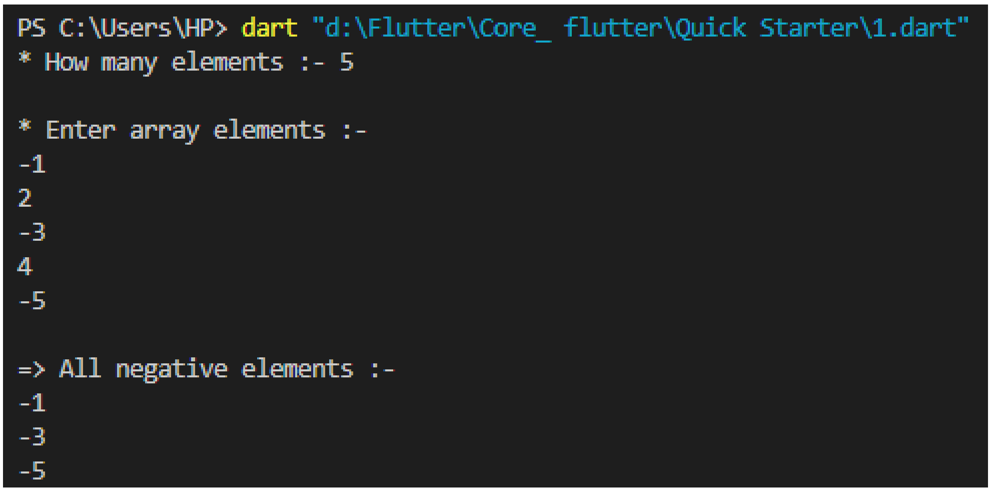
Project - 1

Aim: Write a dart program to print all negative elements in an array.

Program:

```
import 'dart:io';

void main() {
  var a = [];
  int c = 0;

  stdout.write("* How many elements :- ");
  int n = int.parse(stdin.readLineSync()!);

  print("\n* Enter array elements :- ");
  for (int i = 0; i < n; i++) {
    a.add(int.parse(stdin.readLineSync()!));
  }

  print("\n=> All negative elements :- ");
  for (int i = 0; i < n; i++) {
    if (a[i] < 0) {
      print(a[i]);
    } else {
      c++;
    }
  }
  if (c == 0) {
    print("\n- No negative elements in this array...\n");
  }
}
```

Output:



```
PS C:\Users\HP> dart "d:\Flutter\Core_ flutter\Quick Starter\1.dart"
* How many elements :- 5

* Enter array elements :-
-1
2
-3
4
-5

=> All negative elements :-
-1
-3
-5
```

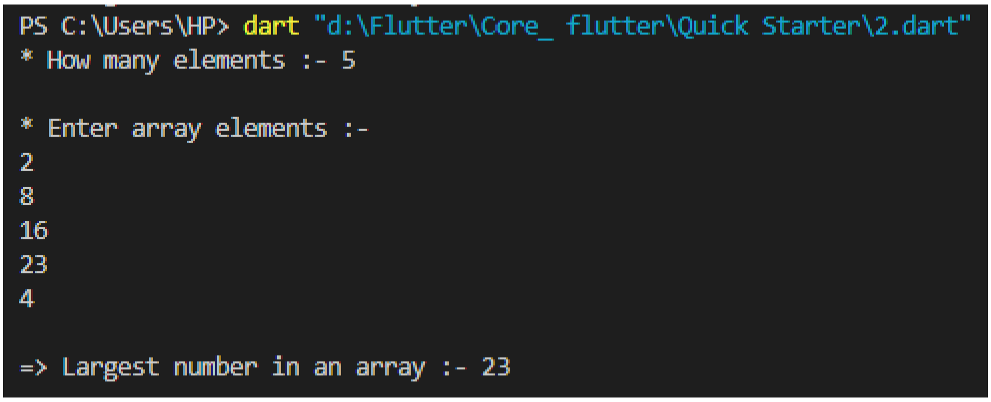
Project - 2

Aim: Write a dart program to find largest number in an array.

