const Text("Invalid Button Press"),  content: Text(msg),  actions: [  TextButton(  onPressed: () {  Navigator.pop(context);  },  child: const Text('OK'),  ),  ],  );  // Show the dialog  showDialog(  context: context,  builder: (BuildContext context) {  return alert;  },  );  }  void \_onPressed(String btntext) {  setState(() {  if(btntext.compareTo("Clear")==0) {  \_mainscreen.text = "";  }  else if(btntext.compareTo('=') == 0 )  {  if(previous.compareTo("+")==0)  {  callAlert("Invalid expression");  \_mainscreen.text = "";  return;  }  var tokens =\_mainscreen.text.split('+');  int sum=0;  for(var item in tokens) {  sum = sum + int.parse(item);  }  \_mainscreen.text = "$sum";  }  // if the current button (btntext) pressed is same to the previous button  // and the previous button pressed was plus +  else if( (previous.compareTo(btntext) == 0 && previous.compareTo("+") == 0) ) {  callAlert("Cannot press an '+' here");  }  else {  previous = btntext;  \_mainscreen.text = "${\_mainscreen.text}$btntext";  }  });  }  @override  Widget build(BuildContext context) {  return Column(  mainAxisAlignment: MainAxisAlignment.center,  children: [    TextField(  controller: \_mainscreen,  decoration: const InputDecoration(  hintText: '0',  ),  ),  const SizedBox(height: 10),  Row(  children: [  ElevatedButton(  onPressed: ()=>\_onPressed("1"),  child: const Text('1'),  ),  const SizedBox(width: 10),  ElevatedButton(  onPressed: ()=>\_onPressed("2"),  child: const Text('2'),  ),  const SizedBox(width: 10),  ElevatedButton(  onPressed: ()=>\_onPressed("3"),  child: const Text('3'),  ),  const SizedBox(width: 10),  ElevatedButton(  onPressed: ()=>\_onPressed("+"),  child: const Text('+'),  ),  const SizedBox(width: 10),  ElevatedButton(  onPressed: ()=>\_onPressed("="),  child: const Text('='),  ),  const SizedBox(width: 10),    ],  ),  ElevatedButton(  onPressed: ()=>\_onPressed("Clear"),  child: const Text('Clear'),  ),  ]  );  }  } |

**Question C15:**

Show a list of students, such that :

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **CGPA** |
| 1 | Javed | 3.0 |
| 2 | Noman | 2.7 |
| 3 | Ali | 3.7 |
| 4 | Faisal | 3.3 |
| 5 | Shahid | 4.0 |
| 6 | Kamal | 3.1 |
| 7 | Zahid | 2.3 |

The students whose CGPA are in the range between 2 and less than 3 should be shown in bold and red font.

The students whose CGPA are in the range between 3 and less than 3.7 should be shown in blue font without bold

The students whose CGPA are greater than and equal to 3.7 should be shown in italic, bold, and green font.

|  |
| --- |
| /\*  Question C15:  Show a list of students, such that :  ID Name CGPA  1 Javed 3.0  2 Noman 2.7  3 Ali 3.7  4 Faisal 3.3  5 Shahid 4.0  6 Kamal 3.1  7 Zahid 2.3  The students whose CGPA are in the range between 2 and less than 3 should be shown in bold and  red font.  The students whose CGPA are in the range between 3 and less than 3.7 should be shown in blue font  without bold  The students whose CGPA are greater than and equal to 3.7 should be shown in italic, bold, and  green font  \*/  import 'package:flutter/material.dart';  class Student {  final int id;  final String name;  final double gpa;  Student(this.id, this.name, this.gpa);  }  void main() {  runApp(MaterialApp(  title: 'Flutter Demo',  theme: ThemeData(  colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),  useMaterial3: true,  ),  home: MyApp()));  }  class MyApp extends StatelessWidget {  MyApp({super.key});  final List<Student> students = [  Student(1, "Javed", 3.0),  Student(2, "Noman", 2.7),  Student(3, "Ali", 3.7),  Student(4, "Faisal", 3.3),  Student(5, "Shahid", 4.0),  Student(6, "Kamal", 3.1),  Student(7, "Zahid", 2.3)  ];  TextStyle? \_getTextStyle(double gpa) {  FontWeight fw = FontWeight.normal;  FontStyle fs = FontStyle.normal;  Color color = Colors.black;  if (gpa >= 2 && gpa < 3) {  color = Colors.red;  fw = FontWeight.bold;  fs = FontStyle.normal;  } else if (gpa >= 3 && gpa < 3.7) {  color = Colors.blue;  fw = FontWeight.normal;  fs = FontStyle.normal;  } else if (gpa >= 3.7) {  color = Colors.green;  fw = FontWeight.normal;  fs = FontStyle.italic;  }  return TextStyle(  fontWeight: fw,  fontStyle: fs,  color: color,  );  }  @override  Widget build(BuildContext context) {  return Scaffold(  body: Container(  padding: const EdgeInsets.all(50),  child: ListView.builder(  itemCount: students.length,  itemBuilder: (context, index) {  final item = students[index];  TextStyle? newstyle = \_getTextStyle(item.gpa);  return ListTile(  title: Row(  mainAxisAlignment: MainAxisAlignment.spaceBetween,  children: [  Text("${item.id}", style: newstyle),  Text(item.name, style: newstyle),  Text("${item.gpa}", style: newstyle),  ]));  })));  }  } |

**Question C35:**

Write code to show the following list using flutter <ListView> widget

[ {name: ‘Ali’, age: 33, city: ‘Karachi’}, {name: ‘Faisal’, age: 20, city: ‘Lahore’}, {name: ‘Noman’, age: 53, city: ‘Karachi’},]

|  |
| --- |
| /\*  Question C35:  Write code to show the following list using flutter <ListView> widget  [ {name: 'Ali', age: 33, city: 'Karachi'}, {name: 'Faisal', age: 20, city: 'Lahore'}, {name:  'Noman', age: 53, city: 'Karachi'},]  \*/  import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(title: 'Flutter Demo', home: MyApp()));  }  class MyApp extends StatelessWidget {  MyApp({super.key});  final List<Map<String, dynamic>> students = [  {"name": "Ali", "age": "33", "city": "Karachi"},  {"name": "Faisal", "age": "20", "city": "Lahore"},  {"name": "Noman", "age": "53", "city": "Islamabad" },  {"name": "Khan", "age": "43", "city": "Faisalabad"},  ];  @override  Widget build(BuildContext context) {  return Scaffold(  body: Container(  padding: const EdgeInsets.all(20),  child: ListView.builder(  itemCount: students.length,  itemBuilder: (context, index) {  final item = students[index];  return ListTile(  subtitle: Row(children: [  Expanded(child: Text(item["name"])),  Expanded(child: Text(item["age"])),  Expanded(child: Text(item["city"])),  ]));  })));  }  } |

**Output:**

A screenshot of a phone

Description automatically generated

**Question C35A:**

Write code to show the following list using flutter <ListView> widget

[ {name: ‘Ali’, age: 33, city: ‘Karachi’}, {name: ‘Faisal’, age: 20, city: ‘Lahore’}, {name: ‘Noman’, age: 53, city: ‘Karachi’},].   
Now add the styling to the same example so that it should look like the following:

A screenshot of a phone

Description automatically generated

Answer:

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {    runApp(MaterialApp(title: 'Flutter Demo', home: MyApp()));  }  class MyApp extends StatelessWidget {    MyApp({super.key});    final List<Map<String, dynamic>> students = [      {"name": "Ali", "age": "33", "city": "Karachi"},      {"name": "Faisal", "age": "20", "city": "Lahore"},      {"name": "Noman", "age": "53", "city": "Islamabad"},      {"name": "Khan", "age": "43", "city": "Faisalabad"},    ];    @override    Widget build(BuildContext context) {      return Scaffold(        appBar: AppBar(          title: const Text('Student List'),          backgroundColor: Colors.teal,        ),        body: Container(          padding: const EdgeInsets.all(16),          child: ListView.builder(            itemCount: students.length,            itemBuilder: (context, index) {              final item = students[index];              return Card(                elevation: 5,                margin: const EdgeInsets.symmetric(vertical: 8),                shape: RoundedRectangleBorder(                  borderRadius: BorderRadius.circular(10),                ),                child: ListTile(                  contentPadding: const EdgeInsets.all(16),                  leading: CircleAvatar(                    backgroundColor: Colors.teal.shade700,                    child: Text(                      item['name'][0],                      style: const TextStyle(color: Colors.white),                    ),                  ),                  title: Text(                    item["name"],                    style: const TextStyle(                      fontWeight: FontWeight.bold,                      fontSize: 18,                    ),                  ),                  subtitle: Column(                    crossAxisAlignment: CrossAxisAlignment.start,                    children: [                      const SizedBox(height: 4),                      Text("Age: ${item["age"]}"),                      Text("City: ${item["city"]}"),                    ],                  ),                  trailing: Icon(                    Icons.arrow\_forward\_ios,                    color: Colors.grey.shade600,                  ),                  onTap: () {                    // Add action on tap if needed                  },                ),              );            },          ),        ),      );    }  } |

**Question C33:**

We have following record of Student objects:

|  |  |  |
| --- | --- | --- |
| **RegNo** | **Name** | **Marks** |
| 1 | Ali | 80 |
| 2 | Noman | 60 |
| 3 | Faisal | 40 |
| 4 | Javed | 55 |

You need to show the above record using flutter’s ListView widget. Display a 4th column in the output that displays ‘pass’ if marks are greater than 50 and fail otherwise.

For example:

|  |  |  |  |
| --- | --- | --- | --- |
| **RegNo** | **Name** | **Marks** | **Status** |
| 1 | Ali | 80 | Pass |
| 2 | Noman | 60 | Pass |
| 3 | Faisal | 40 | Fail |
| 4 | Javed | 55 | Pass |

Solution

|  |
| --- |
| /\*  Question C33:  We have following record of students:  RegNo Name Marks  1 Ali 80  2 Noman 60  3 Faisal 40  4 Javed 55  You need to show the above record map function flutter’s ListView widget. Display a 4th column in the output that displays ‘pass’ if marks are greater than 50 and fail otherwise.  For example:  RegNo Name Marks Status  1 Ali 80 Pass  2 Noman 60 Pass  3 Faisal 40 Fail  4 Javed 55 Pass  \*/  Example output:    import 'package:flutter/material.dart';  class Student {  final int regno;  final String name;  final double marks;  Student(this.regno, this.name, this.marks);  }  void main() {  runApp(MaterialApp(title: 'Flutter Demo', home: MyApp()));  }  class MyApp extends StatelessWidget {  MyApp({super.key});  final List<Student> students = [  Student(35, "Ali", 33),  Student(15, "Nasir", 63),  Student(20, "Javed", 20),  ];  @override  Widget build(BuildContext context) {  return Scaffold(  body: Container(  padding: const EdgeInsets.all(20),  child: ListView.builder(  itemCount: students.length,  itemBuilder: (context, index) {  final item = students[index];  return ListTile(  subtitle: Row(children: [  Expanded(child: Text("${item.regno}")),  Expanded(child: Text(item.name)),  Expanded(child: Text("${item.marks}")),  Expanded(child: Text(item.marks >= 50 ? "Pass" : "Fail")),  ]));  })));  }  } |

**Question D1:**

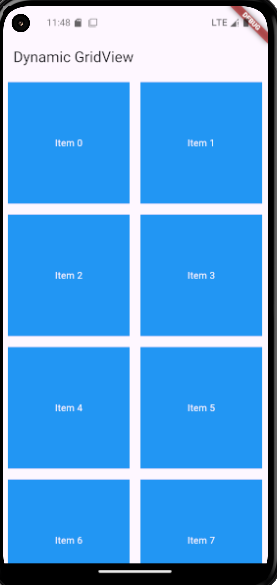
Write code to show the following list using flutter GridView widget

[ {name: ‘Ali’, age: 33, city: ‘Karachi’}, {name: ‘Faisal’, age: 20, city: ‘Lahore’}, {name: ‘Noman’, age: 53, city: ‘Karachi’},]

|  |
| --- |
| /\*  Write code to show the following list using flutter GridView widget  [ {name: ‘Ali’, age: 33, city: ‘Karachi’}, {name: ‘Faisal’, age: 20, city: ‘Lahore’}, {name:  ‘Noman’, age: 53, city: ‘Karachi’},]  \*/  import 'package:flutter/material.dart';  void main() {  runApp(MyApp());  }  class MyApp extends StatelessWidget {  MyApp({super.key});  final List<Map<String, dynamic>> users = [  {'name': 'Ali', 'age': 33, 'city': 'Karachi'},  {'name': 'Faisal', 'age': 20, 'city': 'Lahore'},  {'name': 'Noman', 'age': 53, 'city': 'Karachi'},  ];  @override  Widget build(BuildContext context) {  const title = 'Grid List';  return MaterialApp(  title: title,  home: Scaffold(  appBar: AppBar(  title: const Text(title),  ),  body: GridView.count(  // Create a grid with 2 columns. If you change the scrollDirection to  // horizontal, this produces 2 rows.  crossAxisCount: 2,  // Generate 100 widgets that display their index in the List.  children: List.generate(users.length, (index) {  return Card(  elevation: 4.0,  child: Padding(  padding: const EdgeInsets.all(8.0),  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  Text('Name: ${users[index]['name']}',  style: const TextStyle(  fontSize: 16.0, fontWeight: FontWeight.bold)),  const SizedBox(height: 10.0),  Text('Age: ${users[index]['age']}',  style: const TextStyle(fontSize: 14.0)),  const SizedBox(height: 10.0),  Text('City: ${users[index]['city']}',  style: const TextStyle(fontSize: 14.0)),  ],  ),  ),  );  }),  ),  ),  );  }  } |

Question D1A:

Write a GridView to show by default 2 columns in Portrait layout, and then landscape layout, it should show more items per row, as the following:

 A screenshot of a computer

Description automatically generated

Solution:

|  |
| --- |
| import 'package:flutter/material.dart';  void main() => runApp(MyApp());  class MyApp extends StatelessWidget {    @override    Widget build(BuildContext context) {      return MaterialApp(        home: Scaffold(          appBar: AppBar(title: Text('Dynamic GridView')),          body: MyGridView(),        ),      );    }  }  class MyGridView extends StatelessWidget {    @override    Widget build(BuildContext context) {      return LayoutBuilder(        builder: (context, constraints) {          int crossAxisCount = (constraints.maxWidth / 200).floor();          if (crossAxisCount < 2) {            crossAxisCount = 2; // Ensure at least 2 items in portrait layout          }          return GridView.count(            crossAxisCount: crossAxisCount,            children: List.generate(20, (index) {              return Container(                margin: EdgeInsets.all(8.0),                color: Colors.blue,                child: Center(                  child: Text(                    'Item $index',                    style: TextStyle(color: Colors.white),                  ),                ),              );            }),          );        },      );    }  } |

Note: to make it one column per row at portrait layout, remove these lines:

if (crossAxisCount < 2) {

crossAxisCount = 2; // Ensure at least 2 items in portrait layout

}

**Question C34:**

You have a layout as given in the following.

The capital of Pakistan is *<Text>*

KARACHI

LAHORE

ISLAMABAD

You need to write a “single method” for all the three buttons. The prototype of method is:

function button\_Click(String).

In this method, you need to get the text of the button clicked. If the text is matching with the string “ISLAMABAD”, the <Text> should be assigned value ISLAMABAD, otherwise it remains blank.

|  |
| --- |
| /\*  Question C34:  You have a layout as given in the following.    You need to write a “single method” for all the three buttons. The prototype of method is:  function button\_Click(String).  In this method, you need to get the text of the button clicked. If the text is matching with the string “ISLAMABAD”, the <Text> should be assigned value ISLAMABAD, otherwise it remains blank.  \*/  import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(title: 'Flutter Demo', home: MyApp()));  }  class MyApp extends StatefulWidget {  const MyApp({super.key});  @override  State<MyApp> createState() => \_MyAppState();  }  class \_MyAppState extends State<MyApp> {  String cityInfo = "";  void button\_click(String city) {  debugPrint("CITY $city");  setState(() {  cityInfo = city == "ISLAMABAD" ? "ISLAMABAD" : "";  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  body: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: [  Text("The capital of Pakistan is ${cityInfo}"),  Row(children: [  ElevatedButton(  onPressed: () => button\_click("KARACHI"), child: Text("KARACHI")),  ElevatedButton(  onPressed: () => button\_click("LAHORE"), child: Text("LAHORE")),  ElevatedButton(  onPressed: () => button\_click("ISLAMABAD"),  child: Text("ISLAMABAD")),  ])  ],  ));  }  } |

**Question C30:**

Given the following design:

0

A

B

C

Range

(1 to 3)

Range

(4 to 6)

Range

(7 to 9)

0

0

Click Here

You need to generate a random number from 1 to 9. If the random number is from 1 to 3, increment by one in text box A, if the random number is from 4 to 6, increment by one in text box B, if the random number is between 7 to 9, increment by one in the text box C. The program should stop executing when any of the text boxes value crosses 5.

|  |
| --- |
| import 'dart:math';  import 'package:flutter/material.dart';  void main() => runApp(MyApp());  class MyApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(title: Text('Random Increment Example')),  body: RandomIncrementer(),  ),  );  }  }  class RandomIncrementer extends StatefulWidget {  @override  \_RandomIncrementerState createState() => \_RandomIncrementerState();  }  class \_RandomIncrementerState extends State<RandomIncrementer> {  int valueA = 0;  int valueB = 0;  int valueC = 0;  int target = 10;  bool isRunning = true;  void \_incrementValue() {  if (!isRunning) return;  int randomNumber = Random().nextInt(9) + 1;  setState(() {  if (randomNumber >= 1 && randomNumber <= 3) {  valueA++;  } else if (randomNumber >= 4 && randomNumber <= 6) {  valueB++;  } else if (randomNumber >= 7 && randomNumber <= 9) {  valueC++;  }  if (valueA > target || valueB > target || valueC > target) {  isRunning = false;  }  });  }  @override  Widget build(BuildContext context) {  return Padding(  padding: const EdgeInsets.all(16.0),  child: Column(children: [  Row(  mainAxisAlignment: MainAxisAlignment.spaceEvenly,  children: [  Column(  children: [  Text("A"),  Text('$valueA', style: TextStyle(fontSize: 24.0)),  Text("Range"),  Text("1 to 3")  ],  ),  Column(children: [  Text("B"),  Text('$valueB', style: TextStyle(fontSize: 24.0)),  Text("Range"),  Text("4 to 6")  ]),  Column(  children: [  Text("C"),  Text('$valueC', style: TextStyle(fontSize: 24.0)),  Text("Range"),  Text("7 to 9")  ],  ),  ],  ),  Row(  mainAxisAlignment: MainAxisAlignment.spaceEvenly,  children: [  ElevatedButton(  onPressed: isRunning == true ? \_incrementValue : null,  child: Text('Click Here'),  ),  ],  )  ]),  );  }  } |

**Question C31:**

We want to implement a cricket scoring game machine. It is a competition between 3 three players. Each player has to reach a target score of 10. Each player will play at his turn (when his button is enabled). At a player’s turn, a random number will be generated from 1 to 6, and will be added into the existing score of the player. At one time, the button of one player is enabled who has current turn. When a player reaches 10, his button should be disabled forever, and the competition will continue between remaining two players. When the second player wins, the game will be over, and the final scores and number of turns of first-two should be displayed on the screen as Match Summary (see below). The winner is the one with the maximum current score. If the score of two players is same, the player with lesser number of turns should be the winner.

Target: 10

Target: 10

Target: 10

Current Score: 0

Outcome: 0

Current Score: 0

Outcome: 0

Current Score: 0

Outcome: 0

Player 1

Player 2

Player 3

**Player 1**

**Player 2**

**Player 3**

Click a button to a Generate random number from 1 to 6

Turn Number: 0

Turn Number: 0

Turn Number: 0

**Here is the first case of match summary, where the highest score player is the winner.**

**Match Summary**

Position 1 : Player 2, Score 9, No of Turns: 4

Position 2: Player 1, Score: 5, No of Turns: 2,

**Here is the second case of match summary, where both players have the same score, and the player with the lesser number of turns is declared the winner.**

**Match Summary**

Position 1 : Player 2, Score 11, No of Turns: 4

Position 2: Player 1, Score: 11, No of Turns: 6,

(In above, we are first sorting on base of score, and then on base of number of turns).

|  |
| --- |
| import 'dart:math';  import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return MaterialApp(  title: 'Scoring Game Machine',  theme: ThemeData(  primarySwatch: Colors.blue,  ),  home: const GameScreen(),  );  }  }  class Player {  int? playerNo;  int? currentScore;  int? turnNumber;  bool? buttonStatus;  bool? enabled;  Player(  {this.playerNo,  this.currentScore,  this.turnNumber,  this.buttonStatus,  this.enabled});  }  class GameScreen extends StatefulWidget {  const GameScreen({super.key});  @override  State<GameScreen> createState() => \_GameScreenState();  }  class \_GameScreenState extends State<GameScreen> {  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Scoring Game'),  ),  body: const Center(  child: PlayerWidget(),  ),  );  }  }  class PlayerWidget extends StatefulWidget {  const PlayerWidget({super.key});  @override  State<PlayerWidget> createState() => \_PlayerWidgetState();  }  class \_PlayerWidgetState extends State<PlayerWidget> {  final int targetScore = 10;  bool showsummary = false;  int? turnoutcome = 0;  List<Map<String, dynamic>> matchSummary = [];  List<Player> players = [  Player(  playerNo: 1,  currentScore: 0,  turnNumber: 0,  buttonStatus: true,  enabled: true),  Player(  playerNo: 2,  currentScore: 0,  turnNumber: 0,  buttonStatus: false,  enabled: true),  Player(  playerNo: 3,  currentScore: 0,  turnNumber: 0,  buttonStatus: false,  enabled: true),  ];  void playerTurn(int player) {  int score = Random().nextInt(6) + 1;  setState(() {  turnoutcome = score;  if (player == 0) {  players[0].buttonStatus = false;  players[0].currentScore = players[0].currentScore! + score;  players[0].turnNumber = players[0].turnNumber! + 1;  players[1].buttonStatus = true;  if (players[0].currentScore! >= targetScore) {  players[0].enabled = false;  matchSummary.add({  "Player": "1",  "Score": "${players[0].currentScore}",  "Turns": "${players[0].turnNumber}"  });  }  if (players[1].enabled == false) {  players[2].buttonStatus = true;  }  } else if (player == 1) {  players[0].buttonStatus = false;  players[1].buttonStatus = false;  players[2].buttonStatus = true;  players[1].currentScore = players[1].currentScore! + score;  players[1].turnNumber = players[1].turnNumber! + 1;  if (players[1].currentScore! >= targetScore) {  players[1].enabled = false;  matchSummary.add({  "Player": "2",  "Score": "${players[1].currentScore}",  "Turns": "${players[1].turnNumber}"  });  }  if (players[2].enabled == false) {  players[0].buttonStatus = true;  }  } else if (player == 2) {  players[0].buttonStatus = true;  players[1].buttonStatus = false;  players[2].buttonStatus = false;  players[2].currentScore = players[2].currentScore! + score;  players[2].turnNumber = players[2].turnNumber! + 1;  if (players[2].currentScore! >= targetScore) {  players[2].enabled = false;  matchSummary.add({  "Player": "3",  "Score": "${players[2].currentScore}",  "Turns": "${players[2].turnNumber}"  });  }  if (players[0].enabled == false) {  players[1].buttonStatus = true;  }  }  int count = 0;  for (int i = 0; i < players.length; i++) {  if (players[i].enabled == false) {  count = count + 1;  }  }  if (count == players.length - 1) {  showsummary = true;  // First give priority to sort in terms of "score" in descending order.  // and then give priority to sort in terms of "turns" in ascending order  matchSummary.sort((a, b) {  int ans = b['Score'].compareTo(a['Score']);  if (ans == 0) {  return a['Turns'].compareTo(b['Turns']);  } else {  return ans;  }  });  players[0].buttonStatus = false;  players[1].buttonStatus = false;  players[2].buttonStatus = false;  }  });  }  @override  Widget build(BuildContext context) {  return Column(  children: [  Row(  mainAxisAlignment: MainAxisAlignment.spaceEvenly,  children: [  Column(  children: [  Text("Player: ${players[0].playerNo}"),  Text("Outcome: $turnoutcome"),  Text("Current Score: ${players[0].currentScore}"),  Text("Target: $targetScore"),  Text("Turn Numbrer: ${players[0].turnNumber}"),  ElevatedButton(  onPressed: players[0].buttonStatus == true &&  players[0].enabled == true  ? () => playerTurn(0)  : null,  style:  ElevatedButton.styleFrom(fixedSize: const Size(100, 50)),  child: Text('Player ${players[0].playerNo}'),  ),  ],  ),  Column(  children: [  Text("Player: ${players[1].playerNo}"),  Text("Outcome: $turnoutcome"),  Text("Current Score: ${players[1].currentScore}"),  Text("Target: $targetScore"),  Text("Turn Numbrer: ${players[1].turnNumber}"),  ElevatedButton(  onPressed: players[1].buttonStatus == true &&  players[1].enabled == true  ? () => playerTurn(1)  : null,  style:  ElevatedButton.styleFrom(fixedSize: const Size(100, 50)),  child: Text('Player ${players[1].playerNo}'),  ),  ],  ),  Column(  children: [  Text("Player: ${players[2].playerNo}"),  Text("Outcome: $turnoutcome"),  Text("Current Score: ${players[2].currentScore}"),  Text("Target: $targetScore"),  Text("Turn Numbrer: ${players[2].turnNumber}"),  ElevatedButton(  onPressed: players[2].buttonStatus == true &&  players[2].enabled == true  ? () => playerTurn(2)  : null,  style:  ElevatedButton.styleFrom(fixedSize: const Size(100, 50)),  child: Text('Player ${players[2].playerNo}'),  ),  ],  ),  ],  ),  showsummary == true  ? MatchSummary(matchSummary: matchSummary)  : const SizedBox()  ],  );  }  }  class MatchSummary extends StatelessWidget {  final List<Map<String, dynamic>> matchSummary;  const MatchSummary({super.key, required this.matchSummary});  @override  Widget build(BuildContext context) {  return Column(  children: [  const SizedBox(height: 20),  const Text('Match Summary'),  for (int i = 0; i < matchSummary.length; i++)  Text(  "Position: ${i + 1}: Player: ${matchSummary[i]['Player']} Score: ${matchSummary[i]['Score']} Turns: ${matchSummary[i]['Turns']}"),  ],  );  }  } |

**Question C32:**

Suppose you want to build a game in which a random value is generated representing fire or wood or water on button click. Another random value representing fire or wood or water is generated for computer on button click. The winner is decided on the following priority: Fire > Wood, Wood > Water, Water > Fire. The player that gets the higher priority value is the winner. If both get same priority value, it is draw. Write the flutter design and code.

fire

Generate user value

**User value**

wood

**Computer value**

**Winner**

user

Generate computer value

|  |
| --- |
| import 'dart:math';  import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(home: FireWoodWater()));  }  class FireWoodWater extends StatefulWidget {  const FireWoodWater({super.key});  @override  State<FireWoodWater> createState() => \_FireWoodWaterState();  }  class \_FireWoodWaterState extends State<FireWoodWater> {  List<String> list = ["Fire", "Wood", "Water"];  int? compRand;  String compVal = "";  bool compbtn = true;  int? userRand;  String userVal = "";  bool userbtn = true;  String winner = "";  void generateTurn(String player) {  setState(() {  if (player == "computer") {  compRand = Random().nextInt(list.length);  compVal = list[compRand!];  compbtn = false;  } else {  userRand = Random().nextInt(list.length);  userVal = list[userRand!];  userbtn = false;  }  if (userVal != "" && compVal != "") {  if (userVal == "Fire" && compVal == "Fire") {  winner = "Draw";  } else if (userVal == "Fire" && compVal == "Wood") {  winner = "User";  } else if (userVal == "Fire" && compVal == "Water") {  winner = "Computer";  } else if (userVal == "Wood" && compVal == "Fire") {  winner = "Computer";  } else if (userVal == "Wood" && compVal == "Wood") {  winner = "Draw";  } else if (userVal == "Wood" && compVal == "Water") {  winner = "User";  }  if (userVal == "Water" && compVal == "Fire") {  winner = "User";  } else if (userVal == "Water" && compVal == "Wood") {  winner = "Computer";  } else if (userVal == "Water" && compVal == "Water") {  winner = "Draw";  }  }  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(title: Center(child: Text('Fire, wood, water'))),  body: Padding(  padding: EdgeInsets.only(left: 50),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Row(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Expanded(  child: Text("User Value"),  ),  SizedBox(width: 20), // Space between columns  Expanded(  child: Text(userVal),  ),  ],  ),  Row(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Expanded(  child: Text("Computer Value"),  ),  SizedBox(width: 20), // Space between columns  Expanded(  child: Text(compVal),  ),  ],  ),  Row(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Expanded(  child: Text("Winner"),  ),  SizedBox(width: 20), // Space between columns  Expanded(  child: Text(winner!),  ),  ],  ),  SizedBox(height: 15),  Row(  children: [  Container(  width: 130,  height: 70,  child: ElevatedButton(  child: Text("Generate user value"),  onPressed:  userbtn == true ? () => generateTurn("user") : null,  ),  ),  SizedBox(width: 20), // Space between columns  Container(  width: 130,  height: 70,  child: ElevatedButton(  child: Text("Generate computer value"),  onPressed:  compbtn == true ? () => generateTurn("computer") : null,  ),  ),  SizedBox(width: 30), // Space between columns  ],  ),  ],  ),  ),  );  }  } |

**Question C38:**

Suppose you have an <Text> field and two buttons. The first button is labeled as BLUE and the second button is labeled as GREEN. When the BLUE button is clicked, the color of text in <Text> should changed to BLUE, and when GREEN button is clicked, the color of text in <Text> should change to GREEN..

|  |
| --- |
|  |

**Question C39:**

RED

GREEN

**BLUE**

BLUE is clicked

Suppose you have a layout like the above. In the example, the blue button is clicked, and its text size is increased, and text color is changed to black.

The buttons are created by using array of color names, and the text in the buttons is shown in upper case. When a button is clicked, the color of the text below is changed and the name of color is shown as shown in the above example. Moreover, the button that is clicked has font weight changed to bold and font size increased to indicate which button is currently clicked.

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(home: MyApp()));  }  class MyApp extends StatefulWidget {  const MyApp({super.key});  @override  State<MyApp> createState() => \_MyAppState();  }  class \_MyAppState extends State<MyApp> {  Color currentColor = Colors.red;  String currentColorName = "RED";  Color bgcolorRed = Colors.red;  Color bgcolorGreen = Colors.green;  Color bgcolorBlue = Colors.blue;  Color fgcolorRed = Colors.black;  Color fgcolorGreen = Colors.white;  Color fgcolorBlue = Colors.white;  void changeColor(String colorName) {  setState(() {  if (colorName.compareTo("RED") == 0) {  fgcolorRed = Colors.black;  fgcolorGreen = Colors.white;  fgcolorBlue = Colors.white;  currentColor = Colors.red;  currentColorName = "RED";  } else if (colorName.compareTo("GREEN") == 0) {  fgcolorRed = Colors.white;  fgcolorGreen = Colors.black;  fgcolorBlue = Colors.white;  currentColor = Colors.green;  currentColorName = "GREEN";  } else if (colorName.compareTo("BLUE") == 0) {  fgcolorRed = Colors.white;  fgcolorGreen = Colors.white;  fgcolorBlue = Colors.black;  currentColor = Colors.blue;  currentColorName = "BLUE";  }  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(title: Center(child: Text(''))),  body: Column(  mainAxisAlignment: MainAxisAlignment.start,  children: [  Row(  mainAxisAlignment: MainAxisAlignment.center,  children: [  ElevatedButton(  onPressed: () => changeColor("RED"),  style: ElevatedButton.styleFrom(  backgroundColor: bgcolorRed, // Background color  foregroundColor: fgcolorRed, // Text color  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(12), // Rounded corners  side: BorderSide(  color: const Color.fromARGB(255, 0, 0, 0), // Border color  width: 1, // Border width  ),  ),  minimumSize: Size(80, 40),  padding: EdgeInsets.symmetric(  horizontal: 16, vertical: 8), // Button padding  ),  child: Text(  "RED",  style: TextStyle(fontSize: 18),  ),  ),  SizedBox(width: 20), // Space between columns  ElevatedButton(  onPressed: () => changeColor("GREEN"),  style: ElevatedButton.styleFrom(  backgroundColor: bgcolorGreen, // Background color  foregroundColor: fgcolorGreen, // Text color  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(12), // Rounded corners  side: BorderSide(  color: const Color.fromARGB(255, 0, 0, 0), // Border color  width: 1, // Border width  ),  ),  minimumSize: Size(80, 40),  padding: EdgeInsets.symmetric(  horizontal: 16, vertical: 8), // Button padding  ),  child: Text(  "GREEN",  style: TextStyle(fontSize: 18),  ),  ),  SizedBox(width: 20), // Space between columns  ElevatedButton(  onPressed: () => changeColor("BLUE"),  style: ElevatedButton.styleFrom(  backgroundColor: bgcolorBlue, // Background color  foregroundColor: fgcolorBlue, // Text color  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(12), // Rounded corners  side: BorderSide(  color: const Color.fromARGB(255, 0, 0, 0), // Border color  width: 1, // Border width  ),  ),  minimumSize: Size(80, 40), // Set the fixed width and height  padding: EdgeInsets.symmetric(  horizontal: 16, vertical: 8), // Button padding  ),  child: Text("BLUE",  style: TextStyle(  fontWeight: FontWeight.bold,  fontSize: 18,  )),  ),  ],  ),  SizedBox(height: 20),  Container(  width: 200,  padding: EdgeInsets.all(8.0),  child: Center(  child: Text(  "$currentColorName is clicked",  style: TextStyle(color: Colors.white, fontSize: 20),  )),  color: currentColor)  ],  ),  );  }  } |
|  |

**Question C11\*:**

You need to develop a snakes and ladders game, as shown below:

A screenshot of a game

Description automatically generated

The snakes & ladders board can be downloaded from:

<http://hancockmcdonald.com/sites/hancockmcdonald.com/files/file-downloads/SnakesLaddersBoard.jpg>

The player image can be downloaded from:

http://clipart-library.com/images/kT85jnpXc.png

(NOTE: You may change background board or player images).

The player will promote or demote based on arrival on ladder or snake head respectively. Implement complete logic. The game should end when a player crosses last digit.

|  |
| --- |
| https://github.com/osmankhalid2005/flutter/blob/963b4fa7a08a4bb479c01e1928fe2d1c945e86e9/problem\_set\_solutions/C11.zip |

**Question C37:**

The following is the game board of a Tic Tac Toe game.

|  |  |  |
| --- | --- | --- |
| 6 | 7 | 8 |
| 3 | 4 | 5 |
| 0 | 1 | 2 |

The cells are numbered from 0 to 8 (making up a total of 9 cells). You need to develop a game to be played between two computer players A and B. Player A’s turn takes place when a method: playerATurn() is called, whereas for player B, the method playerBTurn() is called. During each turn, a random number is selected from 0 to 8 to represent the array index of the player’s List, and the player’s value (either zero ‘O’ or cross ‘X’) is inserted into that index. However, if the generated random number is already present in any of the player’s visitedCells list, the player’s turn will be skipped, and next player will take the turn. The following is the winning criteria of a player. For example, the player A wins if:

visitedCellsA[0].value == visitedCellsA[1].value == visitedCellsA[2].value, Or

visitedCellsA[3].value == visitedCellsA[4].value == visitedCellsA[5].value, Or

visitedCellsA[6].value == visitedCellsA[7].value == visitedCellsA[8].value.

Similarly, the player B wins if:

visitedCellsB[0].value == visitedCellsB[1].value == visitedCellsB[2].value, Or

visitedCellsB[3].value == visitedCellsB[4].value == visitedCellsB[5].value, Or

visitedCellsB[6].value == visitedCellsB[7].value == visitedCellsB[8].value.

Write the functions playerATurn() and playerBTurn(). Display about the winning player or draw match.

|  |
| --- |
|  |

**Question C18\*:**

You need to create a mobile phone portrait layout:



|  |
| --- |
| import 'package:flutter/material.dart';  void main() => runApp(MyApp());  class MyApp extends StatelessWidget {  MyApp({super.key});    @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(  title: Text('Form Example'),  ),  body: MyForm(),  ),  );  }  }  class MyForm extends StatefulWidget {  @override  \_MyFormState createState() => \_MyFormState();  }  class \_MyFormState extends State<MyForm> {  final \_formKey = GlobalKey<FormState>();  final TextEditingController \_emailController = TextEditingController();  final TextEditingController \_nameController = TextEditingController();  final TextEditingController \_addressController = TextEditingController();  String \_gender = '';  String \_country = '';  List<String> \_subjects = [];  List<String> \_skills = [];  List<String> \_countries = ['USA', 'Canada', 'UK', 'Australia', 'Pakistan'];  List<String> \_availableSubjects = ['Phy', 'Chem', 'Bio'];  List<String> \_availableSkills = ['C++', 'Java', 'Javascript', 'C#'];  @override  void dispose() {  \_emailController.dispose();  \_nameController.dispose();  \_addressController.dispose();  super.dispose();  }  @override  Widget build(BuildContext context) {  return Padding(  padding: const EdgeInsets.all(16.0),  child: Form(  key: \_formKey,  child: ListView(  children: <Widget>[  TextFormField(  controller: \_emailController,  decoration:  InputDecoration(labelText: 'Email', hintText: '30 chars max'),  maxLength: 30,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your email';  }  return null;  },  ),  TextFormField(  controller: \_nameController,  decoration:  InputDecoration(labelText: 'Name', hintText: '50 chars max'),  maxLength: 50,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your name';  }  return null;  },  ),  Text("\nGender"),  Row(  children: <Widget>[  Expanded(  child: RadioListTile<String>(  title: const Text('Male'),  value: 'Male',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  Expanded(  child: RadioListTile<String>(  title: const Text('Female'),  value: 'Female',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  ],  ),  DropdownButtonFormField<String>(  decoration: InputDecoration(labelText: 'Country'),  items: \_countries.map((String country) {  return DropdownMenuItem<String>(  value: country,  child: Text(country),  );  }).toList(),  onChanged: (value) {  setState(() {  \_country = value!;  });  },  validator: (value) {  if (value == null) {  return 'Please select a country';  }  return null;  },  ),  Text("\nSubjects"),  Wrap(  spacing: 10.0,  runSpacing: 10.0,  children: \_availableSubjects.map((subject) {  return Row(  mainAxisSize: MainAxisSize.min,  children: <Widget>[  Checkbox(  value: \_subjects.contains(subject),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_subjects.add(subject);  } else {  \_subjects.remove(subject);  }  });  },  ),  Text(subject),  ],  );  }).toList(),  ),  Text('Skills'),  Container(  decoration: BoxDecoration(  border: Border.all(),  borderRadius: BorderRadius.circular(5.0),  ),  child: Column(  children: \_availableSkills.map((skill) {  return CheckboxListTile(  title: Text(skill),  value: \_skills.contains(skill),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_skills.add(skill);  } else {  \_skills.remove(skill);  }  });  },  );  }).toList(),  ),  ),  TextFormField(  controller: \_addressController,  decoration: InputDecoration(labelText: 'Address'),  maxLines: 3,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your address';  }  return null;  },  ),  SizedBox(height: 20),  ElevatedButton(  onPressed: () {  debugPrint("HERE: ${\_formKey.currentState!.validate()}");  if (\_formKey.currentState!.validate()) {  // Handle form submission  debugPrint('Email: ${\_emailController.text}');  debugPrint('Name: ${\_nameController.text}');  debugPrint('Gender: $\_gender');  debugPrint('Country: $\_country');  debugPrint('Subjects: $\_subjects');  debugPrint('Skills: $\_skills');  debugPrint('Address: ${\_addressController.text}');  }  },  child: Text('Submit'),  ),  ],  ),  ),  );  }  } |

**Question C36\*:**

When the application in C18 opens in tablet layout (landscape), it should be shown like this:

Submit

30 chars max

Email

50 chars max

Name

Gender

Male

Female

Country

Subjects

Phy

Chem

Address

Bio

C++

Java

Javascript

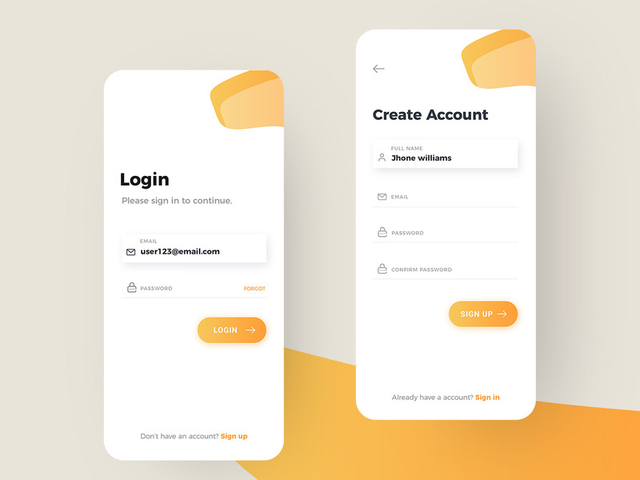
C#

Skills

|  |
| --- |
| import 'package:flutter/material.dart';  void main() => runApp(MyApp());  class MyApp extends StatelessWidget {  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(  title: Text('Form Example'),  ),  body: MyForm(),  ),  );  }  }  class MyForm extends StatefulWidget {  @override  \_MyFormState createState() => \_MyFormState();  }  class \_MyFormState extends State<MyForm> {  final \_formKey = GlobalKey<FormState>();  final TextEditingController \_emailController = TextEditingController();  final TextEditingController \_nameController = TextEditingController();  final TextEditingController \_addressController = TextEditingController();  String \_gender = '';  String \_country = '';  List<String> \_subjects = [];  List<String> \_skills = [];  List<String> \_countries = ['USA', 'Canada', 'UK', 'Australia', 'Pakistan'];  List<String> \_availableSubjects = ['Phy', 'Chem', 'Bio'];  List<String> \_availableSkills = ['C++', 'Java', 'Javascript', 'C#'];  @override  void dispose() {  \_emailController.dispose();  \_nameController.dispose();  \_addressController.dispose();  super.dispose();  }  @override  Widget build(BuildContext context) {  return LayoutBuilder(  builder: (BuildContext context, BoxConstraints constraints) {  bool isWideScreen = constraints.maxWidth > 600;  return Padding(  padding: const EdgeInsets.all(16.0),  child: Form(  key: \_formKey,  child: isWideScreen  ? Row(  children: <Widget>[  Expanded(  child: SingleChildScrollView(  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  TextFormField(  controller: \_emailController,  decoration: InputDecoration(  labelText: 'Email',  hintText: '30 chars max'),  maxLength: 30,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your email';  }  return null;  },  ),  Text("\nGender"),  Row(  children: <Widget>[  Expanded(  child: RadioListTile<String>(  title: const Text('Male'),  value: 'Male',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  Expanded(  child: RadioListTile<String>(  title: const Text('Female'),  value: 'Female',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  ],  ),  Text("\nSubjects"),  Wrap(  //spacing: 10.0,  //runSpacing: 10.0,  children: \_availableSubjects.map((subject) {  return Row(  mainAxisSize: MainAxisSize.min,  children: <Widget>[  Checkbox(  value: \_subjects.contains(subject),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_subjects.add(subject);  } else {  \_subjects.remove(subject);  }  });  },  ),  Text(subject),  ],  );  }).toList(),  ),  Text('Skills'),  Container(  decoration: BoxDecoration(  border: Border.all(),  borderRadius: BorderRadius.circular(5.0),  ),  child: Column(  children: \_availableSkills.map((skill) {  return CheckboxListTile(  title: Text(skill),  value: \_skills.contains(skill),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_skills.add(skill);  } else {  \_skills.remove(skill);  }  });  },  );  }).toList(),  ),  ),  ],  ),  ),  ),  SizedBox(width: 16.0),  Expanded(  child: SingleChildScrollView(  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  TextFormField(  controller: \_nameController,  decoration: InputDecoration(  labelText: 'Name',  hintText: '50 chars max'),  maxLength: 50,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your name';  }  return null;  },  ),  DropdownButtonFormField<String>(  decoration:  InputDecoration(labelText: 'Country'),  items: \_countries.map((String country) {  return DropdownMenuItem<String>(  value: country,  child: Text(country),  );  }).toList(),  onChanged: (value) {  setState(() {  \_country = value!;  });  },  validator: (value) {  if (value == null) {  return 'Please select a country';  }  return null;  },  ),  TextFormField(  controller: \_addressController,  decoration:  InputDecoration(labelText: 'Address'),  maxLines: 3,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your address';  }  return null;  },  ),  SizedBox(height: 20),  ElevatedButton(  onPressed: () {  if (\_formKey.currentState!.validate()) {  // Handle form submission  debugPrint(  'Email: ${\_emailController.text}');  debugPrint('Name: ${\_nameController.text}');  debugPrint('Gender: $\_gender');  debugPrint('Country: $\_country');  debugPrint('Subjects: $\_subjects');  debugPrint('Skills: $\_skills');  debugPrint(  'Address: ${\_addressController.text}');  }  },  child: Text('Submit'),  ),  ],  ),  ),  ),  ],  )  : ListView(  children: <Widget>[  TextFormField(  controller: \_emailController,  decoration: InputDecoration(  labelText: 'Email', hintText: '30 chars max'),  maxLength: 30,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your email';  }  return null;  },  ),  TextFormField(  controller: \_nameController,  decoration: InputDecoration(  labelText: 'Name', hintText: '50 chars max'),  maxLength: 50,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your name';  }  return null;  },  ),  Text("\nGender"),  Row(  children: <Widget>[  Expanded(  child: RadioListTile<String>(  title: const Text('Male'),  value: 'Male',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  Expanded(  child: RadioListTile<String>(  title: const Text('Female'),  value: 'Female',  groupValue: \_gender,  onChanged: (value) {  setState(() {  \_gender = value!;  });  },  ),  ),  ],  ),  DropdownButtonFormField<String>(  decoration: InputDecoration(labelText: 'Country'),  items: \_countries.map((String country) {  return DropdownMenuItem<String>(  value: country,  child: Text(country),  );  }).toList(),  onChanged: (value) {  setState(() {  \_country = value!;  });  },  validator: (value) {  if (value == null) {  return 'Please select a country';  }  return null;  },  ),  Text("\nSubjects"),  Wrap(  spacing: 10.0,  runSpacing: 10.0,  children: \_availableSubjects.map((subject) {  return Row(  mainAxisSize: MainAxisSize.min,  children: <Widget>[  Checkbox(  value: \_subjects.contains(subject),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_subjects.add(subject);  } else {  \_subjects.remove(subject);  }  });  },  ),  Text(subject),  ],  );  }).toList(),  ),  Text('Skills'),  Container(  decoration: BoxDecoration(  border: Border.all(),  borderRadius: BorderRadius.circular(5.0),  ),  child: Column(  children: \_availableSkills.map((skill) {  return CheckboxListTile(  title: Text(skill),  value: \_skills.contains(skill),  onChanged: (bool? value) {  setState(() {  if (value!) {  \_skills.add(skill);  } else {  \_skills.remove(skill);  }  });  },  );  }).toList(),  ),  ),  TextFormField(  controller: \_addressController,  decoration: InputDecoration(labelText: 'Address'),  maxLines: 3,  validator: (value) {  if (value!.isEmpty) {  return 'Please enter your address';  }  return null;  },  ),  SizedBox(height: 20),  ElevatedButton(  onPressed: () {  if (\_formKey.currentState!.validate()) {  // Handle form submission  debugPrint('Email: ${\_emailController.text}');  debugPrint('Name: ${\_nameController.text}');  debugPrint('Gender: $\_gender');  debugPrint('Country: $\_country');  debugPrint('Subjects: $\_subjects');  debugPrint('Skills: $\_skills');  debugPrint('Address: ${\_addressController.text}');  }  },  child: Text('Submit'),  ),  ],  ),  ),  );  },  );  }  } |

**Question C41:**

Create a login and a registration page, with proper flutter styling. Here is a sample, but your work can be different and better than this.



|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(  home: LoginPage(),  ));  }  class LoginPage extends StatelessWidget {  const LoginPage({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  body: SingleChildScrollView(  child: Padding(  padding: const EdgeInsets.symmetric(horizontal: 20.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const SizedBox(height: 50), // Spacing from top  IconButton(  icon: const Icon(Icons.arrow\_back),  onPressed: () {},  ),  const SizedBox(height: 20),  const Text(  'Login',  style: TextStyle(  fontSize: 30,  fontWeight: FontWeight.bold,  ),  ),  const SizedBox(height: 5),  const Text(  'Please sign in to continue.',  style: TextStyle(  fontSize: 16,  color: Colors.grey,  ),  ),  const SizedBox(height: 20),  \_buildTextField(  context,  icon: Icons.email,  label: 'Email',  hintText: 'user123@email.com',  ),  const SizedBox(height: 20),  \_buildPasswordField(context),  const SizedBox(height: 30),  Center(  child: ElevatedButton(  style: ElevatedButton.styleFrom(  backgroundColor: Colors.orange,  padding: const EdgeInsets.symmetric(  horizontal: 100, vertical: 20),  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(20),  ),  ),  onPressed: () {},  child: const Text(  'LOGIN',  style: TextStyle(  color: Colors.white,  fontSize: 16,  ),  ),  ),  ),  const SizedBox(height: 20),  Center(  child: GestureDetector(  onTap: () {  // Navigate to Sign Up page  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const SignUpPage()),  );  },  child: const Text(  "Don't have an account? Sign up",  style: TextStyle(  color: Colors.orange,  fontSize: 16,  ),  ),  ),  ),  ],  ),  ),  ),  );  }  Widget \_buildTextField(BuildContext context,  {required IconData icon, required String label, String? hintText}) {  return TextField(  decoration: InputDecoration(  prefixIcon: Icon(icon),  labelText: label,  hintText: hintText,  enabledBorder: const UnderlineInputBorder(  borderSide: BorderSide(color: Colors.grey),  ),  focusedBorder: OutlineInputBorder(  borderRadius: BorderRadius.circular(10),  borderSide: const BorderSide(color: Colors.orange),  ),  fillColor: Colors.white,  filled: true,  ),  );  }  Widget \_buildPasswordField(BuildContext context) {  return Row(  children: [  Expanded(  child: TextField(  obscureText: true,  decoration: InputDecoration(  prefixIcon: const Icon(Icons.lock),  labelText: 'Password',  enabledBorder: const UnderlineInputBorder(  borderSide: BorderSide(color: Colors.grey),  ),  focusedBorder: OutlineInputBorder(  borderRadius: BorderRadius.circular(10),  borderSide: const BorderSide(color: Colors.orange),  ),  fillColor: Colors.white,  filled: true,  ),  ),  ),  TextButton(  onPressed: () {  // Forgot password action  },  child: const Text(  'FORGOT',  style: TextStyle(  color: Colors.orange,  fontWeight: FontWeight.bold,  ),  ),  ),  ],  );  }  }  class SignUpPage extends StatelessWidget {  const SignUpPage({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  body: SingleChildScrollView(  child: Padding(  padding: const EdgeInsets.symmetric(horizontal: 20.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const SizedBox(height: 50), // Spacing from top  IconButton(  icon: const Icon(Icons.arrow\_back),  onPressed: () {},  ),  const SizedBox(height: 20),  const Text(  'Create Account',  style: TextStyle(  fontSize: 30,  fontWeight: FontWeight.bold,  ),  ),  const SizedBox(height: 20),  \_buildTextField(  context,  icon: Icons.person,  label: 'Full Name',  ),  const SizedBox(height: 20),  \_buildTextField(  context,  icon: Icons.email,  label: 'Email',  ),  const SizedBox(height: 20),  \_buildTextField(  context,  icon: Icons.lock,  label: 'Password',  obscureText: true,  ),  const SizedBox(height: 20),  \_buildTextField(  context,  icon: Icons.lock,  label: 'Confirm Password',  obscureText: true,  ),  const SizedBox(height: 30),  Center(  child: ElevatedButton(  style: ElevatedButton.styleFrom(  backgroundColor: Colors.orange,  padding: const EdgeInsets.symmetric(  horizontal: 100, vertical: 20),  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(20),  ),  ),  onPressed: () {},  child: const Text(  'SIGN UP',  style: TextStyle(  color: Colors.white,  fontSize: 16,  ),  ),  ),  ),  const SizedBox(height: 20),  Center(  child: GestureDetector(  onTap: () {  // Navigate to Sign In page  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const LoginPage()),  );  },  child: const Text(  'Already have an account? Sign in',  style: TextStyle(  color: Colors.orange,  fontSize: 16,  ),  ),  ),  ),  ],  ),  ),  ),  );  }  Widget \_buildTextField(BuildContext context,  {required IconData icon,  required String label,  bool obscureText = false}) {  return TextField(  obscureText: obscureText,  decoration: InputDecoration(  prefixIcon: Icon(icon),  labelText: label,  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(10),  ),  filled: true,  fillColor: Colors.white,  ),  );  }  } |

**Question C40:**

Write the following layout.

E-mail

Name

Country

City

Address

**SUBMIT**

When the screen size is reduced, the layout should be changed to:

E-mail

Name

Country

City

Address

SUBMIT

Note that labels are showing “above” the text boxes.

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return MaterialApp(  home: Scaffold(  appBar: AppBar(title: const Text('Responsive Form')),  body: const ResponsiveForm(),  ),  );  }  }  class ResponsiveForm extends StatefulWidget {  const ResponsiveForm({super.key});  @override  \_ResponsiveFormState createState() => \_ResponsiveFormState();  }  class \_ResponsiveFormState extends State<ResponsiveForm> {  final TextEditingController emailController = TextEditingController();  final TextEditingController nameController = TextEditingController();  final TextEditingController countryController = TextEditingController();  final TextEditingController cityController = TextEditingController();  final TextEditingController addressController = TextEditingController();  void \_submit() {  debugPrint('E-mail: ${emailController.text}');  debugPrint('Name: ${nameController.text}');  debugPrint('Country: ${countryController.text}');  debugPrint('City: ${cityController.text}');  debugPrint('Address: ${addressController.text}');  }  @override  void dispose() {  emailController.dispose();  nameController.dispose();  countryController.dispose();  cityController.dispose();  addressController.dispose();  super.dispose();  }  @override  Widget build(BuildContext context) {  return LayoutBuilder(  builder: (context, constraints) {  if (constraints.maxWidth > 600) {  // Tablet (landscape) layout  return Padding(  padding: const EdgeInsets.all(16.0),  child: GridView.count(  crossAxisCount: 2,  crossAxisSpacing: 16.0,  mainAxisSpacing: 16.0,  childAspectRatio: 4.0,  children: [  TextField(  controller: emailController,  decoration: const InputDecoration(labelText: 'E-mail'),  ),  TextField(  controller: nameController,  decoration: const InputDecoration(labelText: 'Name'),  ),  TextField(  controller: countryController,  decoration: const InputDecoration(labelText: 'Country'),  ),  TextField(  controller: cityController,  decoration: const InputDecoration(labelText: 'City'),  ),  TextField(  controller: addressController,  decoration: const InputDecoration(labelText: 'Address'),  maxLines: 3,  ),  const SizedBox(), // Empty space to balance the grid  ElevatedButton(  onPressed: \_submit,  child: const Text('SUBMIT'),  ),  ],  ),  );  } else {  // Phone (portrait) layout  return Padding(  padding: const EdgeInsets.all(16.0),  child: SingleChildScrollView(  child: Column(  crossAxisAlignment: CrossAxisAlignment.stretch,  children: [  TextField(  controller: emailController,  decoration: const InputDecoration(labelText: 'E-mail'),  ),  const SizedBox(height: 16.0),  TextField(  controller: nameController,  decoration: const InputDecoration(labelText: 'Name'),  ),  const SizedBox(height: 16.0),  TextField(  controller: countryController,  decoration: const InputDecoration(labelText: 'Country'),  ),  const SizedBox(height: 16.0),  TextField(  controller: cityController,  decoration: const InputDecoration(labelText: 'City'),  ),  const SizedBox(height: 16.0),  TextField(  controller: addressController,  decoration: const InputDecoration(labelText: 'Address'),  maxLines: 3,  ),  const SizedBox(height: 16.0),  ElevatedButton(  onPressed: \_submit,  child: const Text('SUBMIT'),  ),  ],  ),  ),  );  }  },  );  }  } |

**Question C12\*:**

**The following is the website view of Standford university’s computer science department.**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**You need to re-design the above view in mobile layout as follows:**

**You may need to use scrolling.**

**A screenshot of a cell phone

Description automatically generated**

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const StanfordApp());  }  class StanfordApp extends StatelessWidget {  const StanfordApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: StanfordHomePage(),  debugShowCheckedModeBanner: false,  );  }  }  class StanfordHomePage extends StatelessWidget {  const StanfordHomePage({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text(  'Stanford University',  style: TextStyle(color: Colors.white),  ),  backgroundColor: const Color.fromARGB(255, 109, 1, 1),  actions: [  TextButton(  onPressed: () {},  child: TextButton(  onPressed: () {},  child: const Text('CSID Login',  style: TextStyle(color: Colors.white)),  ),  ),  ],  ),  body: SingleChildScrollView(  child:  Column(crossAxisAlignment: CrossAxisAlignment.stretch, children: [  const BannerText(),  const MenuAndSearchBar(),  const SliderScreen(),  Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const SectionHeading(heading: "News"),  const SizedBox(height: 20),  \_buildNewsSection(),  const SizedBox(height: 16),  const SectionHeading(heading: "Events"),  const SizedBox(height: 16),  \_buildEventsSection(),  ])),  Container(  height: 20,  color: const Color.fromARGB(255, 233, 236, 221),  ),  Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const SizedBox(height: 16),  \_buildContactInfoSection(),  ])),  Container(  height: 40,  color: Colors.red[900]!,  ),  Container(  height: 15,  decoration: BoxDecoration(  gradient: LinearGradient(  begin: Alignment.topCenter,  end: Alignment.bottomCenter,  colors: [  const Color.fromARGB(255, 160, 28, 28)!,  Colors.red[900]!,  ],  ))),  const FooterNavigation(),  ]),  ),  drawer: const MakeMenuItems(),  );  }  Widget \_buildNewsSection() {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  \_buildNewsItem(  'https://via.placeholder.com/150', // Placeholder image URL  'THURSDAY, FEBRUARY 25, 2021',  'John Hennessy honored for inventing the chip architecture behind computing',  'Stanford’s President Emeritus and collaborator David Patterson share the BBVA Foundation Frontiers of Knowledge Award for this feat, and for co-authoring a textbook to train chip engineers.',  ),  const SizedBox(height: 25),  \_buildNewsItem(  'https://via.placeholder.com/150', // Placeholder image URL  'TUESDAY, FEBRUARY 9, 2021',  'Once incarcerated, a transfer student forges a new path at Stanford',  'Jason Spyres, who began his university studies as a transfer student in 2018, set his sights on the Farm after hearing an inspiring talk by a Stanford admission officer.',  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(padding: EdgeInsets.zero),  child: const Text(  "More News Stories »",  style: TextStyle(  fontStyle: FontStyle.italic, color: Color(0xE23E3E3E)),  ),  ),  ],  );  }  Widget \_buildNewsItem(  String imageUrl, String date, String title, String description) {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Row(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Container(  color: const Color.fromARGB(255, 176, 176, 176),  padding: const EdgeInsets.all(4),  child: Image.network(  imageUrl, // Placeholder image URL  width: 100,  height: 100,  fit: BoxFit.cover,  ),  ),  const SizedBox(width: 16),  Expanded(  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text(  date,  style: const TextStyle(  fontWeight: FontWeight.bold,  fontSize: 14,  ),  ),  const SizedBox(height: 8),  Text(  title,  style: const TextStyle(  fontWeight: FontWeight.bold,  fontSize: 16,  color: Color.fromARGB(255, 142, 0, 0),  ),  ),  const SizedBox(height: 8),  ],  ),  ),  ],  ),  const SizedBox(height: 8),  Text(  description,  style: const TextStyle(  fontSize: 14, color: Color.fromARGB(255, 37, 37, 37)),  ),  ],  );  }  Widget \_buildEventsSection() {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const SizedBox(height: 8),  \_buildEventItem('SUNDAY, JANUARY 2', 'Stanford Winter Closure Ends',  '12:00 am to 11:45 pm'),  \_buildEventItem('MONDAY, JANUARY 3', 'First Day of Winter Quarter',  '12:00 am to 11:45 pm'),  \_buildEventItem(  'WEDNESDAY, JANUARY 12',  'ETL: Sara Menker, Founder and CEO, Gro Intelligence, Hans Tung, Managing Partner, GGV Capital',  '4:00 pm to 5:00 pm'),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(padding: EdgeInsets.zero),  child: const Text(  "More Events »",  style: TextStyle(  fontStyle: FontStyle.italic, color: Color(0xE23E3E3E)),  ),  ),  ],  );  }  Widget \_buildEventItem(String date, String title, String time) {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text(date,  style: const TextStyle(  fontWeight: FontWeight.bold,  color: Color.fromARGB(255, 65, 65, 65))),  Text(title,  style: const TextStyle(  fontSize: 16, color: Color.fromARGB(255, 110, 7, 0))),  Text(time),  const SizedBox(height: 16),  ],  );  }  Widget \_buildContactInfoSection() {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  const Text(  'Gates Computer Science Building\n353 Jane Stanford Way\nStanford, CA 94305',  style: TextStyle(fontSize: 16),  ),  const SizedBox(height: 8),  const Text('Phone: (650) 723-2300', style: TextStyle(fontSize: 16)),  const SizedBox(height: 8),  const Text('Admissions :', style: TextStyle(fontSize: 16)),  const Text('admissions@cs.stanford.edu',  style:  TextStyle(fontSize: 16, color: Color.fromARGB(255, 110, 7, 0))),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(padding: EdgeInsets.zero),  child: const Text('Campus Map »'),  ),  ],  );  }  }  class SliderScreen extends StatefulWidget {  const SliderScreen({super.key});  @override  State<SliderScreen> createState() => \_SliderScreenState();  }  class \_SliderScreenState extends State<SliderScreen> {  int \_currentIndex = 0;  final List<String> \_images = [  'https://via.placeholder.com/150/FFFF00/000000',  'https://via.placeholder.com/150/0000FF/808080',  ];  void \_previousImage() {  setState(() {  \_currentIndex = (\_currentIndex - 1) % \_images.length;  });  }  void \_nextImage() {  setState(() {  \_currentIndex = (\_currentIndex + 1) % \_images.length;  });  }  @override  Widget build(BuildContext context) {  return Padding(  padding: const EdgeInsets.symmetric(horizontal: 10),  child: Column(  mainAxisSize: MainAxisSize.min,  children: [  Container(  color: const Color.fromARGB(255, 225, 225, 225),  padding: const EdgeInsets.all(8),  child: Stack(  children: [  Image.network(  \_images[\_currentIndex],  width: 400,  height: 200,  fit: BoxFit.cover,  ),  Positioned(  left: 10,  top: 75,  child: CircleAvatar(  backgroundColor: Colors.black54,  child: IconButton(  icon: const Icon(Icons.arrow\_back, color: Colors.white),  onPressed: \_previousImage,  ),  ),  ),  Positioned(  right: 10,  top: 75,  child: CircleAvatar(  backgroundColor: Colors.black54,  child: IconButton(  icon:  const Icon(Icons.arrow\_forward, color: Colors.white),  onPressed: \_nextImage,  ),  ),  ),  ],  ),  ),  Container(  color: const Color.fromARGB(255, 225, 225, 225),  padding: const EdgeInsets.all(8),  child: const Padding(  padding: EdgeInsets.symmetric(horizontal: 8),  child: Text(  'Pat Hanrahan: “Curiosity and passion determine success”',  style: TextStyle(  color: Color.fromARGB(255, 107, 0, 0),  fontSize: 18,  ),  textAlign: TextAlign.left,  ),  )),  ],  ),  );  }  }  class SectionHeading extends StatelessWidget {  final String heading;  const SectionHeading({super.key, required this.heading});  @override  Widget build(BuildContext context) {  return Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  Text(  heading,  style: const TextStyle(  fontSize: 40,  color: Colors.black,  ),  ),  Container(  height: 1,  color: Colors.grey[400],  ),  ],  );  }  }  class FooterNavigation extends StatelessWidget {  const FooterNavigation({super.key});  @override  Widget build(BuildContext context) {  return Container(  color: Colors.red[800],  child: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  mainAxisAlignment: MainAxisAlignment.start,  crossAxisAlignment: CrossAxisAlignment.center,  children: <Widget>[  const Text(  'Stanford',  style: TextStyle(  fontSize: 32,  color: Colors.white,  fontWeight: FontWeight.bold,  fontFamily: "Times"),  textAlign: TextAlign.center,  ),  const Text(  'University',  style: TextStyle(  fontSize: 24,  color: Colors.white,  fontWeight: FontWeight.bold,  fontFamily: "Times"),  textAlign: TextAlign.center,  ),  const SizedBox(height: 16),  Row(  mainAxisAlignment: MainAxisAlignment.center,  crossAxisAlignment: CrossAxisAlignment.start,  children: <Widget>[  Expanded(  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Stanford Home',  style: TextStyle(  color: Colors.white,  fontSize: 20,  ),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Maps & Directions',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Search',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Stanford',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Emergency Info',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  ],  )),  const SizedBox(width: 8),  Expanded(  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Terms of Use',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Copyright',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Trademarks',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Non-Discrimination',  style: TextStyle(color: Colors.white, fontSize: 20),  )),  TextButton(  onPressed: () {},  style: TextButton.styleFrom(),  child: const Text(  'Accessibility',  style: TextStyle(color: Colors.white, fontSize: 20),  ),  ),  ],  ),  ),  ],  ),  const SizedBox(height: 16),  const Text(  '© Stanford University, Stanford, California 94305.',  style: TextStyle(  color: Colors.white,  fontSize: 16,  ),  textAlign: TextAlign.center,  ),  ],  ),  ),  );  }  }  class BannerText extends StatelessWidget {  const BannerText({super.key});  @override  Widget build(BuildContext context) {  return const Padding(  padding: EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text(  'Stanford',  style: TextStyle(  color: Colors.red,  fontSize: 32,  fontWeight: FontWeight.bold,  ),  ),  Text(  'ENGINEERING',  style: TextStyle(  color: Colors.black,  fontSize: 28,  fontWeight: FontWeight.bold,  ),  ),  Text(  'Computer Science',  style: TextStyle(  color: Colors.black54,  fontSize: 24,  ),  ),  ],  ),  );  }  }  class MenuAndSearchBar extends StatelessWidget {  const MenuAndSearchBar({super.key});  @override  Widget build(BuildContext context) {  return Container(  color: const Color.fromARGB(255, 24, 24, 24),  child: Padding(  padding: const EdgeInsets.all(8.0),  child: Row(  children: [  Builder(builder: (context) {  return Padding(  padding: const EdgeInsets.all(4.0),  child: Container(  color: Colors  .red[900], // Maroon background color for drawer icon  child: IconButton(  icon: const Icon(Icons.menu, color: Colors.white),  onPressed: () {  Scaffold.of(context).openDrawer();  },  ),  ),  );  }),  const Spacer(),  Expanded(  flex: 4,  child: Container(  decoration: BoxDecoration(  color: Colors.grey[300], // Cream color background  borderRadius: const BorderRadius.only(  topLeft: Radius.circular(5),  bottomLeft: Radius.circular(5),  ),  ),  child: const TextField(  decoration: InputDecoration(  hintText: 'Search this site...',  hintStyle: TextStyle(color: Colors.black54),  border: InputBorder.none,  contentPadding: EdgeInsets.symmetric(horizontal: 10),  ),  style: TextStyle(color: Colors.black),  ),  ),  ),  Expanded(  flex: 1,  child: Container(  decoration: BoxDecoration(  color: Colors.grey[300], // Cream color background  borderRadius: const BorderRadius.only(  topRight: Radius.circular(5),  bottomRight: Radius.circular(5),  ),  ),  child: IconButton(  icon: const Icon(Icons.search, color: Colors.black),  onPressed: () {  // Perform search action  },  ),  ),  ),  ],  ),  ),  );  }  }  class MakeMenuItems extends StatelessWidget {  const MakeMenuItems({super.key});  @override  Widget build(BuildContext context) {  return Drawer(  child: ListView(  padding: EdgeInsets.zero,  children: <Widget>[  const DrawerHeader(  decoration: BoxDecoration(  color: Colors.red,  ),  child: Text(  'Drawer Header',  style: TextStyle(  color: Colors.white,  fontSize: 24,  ),  ),  ),  ListTile(  leading: const Icon(Icons.home),  title: const Text('Home'),  onTap: () {  // Navigate to home  Navigator.pop(context);  },  ),  ListTile(  leading: const Icon(Icons.settings),  title: const Text('Settings'),  onTap: () {  // Navigate to settings  Navigator.pop(context);  },  ),  ],  ),  );  }  } |

# NAVIGATION IN FLUTTER (CLO-2)

**Question A7:**

Write code that launches a screen Display from Home screen. Send two numbers from Home to Display, where they should be shown separately in TextField widgets.

|  |
| --- |
| /\*  Question A7:  Write code that launches a screen Display from Home screen. Send two numbers from Home to Display, where they should be shown separately in TextField widgets.  \*/  import 'package:flutter/material.dart';  void main() {  runApp(  const MaterialApp(title: 'Passing Data', home: HomeScreen()),  );  }  class HomeScreen extends StatelessWidget {  const HomeScreen({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Send Data'),  ),  body: Center(  child: Column(  mainAxisAlignment: MainAxisAlignment.center,  children: [  ElevatedButton(  onPressed: () {  Map<String, dynamic> data = {"num1": 10, "num2": 20};  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const DataReceiver(),  settings: RouteSettings(arguments: data),  ),  );  },  child: const Text('Send'),  ),  ],  ),  ),  );  }  }  class DataReceiver extends StatelessWidget {  const DataReceiver({super.key});  @override  Widget build(BuildContext context) {  final args = ModalRoute.of(context)!.settings.arguments as Map;  return Scaffold(  appBar: AppBar(  title: const Text('Received Data'),  ),  body: Center(  child: Text(  "Number 1: ${args['num1']}\nNumber 2: ${args['num2']}",  style: const TextStyle(fontSize: 20.0),  ),  ),  );  }  } |

**Question A11:**

We have two navigation screens as shown below. Name and email are input in the home screen and when the submit button is clicked, the values are passed to profile screen (using RouteSettings) where they are simply displayed. You need to write the widgets of homescreen and profilescreen.

Email

Name

Submit

Email:

Name

Back

*Ali Shah*

*alishah@mail.com*

HOME SCREEN

PROFILE SCREEN

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(home: HomeScreen()));  }  class HomeScreen extends StatefulWidget {  const HomeScreen({super.key});  @override  State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  final TextEditingController \_email = TextEditingController();  final TextEditingController \_name = TextEditingController();  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('HOME SCREEN')),  ),  body: Column(  children: [  Row(  children: [  const Expanded(  child: Padding(  padding: EdgeInsets.only(left: 50),  child: Text('E-mail'),  ),  ),  SizedBox(  width: 240,  child: Padding(  padding: const EdgeInsets.only(right: 50),  child: SizedBox(  child: TextField(  controller: \_email,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding: EdgeInsets.symmetric(  vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  )  ],  ),  const SizedBox(height: 20),  Row(  children: [  const Expanded(  child: Padding(  padding: EdgeInsets.only(left: 50),  child: Text('Person Name'),  ),  ),  SizedBox(  width: 240,  child: Padding(  padding: const EdgeInsets.only(right: 50),  child: SizedBox(  child: TextField(  controller: \_name,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding: EdgeInsets.symmetric(  vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  )  ],  ),  Row(  children: [  Expanded(  child: Padding(  padding: const EdgeInsets.only(top: 10, left: 80, right: 80),  child: ElevatedButton(  onPressed: () {  Map<String, dynamic> data = {  "email": \_email.text,  "name": \_name.text  };  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const ProfileScreen(),  settings: RouteSettings(arguments: data),  ),  );  },  child: const Text('SUBMIT'),  ),  ),  ),  ],  ),  ],  ),  );  }  }  class ProfileScreen extends StatelessWidget {  const ProfileScreen({super.key});  @override  Widget build(BuildContext context) {  final args = ModalRoute.of(context)!.settings.arguments as Map;  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('PROFILE SCREEN')),  ),  body: Column(  children: [  Row(  children: [  const Expanded(  child: Padding(  padding: EdgeInsets.only(left: 50),  child: Text('E-mail'),  ),  ),  SizedBox(  width: 240,  child: Padding(  padding: const EdgeInsets.only(right: 50),  child: SizedBox(  child: Text(args["email"],  style: const TextStyle(fontStyle: FontStyle.italic)),  ),  ),  )  ],  ),  const SizedBox(height: 20),  Row(  children: [  const Expanded(  child: Padding(  padding: EdgeInsets.only(left: 50),  child: Text('Person Name'),  ),  ),  SizedBox(  width: 240,  child: Padding(  padding: const EdgeInsets.only(right: 50),  child: SizedBox(  child: Text(args["name"],  style: const TextStyle(fontStyle: FontStyle.italic)),  ),  ),  )  ],  ),  ],  ),  );  }  } |

Question 2025\_1

Design a flutter app with the following details.

|  |  |
| --- | --- |
| **InputScreen**:  The inputscreen takes the values of name and age in TextFields and send to the DisplayScreen widget  Name  \_\_\_\_\_\_\_\_\_\_\_\_\_ .  Age  \_\_\_\_\_\_\_\_\_\_\_\_\_ .  Search | **DisplayScreen**  The display screen contains a list of map objects, e.g.. [{“id”:10, “name”:”Ali”, “age”:30},{“id”:20, “name”:”Javed”, “age”:31}],  You need to search the name and age together in the list. If the record is found, it should be displayed in the DisplayScreen using Text widgets as below:    20  Ali  30 |

Solution

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {    runApp(MaterialApp(home: InputScreen()));  }  class InputScreen extends StatefulWidget {    InputScreen({super.key});    @override    State<InputScreen> createState() => \_InputScreenState();  }  class \_InputScreenState extends State<InputScreen> {    TextEditingController namecontroller = TextEditingController();    TextEditingController agecontroller = TextEditingController();    @override    Widget build(BuildContext context) {      return Scaffold(        body: SafeArea(            child: Padding(          padding: EdgeInsets.all(20),          child: Column(            crossAxisAlignment: CrossAxisAlignment.start,            children: [              Text("Name"),              TextField(controller: namecontroller),              SizedBox(                height: 20,              ),              Text("Age"),              TextField(controller: agecontroller),              ElevatedButton(                  onPressed: () {                    Navigator.push(                        context,                        MaterialPageRoute(                            builder: (context) => DisplayScreen(                                name: namecontroller.text,                                age: int.tryParse(agecontroller.text))));                  },                  child: Text("Search"))            ],          ),        )),      );    }  }  class DisplayScreen extends StatefulWidget {    String name;    int? age;    DisplayScreen({super.key, required this.name, required this.age});    @override    State<DisplayScreen> createState() => \_DisplayScreenState();  }  class \_DisplayScreenState extends State<DisplayScreen> {    String? txtname;    int? txtid;    int? txtage;    bool found = false;    int index = 0;    List<Map<dynamic, dynamic>> names = [      {"id": 10, "name": "Ali", "age": 40},      {"id": 20, "name": "Noman", "age": 50},      {"id": 30, "name": "Faisal", "age": 50},    ];    @override    Widget build(BuildContext context) {      names          .where(              (item) => item["name"] == widget.name && item["age"] == widget.age)          .forEach((item) {        txtid = item["id"];        txtname = item["name"];        txtage = item["age"];        found = true;      });      return Scaffold(          appBar: AppBar(            title: Text("Display Screen"),          ),          body: SafeArea(              child: Padding(                  padding: EdgeInsets.all(20),                  child: found                      ? Column(                          crossAxisAlignment: CrossAxisAlignment.start,                          children: [                              Text(txtid.toString()),                              Text(txtname.toString()),                              Text(txtage.toString()),                              ])                      : Text("Record not found"))));    }  } |

aaa

**Question A12:**

Suppose we have a layout like this

0

COUNTER

SEND

RANDOM

The value is 33

Write code for button RANDOM such that when user click button, a random number from one to hundred is shown in text box. Write code for button COUNTER such that when the user click the button, the value in the TextField start incrementing. Write code for button SEND such that when user click on button, the value in TextField is passed to a new screen (Display) and shown as indicated in the figure.

|  |
| --- |
| import 'dart:math';  import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(home: HomeScreen()));  }  class HomeScreen extends StatefulWidget {  const HomeScreen({super.key});  @override  State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  final TextEditingController \_controller = TextEditingController();  int random = 0;  void updateTextBox(String action) {  setState(() {  if (action.compareTo("RANDOM") == 0) {  random = Random().nextInt(100) + 1;  \_controller.text = random.toString();  } else {  if (int.tryParse(\_controller.text) != null) {  int temp = int.parse(\_controller.text);  temp = temp + 1;  \_controller.text = temp.toString();  }  }  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('HOME SCREEN')),  ),  body: Column(  children: [  SizedBox(  width: 240,  child: Padding(  padding: const EdgeInsets.only(right: 0, left: 0),  child: SizedBox(  child: TextField(  controller: \_controller,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding:  EdgeInsets.symmetric(vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  ),  const SizedBox(height: 20),  Row(  children: [  Expanded(  child: Padding(  padding: const EdgeInsets.only(top: 0, left: 10, right: 0),  child: ElevatedButton(  onPressed: () => updateTextBox("RANDOM"),  child: const Text('RANDOM'),  ),  ),  ),  Expanded(  child: Padding(  padding: const EdgeInsets.only(top: 0, left: 10, right: 0),  child: ElevatedButton(  onPressed: () => updateTextBox("COUNTER"),  child: const Text('COUNTER'),  ),  ),  ),  Expanded(  child: Padding(  padding: const EdgeInsets.only(top: 0, left: 10, right: 10),  child: ElevatedButton(  onPressed: () {  Map<String, dynamic> data = {"value": \_controller.text};  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const Display(),  settings: RouteSettings(arguments: data),  ),  );  },  child: const Text('SEND'),  ),  ),  ),  ],  ),  ],  ),  );  }  }  class Display extends StatelessWidget {  const Display({super.key});  @override  Widget build(BuildContext context) {  final args = ModalRoute.of(context)!.settings.arguments as Map;  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('DISPLAY SCREEN')),  ),  body: Column(  children: [  SizedBox(  width: 300,  child: Padding(  padding: const EdgeInsets.only(right: 50, left: 50),  child: SizedBox(  child: Text("The value is: ${args["value"]}"),  ),  ),  ),  ],  ),  );  }  } |

**Question A21:**

When the user click on button in Home screen, the both strings in TextFields should be passed separately to the Display screen where they are shown as concatenated string in display function of Display screen.

Abbottabad

CLICK

Pakistan

Abbottabad, Pakistan

Home

Display

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(home: HomeScreen()));  }  class HomeScreen extends StatelessWidget {  final TextEditingController \_city = TextEditingController(text: "Abbottabad");  final TextEditingController \_country =  TextEditingController(text: "Pakistan");  HomeScreen({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('HOME')),  ),  body: Column(  children: [  SizedBox(  width: 400,  child: Padding(  padding: const EdgeInsets.only(left: 50, right: 50),  child: SizedBox(  child: TextField(  controller: \_city,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding:  EdgeInsets.symmetric(vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  ),  const SizedBox(height: 20),  SizedBox(  width: 400,  child: Padding(  padding: const EdgeInsets.only(left: 50, right: 50),  child: SizedBox(  child: TextField(  controller: \_country,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding:  EdgeInsets.symmetric(vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  ),  Padding(  padding: const EdgeInsets.only(top: 10, left: 80, right: 80),  child: ElevatedButton(  onPressed: () {  Map<String, dynamic> data = {  "city": \_city.text,  "country": \_country.text  };  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const DisplayScreen(),  settings: RouteSettings(arguments: data),  ),  );  },  child: const Text('SUBMIT'),  ),  ),  ],  ),  );  }  }  class DisplayScreen extends StatelessWidget {  const DisplayScreen({super.key});  @override  Widget build(BuildContext context) {  final args = ModalRoute.of(context)!.settings.arguments as Map;  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('DISPLAY')),  ),  body: Column(  children: [  Row(  children: [  Expanded(  child: Padding(  padding: const EdgeInsets.only(left: 50, right: 50),  child: Container(  decoration: BoxDecoration(border: Border.all(width: 1)),  child: Padding(  padding: const EdgeInsets.all(  10), // Adjust the padding value as needed  child: Text("${args["city"]}, ${args["country"]}"),  ),  ),  ),  ),  ],  ),  ],  ),  );  }  } |

**Question A22:**

Pass a number from 1 to 3 from the Home screen to the Display screen. In the Display screen, check which of the number is received, and then write the number in words.   
For example, you passed 3 from Home screen, and in Display screen, you will print “three” as shown below.

3

CLICK

The received number is three

Home

Display

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(home: HomeScreen()));  }  class HomeScreen extends StatelessWidget {  final TextEditingController \_num = TextEditingController();  HomeScreen({super.key});  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('HOME')),  ),  body: Column(  children: [  SizedBox(  width: 400,  child: Padding(  padding: const EdgeInsets.only(left: 50, right: 50),  child: SizedBox(  child: TextField(  controller: \_num,  decoration: const InputDecoration(  border: OutlineInputBorder(),  contentPadding:  EdgeInsets.symmetric(vertical: 0.0, horizontal: 10.0),  ),  ),  ),  ),  ),  Padding(  padding: const EdgeInsets.only(top: 10, left: 80, right: 80),  child: ElevatedButton(  onPressed: () {  Map<String, dynamic> data = {  "num": \_num.text,  };  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const DisplayScreen(),  settings: RouteSettings(arguments: data),  ),  );  },  child: const Text('SUBMIT'),  ),  ),  ],  ),  );  }  }  class DisplayScreen extends StatelessWidget {  const DisplayScreen({super.key});  @override  Widget build(BuildContext context) {  final args = ModalRoute.of(context)!.settings.arguments as Map;  int num = 0;  String result = "";  if (int.tryParse(args["num"]) != null) {  num = int.parse(args["num"]);  if (num == 1) {  result = "Received number is one";  } else if (num == 2) {  result = "Received number is two";  } else if (num == 3) {  result = "Received number is three";  } else {  result = "Invalid number";  }  } else {  result = "Invalid number";  }  return Scaffold(  appBar: AppBar(  title: const Center(child: Text('DISPLAY')),  ),  body: Column(  children: [  Row(  children: [  Expanded(  child: Padding(  padding: const EdgeInsets.only(left: 50, right: 50),  child: Container(  decoration: BoxDecoration(border: Border.all(width: 1)),  child: Padding(  padding: const EdgeInsets.all(  10), // Adjust the padding value as needed  child: Text(result),  ),  ),  ),  ),  ],  ),  ],  ),  );  }  } |

**Question A23:**

Use Flutter Widgets to design following screens:

Home screen

It will show some main categories like:  
  
COMPUTERS, LAPTOPS, HARD DRIVES, FLASH MEMORIES, ETC.

You can use lists to show above categories. Apply the proper theme and styling.

When any of the category name is clicked a new screen should open showing sub-categories of that parent category, and the title of the page should change to the parent category.

For example, when LAPTOPS is clicked the new screen can show models of different laptops along with their pics and prices:

The laptops should be shown as a grid.

HP Pavilion 15

Dell Inspiron

Sony Viao

And so on

When a laptop model is clicked, its individual detail should be shown on separate screen.

You can use different .dart files for different screens and then use import to call in main.dart.  
  
Each subcategory page should have back navigation button and a button to navigate directly to the home screen.

Please make beautiful interfaces, some samples are attached, but you can find more on flutter.dev.





|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(MaterialApp(  debugShowCheckedModeBanner: false,  theme: ThemeData(  colorScheme: ColorScheme.fromSeed(  seedColor: const Color.fromARGB(255, 178, 163, 203)),  cardTheme: const CardTheme(color: Color.fromARGB(255, 255, 254, 250)),  useMaterial3: true,  ),  home: HomePage(),  ));  }  class HomePage extends StatelessWidget {  HomePage({super.key});  final List<Category> cats = [  Category('Laptops', 'https://via.placeholder.com/50'),  Category('Work Stations', 'https://via.placeholder.com/50'),  Category('Hard Drives', 'https://via.placeholder.com/50'),  Category('Flash Drives', 'https://via.placeholder.com/50'),  Category('Solid State Drives', 'https://via.placeholder.com/50'),  Category('Keyboards', 'https://via.placeholder.com/50'),  ];  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Products'),  ),  body: ListView.builder(  itemCount: cats.length,  itemBuilder: (context, index) {  final cat = cats[index];  return Padding(  padding: const EdgeInsets.all(8.0),  child: Card(  elevation: 3, // Adjust the elevation value as needed  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(10),  ),  child: Container(  decoration: BoxDecoration(  color: Colors.blue.shade50,  borderRadius: BorderRadius.circular(10),  ),  child: ListTile(  leading:  Image.network(cat.thumbnailUrl, width: 40, height: 40),  title: Text(cat.name),  onTap: () {  Navigator.push(  context,  MaterialPageRoute(  builder: (context) =>  Products(productCategory: cat.name),  ),  );  },  ),  ),  ),  );  },  ),  );  }  }  class DetailPage extends StatelessWidget {  final Product product;  const DetailPage({super.key, required this.product});  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: Text(product.name),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Card(  elevation: 4,  child: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  mainAxisSize: MainAxisSize.min,  children: [  Image.network(  product.thumbnailUrl,  width: 200,  height: 200,  fit: BoxFit.cover,  ),  const SizedBox(height: 16),  Text(  product.name,  style: const TextStyle(  fontSize: 24, fontWeight: FontWeight.bold),  ),  Text("Price: ${product.price}"),  const SizedBox(height: 12),  Text(product.specs),  ElevatedButton(  onPressed: () {},  style: ElevatedButton.styleFrom(  elevation: 4,  backgroundColor: const Color.fromARGB(  255, 104, 185, 251), // Background color  foregroundColor: Colors.black, // Text color  ),  child: const Text("Add to Cart"),  )  ],  ),  ),  )));  }  }  class Category {  final String name;  final String thumbnailUrl;  Category(this.name, this.thumbnailUrl);  }  class Product {  final String name;  final String thumbnailUrl;  final double price;  final String specs;  Product(  {required this.name,  required this.thumbnailUrl,  required this.price,  required this.specs});  }  class Products extends StatelessWidget {  final String productCategory;  Products({super.key, required this.productCategory});  final List<Product> products = [  Product(  name: 'HP Pavillion',  thumbnailUrl: 'https://via.placeholder.com/50',  price: 339.99,  specs:  "Core i9 Processor, 300 GB SSD, Intel Graphics Card, 15.1 inch screen size",  ),  Product(  name: 'Dell Latitude',  thumbnailUrl: 'https://via.placeholder.com/50',  price: 319.99,  specs:  "Core i7 Processor, 200 GB SSD, Intel Graphics Card, 15.1 inch screen size",  ),  Product(  name: 'Dell Inspiron',  thumbnailUrl: 'https://via.placeholder.com/50',  price: 4419.99,  specs:  "Core i6 Processor, 600 GB SSD, Intel Graphics Card, 15.1 inch screen size",  ),  Product(  name: 'HP Notebook',  thumbnailUrl: 'https://via.placeholder.com/50',  price: 3249.99,  specs:  "Core i9 Processor, 300 GB SSD, Intel Graphics Card, 15.1 inch screen size",  ),  Product(  name: 'Acer Max',  thumbnailUrl: 'https://via.placeholder.com/50',  price: 249.99,  specs:  "Core i9 Processor, 300 GB SSD, Intel Graphics Card, 15.1 inch screen size",  ),  ];  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: Text(productCategory),  ),  body: Padding(  padding: const EdgeInsets.all(20.0), // Padding around the grid  child: GridView.builder(  gridDelegate: const SliverGridDelegateWithFixedCrossAxisCount(  crossAxisCount: 3, // Number of columns  childAspectRatio: 0.6, // Aspect ratio of the grid items  mainAxisSpacing: 10.0,  crossAxisSpacing: 10.0,  ),  itemCount: products.length,  itemBuilder: (BuildContext context, int index) {  final product = products[index];  return GestureDetector(  onTap: () {  // Navigate to product detail page  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => DetailPage(product: product),  ),  );  },  child: Container(  padding: const EdgeInsets.all(8),  decoration: BoxDecoration(  color: const Color.fromARGB(255, 254, 255, 251),  borderRadius: BorderRadius.circular(10),  boxShadow: const [  BoxShadow(  blurStyle: BlurStyle.normal,  blurRadius: 3,  color: Colors.black)  ]),  // Adjust width as needed  child: Column(  crossAxisAlignment: CrossAxisAlignment.center,  children: [  Image.network(  product.thumbnailUrl,  height: 90, // Adjust height as needed  // Make image fit the width of the card  //width: double.infinity,  fit: BoxFit.cover,  ),  const SizedBox(height: 8), // Space between image and text  Text(  product.name,  style: const TextStyle(  fontSize: 16, fontWeight: FontWeight.bold),  ),  const SizedBox(height: 4),  Text('\$${product.price.toStringAsFixed(2)}'),  ],  ),  ),  );  },  ),  ),  );  }  } |

**Question A24:**

Create an app with 4 screens, and use Tab navigation to navigate between screens

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MaterialApp(home: TabBarDemo()));  }  class TabBarDemo extends StatelessWidget {  const TabBarDemo({super.key});  @override  Widget build(BuildContext context) {  return DefaultTabController(  length: 4,  child: Scaffold(  appBar: AppBar(  bottom: const TabBar(  tabs: [  Tab(icon: Icon(Icons.directions\_car)),  Tab(icon: Icon(Icons.directions\_transit)),  Tab(icon: Icon(Icons.directions\_bike)),  Tab(icon: Icon(Icons.directions\_boat)),  ],  ),  title: const Text('Tabs Demo'),  ),  body: const TabBarView(  children: [  MyScreen(item: "Car"),  MyScreen(item: "Bus"),  MyScreen(item: "Cycle"),  MyScreen(item: "Boat")  ],  ),  ),  );  }  }  class MyScreen extends StatelessWidget {  final String item;  const MyScreen({super.key, required this.item});  @override  Widget build(BuildContext context) {  return Text(item);  }  } |

**Question A26:**

Create a screen with a banner having a search field with button and a single button for sign in / sign out. When the user clicks on sign in button, the text of button changes to sign out and vice versa.

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: BannerScreen(),  debugShowCheckedModeBanner: false,  );  }  }  class BannerScreen extends StatefulWidget {  const BannerScreen({super.key});  @override  \_BannerScreenState createState() => \_BannerScreenState();  }  class \_BannerScreenState extends State<BannerScreen> {  bool \_isSignedIn = false;  void \_toggleSignIn() {  setState(() {  \_isSignedIn = !\_isSignedIn;  });  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  toolbarHeight: 80,  backgroundColor: Colors.teal,  title: Row(  children: [  Expanded(  child: TextField(  decoration: InputDecoration(  hintText: 'Search...',  fillColor: Colors.white,  filled: true,  contentPadding: const EdgeInsets.symmetric(  vertical: 10.0, horizontal: 15.0),  border: OutlineInputBorder(  borderRadius: BorderRadius.circular(25.0),  borderSide: BorderSide.none,  ),  ),  ),  ),  const SizedBox(width: 10.0),  ElevatedButton(  onPressed: () {  // Perform search action  },  style: ElevatedButton.styleFrom(  backgroundColor: Colors.orange,  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(20.0),  ),  padding: const EdgeInsets.symmetric(horizontal: 20.0),  ),  child: const Text('Search'),  ),  const SizedBox(width: 10.0),  ElevatedButton(  onPressed: \_toggleSignIn,  style: ElevatedButton.styleFrom(  backgroundColor: \_isSignedIn ? Colors.red : Colors.green,  shape: RoundedRectangleBorder(  borderRadius: BorderRadius.circular(20.0),  ),  padding: const EdgeInsets.symmetric(horizontal: 20.0),  ),  child: Text(\_isSignedIn ? 'Sign Out' : 'Sign In'),  ),  ],  ),  ),  body: const Center(  child: Text('Content goes here'),  ),  );  }  } |

**Question A27:**

Create an app with 4 screens, screen1, screen2, screen3, and screen4. Use grouping with Tab navigation to place screen1 and screen2 in tab 1 and screen 3 and screen 4 in tab 2

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: MainTabs(),  );  }  }  class MainTabs extends StatefulWidget {  const MainTabs({super.key});  @override  State<MainTabs> createState() => \_MainTabsState();  }  class \_MainTabsState extends State<MainTabs>  with SingleTickerProviderStateMixin {  late TabController \_mainTabController;  @override  void initState() {  super.initState();  \_mainTabController = TabController(length: 2, vsync: this);  }  @override  void dispose() {  \_mainTabController.dispose();  super.dispose();  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Main Tabs'),  bottom: TabBar(  controller: \_mainTabController,  tabs: const [  Tab(text: 'Tab 1'),  Tab(text: 'Tab 2'),  ],  ),  ),  body: TabBarView(  controller: \_mainTabController,  children: const [  Tab1(),  Tab2(),  ],  ),  );  }  }  class Tab1 extends StatelessWidget {  const Tab1({super.key});  @override  Widget build(BuildContext context) {  return DefaultTabController(  length: 2,  child: Scaffold(  appBar: AppBar(  automaticallyImplyLeading: false,  bottom: const TabBar(  tabs: [  Tab(text: 'Screen 1'),  Tab(text: 'Screen 2'),  ],  ),  ),  body: const TabBarView(  children: [  Screen1(),  Screen2(),  ],  ),  ),  );  }  }  class Tab2 extends StatelessWidget {  const Tab2({super.key});  @override  Widget build(BuildContext context) {  return DefaultTabController(  length: 2,  child: Scaffold(  appBar: AppBar(  automaticallyImplyLeading: false,  bottom: const TabBar(  tabs: [  Tab(text: 'Screen 3'),  Tab(text: 'Screen 4'),  ],  ),  ),  body: const TabBarView(  children: [  Screen3(),  Screen4(),  ],  ),  ),  );  }  }  class Screen1 extends StatelessWidget {  const Screen1({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 1'),  );  }  }  class Screen2 extends StatelessWidget {  const Screen2({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 2'),  );  }  }  class Screen3 extends StatelessWidget {  const Screen3({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 3'),  );  }  }  class Screen4 extends StatelessWidget {  const Screen4({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 4'),  );  }  } |

**Question A28:**

Create an app with 4 screens, and use Drawer navigation to navigate between screens

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: HomeScreen(),  );  }  }  class HomeScreen extends StatefulWidget {  const HomeScreen({super.key});  @override  State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  int \_selectedDrawerIndex = 0;  \_getDrawerItemWidget(int pos) {  switch (pos) {  case 0:  return const Screen1();  case 1:  return const Screen2();  case 2:  return const Screen3();  case 3:  return const Screen4();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectItem(int index) {  setState(() {  \_selectedDrawerIndex = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Drawer Navigation'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Navigation Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Screen 1'),  selected: \_selectedDrawerIndex == 0,  onTap: () => \_onSelectItem(0),  ),  ListTile(  title: const Text('Screen 2'),  selected: \_selectedDrawerIndex == 1,  onTap: () => \_onSelectItem(1),  ),  ListTile(  title: const Text('Screen 3'),  selected: \_selectedDrawerIndex == 2,  onTap: () => \_onSelectItem(2),  ),  ListTile(  title: const Text('Screen 4'),  selected: \_selectedDrawerIndex == 3,  onTap: () => \_onSelectItem(3),  ),  ],  ),  ),  body: \_getDrawerItemWidget(\_selectedDrawerIndex),  );  }  }  class Screen1 extends StatelessWidget {  const Screen1({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 1'),  );  }  }  class Screen2 extends StatelessWidget {  const Screen2({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 2'),  );  }  }  class Screen3 extends StatelessWidget {  const Screen3({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 3'),  );  }  }  class Screen4 extends StatelessWidget {  const Screen4({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 4'),  );  }  } |

**Question A29:**

Create an app with 4 screens, screen1, screen2, screen3, and screen4. Use Drawer navigation based grouping to place screen1 and screen2 in drawer 1 and screen 3 and screen 4 in drawer 2

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: MainDrawer(),  );  }  }  class MainDrawer extends StatefulWidget {  const MainDrawer({super.key});  @override  State<MainDrawer> createState() => \_MainDrawerState();  }  class \_MainDrawerState extends State<MainDrawer> {  int \_selectedDrawer = 0;  \_getDrawerContent(int pos) {  switch (pos) {  case 0:  return const Drawer1();  case 1:  return const Drawer2();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectMainDrawer(int index) {  setState(() {  \_selectedDrawer = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Main Drawer Navigation'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Main Navigation Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Drawer 1'),  selected: \_selectedDrawer == 0,  onTap: () => \_onSelectMainDrawer(0),  ),  ListTile(  title: const Text('Drawer 2'),  selected: \_selectedDrawer == 1,  onTap: () => \_onSelectMainDrawer(1),  ),  ],  ),  ),  body: \_getDrawerContent(\_selectedDrawer),  );  }  }  class Drawer1 extends StatefulWidget {  const Drawer1({super.key});  @override  State<Drawer1> createState() => \_Drawer1State();  }  class \_Drawer1State extends State<Drawer1> {  int \_selectedDrawerIndex = 0;  \_getDrawer1ItemWidget(int pos) {  switch (pos) {  case 0:  return const Screen1();  case 1:  return const Screen2();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectDrawer1Item(int index) {  setState(() {  \_selectedDrawerIndex = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Drawer 1'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Drawer 1 Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Screen 1'),  selected: \_selectedDrawerIndex == 0,  onTap: () => \_onSelectDrawer1Item(0),  ),  ListTile(  title: const Text('Screen 2'),  selected: \_selectedDrawerIndex == 1,  onTap: () => \_onSelectDrawer1Item(1),  ),  ],  ),  ),  body: \_getDrawer1ItemWidget(\_selectedDrawerIndex),  );  }  }  class Drawer2 extends StatefulWidget {  const Drawer2({super.key});  @override  State<Drawer2> createState() => \_Drawer2State();  }  class \_Drawer2State extends State<Drawer2> {  int \_selectedDrawerIndex = 0;  \_getDrawer2ItemWidget(int pos) {  switch (pos) {  case 0:  return const Screen3();  case 1:  return const Screen4();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectDrawer2Item(int index) {  setState(() {  \_selectedDrawerIndex = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Drawer 2'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Drawer 2 Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Screen 3'),  selected: \_selectedDrawerIndex == 0,  onTap: () => \_onSelectDrawer2Item(0),  ),  ListTile(  title: const Text('Screen 4'),  selected: \_selectedDrawerIndex == 1,  onTap: () => \_onSelectDrawer2Item(1),  ),  ],  ),  ),  body: \_getDrawer2ItemWidget(\_selectedDrawerIndex),  );  }  }  class Screen1 extends StatelessWidget {  const Screen1({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 1'),  );  }  }  class Screen2 extends StatelessWidget {  const Screen2({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 2'),  );  }  }  class Screen3 extends StatelessWidget {  const Screen3({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 3'),  );  }  }  class Screen4 extends StatelessWidget {  const Screen4({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 4'),  );  }  } |

**Question C13:**

Create the following Form

The data is input on the “input screen”.

When the user clicks the submit button, the data is sent to the “display screen” as shown below.

**INPUT SCREEN**

Submit

Email

Name

Gender

Male

Female

Country

Skills

Designing

Backend

Database

Subjects

Phy

Chem

Bio

Address

**DISPLAY SCREEN**

Save

ali@gmail.com

Email

Ali Khan

Name

Gender

Male

Country

Pakistan

Skills

Designing, Database

Subjects

Phy,

Bio

Address

Lahore, Gulberg

|  |
| --- |
| import 'package:flutter/material.dart';  void main() {  runApp(const MyApp());  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: MainDrawer(),  );  }  }  class MainDrawer extends StatefulWidget {  const MainDrawer({super.key});  @override  State<MainDrawer> createState() => \_MainDrawerState();  }  class \_MainDrawerState extends State<MainDrawer> {  int \_selectedDrawer = 0;  \_getDrawerContent(int pos) {  switch (pos) {  case 0:  return const Drawer1();  case 1:  return const Drawer2();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectMainDrawer(int index) {  setState(() {  \_selectedDrawer = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Main Drawer Navigation'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Main Navigation Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Drawer 1'),  selected: \_selectedDrawer == 0,  onTap: () => \_onSelectMainDrawer(0),  ),  ListTile(  title: const Text('Drawer 2'),  selected: \_selectedDrawer == 1,  onTap: () => \_onSelectMainDrawer(1),  ),  ],  ),  ),  body: \_getDrawerContent(\_selectedDrawer),  );  }  }  class Drawer1 extends StatefulWidget {  const Drawer1({super.key});  @override  State<Drawer1> createState() => \_Drawer1State();  }  class \_Drawer1State extends State<Drawer1> {  int \_selectedDrawerIndex = 0;  \_getDrawer1ItemWidget(int pos) {  switch (pos) {  case 0:  return const Screen1();  case 1:  return const Screen2();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectDrawer1Item(int index) {  setState(() {  \_selectedDrawerIndex = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Drawer 1'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Drawer 1 Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Screen 1'),  selected: \_selectedDrawerIndex == 0,  onTap: () => \_onSelectDrawer1Item(0),  ),  ListTile(  title: const Text('Screen 2'),  selected: \_selectedDrawerIndex == 1,  onTap: () => \_onSelectDrawer1Item(1),  ),  ],  ),  ),  body: \_getDrawer1ItemWidget(\_selectedDrawerIndex),  );  }  }  class Drawer2 extends StatefulWidget {  const Drawer2({super.key});  @override  State<Drawer2> createState() => \_Drawer2State();  }  class \_Drawer2State extends State<Drawer2> {  int \_selectedDrawerIndex = 0;  \_getDrawer2ItemWidget(int pos) {  switch (pos) {  case 0:  return const Screen3();  case 1:  return const Screen4();  default:  return const Center(child: Text("Error"));  }  }  \_onSelectDrawer2Item(int index) {  setState(() {  \_selectedDrawerIndex = index;  });  Navigator.of(context).pop(); // close the drawer  }  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Drawer 2'),  ),  drawer: Drawer(  child: ListView(  children: <Widget>[  const DrawerHeader(  child: Text(  'Drawer 2 Menu',  style: TextStyle(color: Colors.white, fontSize: 25),  ),  decoration: BoxDecoration(  color: Colors.blue,  ),  ),  ListTile(  title: const Text('Screen 3'),  selected: \_selectedDrawerIndex == 0,  onTap: () => \_onSelectDrawer2Item(0),  ),  ListTile(  title: const Text('Screen 4'),  selected: \_selectedDrawerIndex == 1,  onTap: () => \_onSelectDrawer2Item(1),  ),  ],  ),  ),  body: \_getDrawer2ItemWidget(\_selectedDrawerIndex),  );  }  }  class Screen1 extends StatelessWidget {  const Screen1({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 1'),  );  }  }  class Screen2 extends StatelessWidget {  const Screen2({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 2'),  );  }  }  class Screen3 extends StatelessWidget {  const Screen3({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 3'),  );  }  }  class Screen4 extends StatelessWidget {  const Screen4({super.key});  @override  Widget build(BuildContext context) {  return const Center(  child: Text('Screen 4'),  );  }  } |

**Question C19:**

Create two screens screen1 and screen2. Share global data between the screens containing fields: name and age. Also change the values of the name and age in screen2.

|  |
| --- |
| import 'package:flutter/foundation.dart';  import 'package:flutter/material.dart';  import 'package:provider/provider.dart';  void main() {  runApp(  ChangeNotifierProvider(  create: (context) => AppState(),  child: const MyApp(),  ),  );  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: HomeScreen(),  );  }  }  class AppState extends ChangeNotifier {  String \_email = '';  String \_name = '';  String get email => \_email;  String get name => \_name;  void setEmail(String email) {  \_email = email;  notifyListeners();  }  void setName(String name) {  \_name = name;  notifyListeners();  }  }  class HomeScreen extends StatefulWidget {  const HomeScreen({super.key});  @override  State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  final TextEditingController \_emailController = TextEditingController();  final TextEditingController \_nameController = TextEditingController();  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Home Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  children: [  TextField(  controller: \_emailController,  decoration: const InputDecoration(labelText: 'Email'),  ),  TextField(  controller: \_nameController,  decoration: const InputDecoration(labelText: 'Name'),  ),  const SizedBox(height: 20),  ElevatedButton(  onPressed: () {  Provider.of<AppState>(context, listen: false)  .setEmail(\_emailController.text);  Provider.of<AppState>(context, listen: false)  .setName(\_nameController.text);  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const ProfileScreen()),  );  },  child: const Text('Submit'),  ),  ],  ),  ),  );  }  }  class ProfileScreen extends StatelessWidget {  const ProfileScreen({super.key});  @override  Widget build(BuildContext context) {  final appState = Provider.of<AppState>(context);  return Scaffold(  appBar: AppBar(  title: const Text('Profile Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text('Email: ${appState.email}',  style: const TextStyle(fontSize: 20)),  Text('Name: ${appState.name}',  style: const TextStyle(fontSize: 20)),  ElevatedButton(  onPressed: () {  Provider.of<AppState>(context, listen: false)  .setName("THIS IS UPDATED NAME");  },  child: const Text("Update Name")),  ElevatedButton(  onPressed: () {  Navigator.push(  context,  MaterialPageRoute(builder: (context) => const ResultScreen()),  );  },  child: const Text('Submit'),  ),  ],  ),  ),  );  }  }  class ResultScreen extends StatelessWidget {  const ResultScreen({super.key});  @override  Widget build(BuildContext context) {  final appState = Provider.of<AppState>(context);  return Scaffold(  appBar: AppBar(  title: const Text('Result Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text('Email: ${appState.email}',  style: const TextStyle(fontSize: 20)),  Text('Name: ${appState.name}',  style: const TextStyle(fontSize: 20)),  ],  ),  ),  );  }  } |

# DATABASE CONNECTIVITY IN FLUTTER (CLO-3)

**Question C20:**

Write an application that asks for an ID from user in RecordSearch screen. When the user enters the ID, the record is shown against the ID in the RecordView screen. The record is fetched from SQLite Database.

Enter ID

**SEARCH**

25

ID

Osman

NAME

45

AGE

Abbottabad, Pakistan

ADDRESS

Record Search Activity

View Record Activity

|  |
| --- |
|  |

**Question C21:**

Write a program to store the following key value pairs using Shared preferences in Flutter. Create a function to save the information, and a function to retrieve the information..

{‘name’:’Ali’, ‘Age’:’45}

|  |
| --- |
|  |

**Question C22:**

You need to create a mobile application using flutter firestore database. Here is the description of the application.

Customer:

A module to contain customers information that can purchase products from the application

Products:

A module of products. Each product can have multiple pictures uploaded to fire store file storage.

Order:

A module that contains orders for various products, the orders are placed by the customers.

Order Details:

A module that contains order id, and information about products purchased by a customer.

You need to:

Develop a customer module where a customer can place various orders. The products should be maintained in a shopping cart. On checkout a textinput will be provided to enter a fake card info.

A seller module where he can add products and can see the orders placed by the customer. The seller should be able to complete the orders as we usually see in e-commerce applications.

|  |
| --- |
|  |

**Question C14\*:**

Use the tab navigation or drawer navigation to show menu for the assignment.

You need to store and retrieve data from Firestore database.

The user will perform input of data in the following screen. When the submit button is clicked, the data is uploaded in firestore as a new document in the collection “persons”. You may also need subcollections for subjects and skills.

**INPUT DATA SCREEN**

Submit

Email

Name

Gender

Male

Female

Country

Skills

Designing

Backend

Database

Subjects

Phy

Chem

Bio

Address

**DISPLAY ALL DATA SCREEN**

Use flat list or some other list to show the data of all persons, in the following format:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Email** | **Name** | **Country** |
| Select | ali@gmail.com | Ali Khan | Pakistan |
| Select | noman@gmail.com | Noman Ali | Afghanistan |

When the user click on “select” against any record, the data is displayed in the following manner. (Pass the email to the next screen, where the record will be fetched from firestore against the email and showed in display screen).

**DISPLAY SINGLE DATA**

Edit

ali@gmail.com

Email

Ali Khan

Name

Gender

Male

Country

Pakistan

Skills

Designing, Database

Subjects

Phy,

Bio

Address

Lahore, Gulberg

Delete

When the user press delete button, the record should be deleted (after a confirmation alert) and the user should be directed back to the display all data screen.

When user presses on Edit button, the edit data screen should appear where his name and address should be in edit mode. (again, pass the email to edit screen and fetch record from db)

**UPDATE DATA SCREEN**

Update

ali@gmail.com

Email

Name

Address

Lahore, Gulberg

Back

 Ali Khan

When user clicks on Update, the record is updated. When the user click on Back, he is directed back to the display single data screen.

**SEARCH SCREEN.**

Search

Email

ali@gmail.com

When the user click on search button, the email is passed to the display single data screen, where the user’s record is displayed.

|  |
| --- |
|  |

**Question C19**

We have the following layout with a delete button.

25

ID:

Delete

Write a method that deletes the document from firestore database whose id is 25.

|  |
| --- |
| import 'package:flutter/foundation.dart';  import 'package:flutter/material.dart';  import 'package:provider/provider.dart';  void main() {  runApp(  ChangeNotifierProvider(  create: (context) => AppState(),  child: const MyApp(),  ),  );  }  class MyApp extends StatelessWidget {  const MyApp({super.key});  @override  Widget build(BuildContext context) {  return const MaterialApp(  home: HomeScreen(),  );  }  }  class AppState extends ChangeNotifier {  String \_email = '';  String \_name = '';  String get email => \_email;  String get name => \_name;  void setEmail(String email) {  \_email = email;  notifyListeners();  }  void setName(String name) {  \_name = name;  notifyListeners();  }  }  class HomeScreen extends StatefulWidget {  const HomeScreen({super.key});  @override  State<HomeScreen> createState() => \_HomeScreenState();  }  class \_HomeScreenState extends State<HomeScreen> {  final TextEditingController \_emailController = TextEditingController();  final TextEditingController \_nameController = TextEditingController();  @override  Widget build(BuildContext context) {  return Scaffold(  appBar: AppBar(  title: const Text('Home Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  children: [  TextField(  controller: \_emailController,  decoration: const InputDecoration(labelText: 'Email'),  ),  TextField(  controller: \_nameController,  decoration: const InputDecoration(labelText: 'Name'),  ),  const SizedBox(height: 20),  ElevatedButton(  onPressed: () {  Provider.of<AppState>(context, listen: false)  .setEmail(\_emailController.text);  Provider.of<AppState>(context, listen: false)  .setName(\_nameController.text);  Navigator.push(  context,  MaterialPageRoute(  builder: (context) => const ProfileScreen()),  );  },  child: const Text('Submit'),  ),  ],  ),  ),  );  }  }  class ProfileScreen extends StatelessWidget {  const ProfileScreen({super.key});  @override  Widget build(BuildContext context) {  final appState = Provider.of<AppState>(context);  return Scaffold(  appBar: AppBar(  title: const Text('Profile Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text('Email: ${appState.email}',  style: const TextStyle(fontSize: 20)),  Text('Name: ${appState.name}',  style: const TextStyle(fontSize: 20)),  ElevatedButton(  onPressed: () {  Provider.of<AppState>(context, listen: false)  .setName("THIS IS UPDATED NAME");  },  child: const Text("Update Name")),  ElevatedButton(  onPressed: () {  Navigator.push(  context,  MaterialPageRoute(builder: (context) => const ResultScreen()),  );  },  child: const Text('Submit'),  ),  ],  ),  ),  );  }  }  class ResultScreen extends StatelessWidget {  const ResultScreen({super.key});  @override  Widget build(BuildContext context) {  final appState = Provider.of<AppState>(context);  return Scaffold(  appBar: AppBar(  title: const Text('Result Screen'),  ),  body: Padding(  padding: const EdgeInsets.all(16.0),  child: Column(  crossAxisAlignment: CrossAxisAlignment.start,  children: [  Text('Email: ${appState.email}',  style: const TextStyle(fontSize: 20)),  Text('Name: ${appState.name}',  style: const TextStyle(fontSize: 20)),  ],  ),  ),  );  }  } |

**Question C20:**

Consider the following case study. An online shop has multiple products. Multiple customers are registered with the shop. Each customer can place multiple orders. Each order can contain multiple products purchased by the customer. The shop owner wants to maintain a complete history of what products in what quantities on what dates were purchased by which customers. Create a firestore schema to represent the above database (consisting of collections and/or subcollections).

|  |
| --- |
|  |

**Question C21:**

We have a following layout:

25

ID:

INSERT

Ali

NAME:

Supply, Abbottabad

ADDRESS:

Write a method that inserts value in a firestore database collection “persons”.

|  |
| --- |
|  |

**Question C29:**

We have a following layout:

25

ID:

UPDATE

Ali

NAME:

Supply, Abbottabad

ADDRESS:

Write a method that update value in database against ID = 25 using firestore

|  |
| --- |
|  |

**Question C23:**

Suppose we have the following search form:

ID:

SEARCH

NAME:

ADDRESS:

Based on the search fields in the above form, create a compound query to search record in a firestore collection “persons”.

**Question C24:**

You need to insert the following document in a collection name “students” in firestore database

Name: Ali Khan

Address:

{province: ‘punjab’, city: ‘lahore’}

1. Update the city of student from Lahore to Rawalpindi,

Solution:

**Question C25:**

You have a cities collection in firestore database. You need to select cities using compound query such that state of city is ABC, country is PQR, and population is greater than 1000.

You need to fetch record from 100th row and select the next 50 records.

Suppose you have a collection in a Firestore database storing the following values.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ID** | **Name** | **CGPA** |
| View | 1 | Javed | 3.0 |
| View | 2 | Noman | 2.7 |
| View | 3 | Ali | 3.7 |

When a user clicks on View button against any record, the user should be navigated to a new screen showing the individual record of the user in console:

**Question C26:**

Suppose View button against ID=2 is clicked, the new screen should be showing:  
  
ID: 2  
Name: Noman  
CGPA: 2.7  
  
You need to:

(a) Write code to display the Flatlist

(b) Write code of the function component displaying individual user’s values.

**Question C27:**

Consider the following case study. An online shop has multiple products. Multiple customers are registered with the shop. Each customer can place multiple orders. Each order can contain multiple products purchased by the customer. The shop owner wants to maintain a complete history of what products in what quantities on what dates were purchased by which customers. Create a firestore schema to represent the above database (consisting of collections and/or subcollections).

**Question C28:**

You need to create a web / mobile application using flutter and PHP. Here is the description of the application.

Customer:

A table to contain customers information that can purchase products from the website

Products:

A table of products

Order:

A table that contains orders for various products, the orders are placed by the customers.

Order Details:

A table that contains order id, product id as foreign keys and stores which products are ordered by a customer in a particular order.

You need to:

Develop a customer module where a customer can place various orders. The products should be maintained in a shopping cart. On checkout a textinput will be provided to enter a fake card info.

A seller module where he can add products and can see the orders placed by the customer. The seller should be able to complete the orders as we usually see in e-commerce applications.

# FLUTTER ADVANCED TOPICS (CLO-3)

**Question D3:**

Take a picture from your mobile using camera SDK, and upload on Firestore database. Also store the name of the person whose picture is taken.

**Question D4:**

Create an app that shows the current GPS coordinates on the button click.

**Question D5:**

Create a GPS tracker app by attaching a listener, so that when the listener is running, the GPS coordinates are fetched after every few seconds automatically and stored in a file. To store values in a file using flutter, follow this URL:

https://docs.flutter.dev/cookbook/persistence/reading-writing-files