Program:

```
import 'dart:io';

void main() {
  var a = [];

  stdout.write("* How many elements :- ");
  int n = int.parse(stdin.readLineSync()!);

  print("\n* Enter array elements :- ");
  for (int i = 0; i < n; i++) {
    a.add(int.parse(stdin.readLineSync()!));
  }

  stdout.write("\n=> Largest number in an array :- ");
  for (int i = 0; i < n; ++i) {
    if (a[0] < a[i]) {
      a[0] = a[i];
    }
  }
  print(a[0]);
  print("\n");
}
```

Output:



```
PS C:\Users\HP> dart "d:\Flutter\Core_ flutter\Quick Starter\2.dart"
* How many elements :- 5

* Enter array elements :-
2
8
16
23
4

=> Largest number in an array :- 23
```

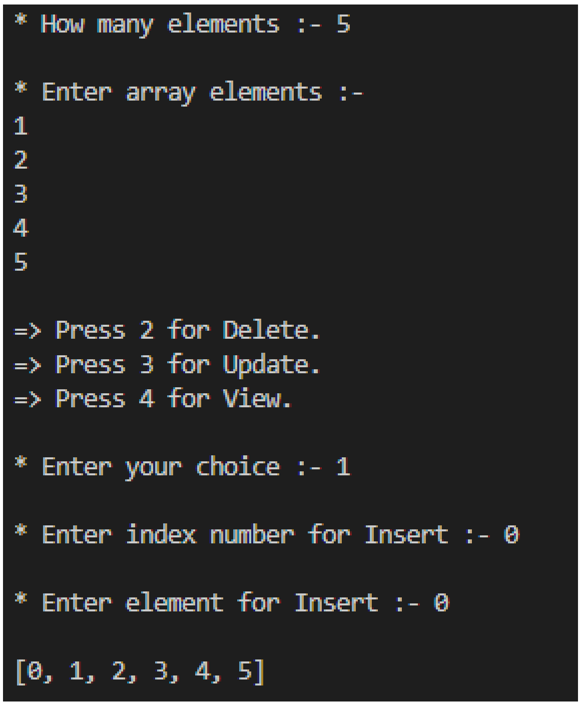
Project - 3

Aim: Write a dart program to Insert, Delete, Update and View operations on the elements in an array using menu-driven programming approach.

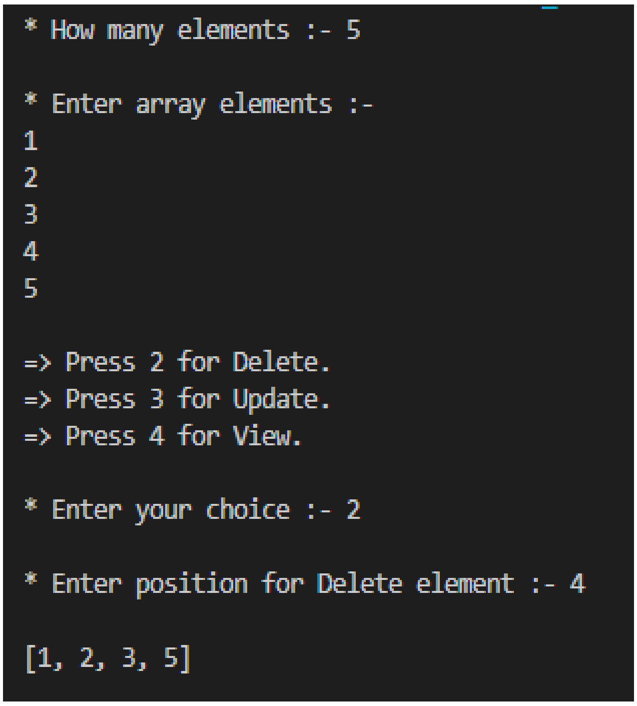
Program:

```
import 'dart:io';

void main() {
  var a = [];
  int x, y;

  stdout.write("* How many elements :- ");
  int n = int.parse(stdin.readLineSync()!);

  print("\n* Enter array elements :- ");
  for (int i = 0; i < n; i++) {
    a.add(int.parse(stdin.readLineSync()!));
  }

  print("\n=> Press 1 for Insert.");
  print("=> Press 2 for Delete.");
  print("=> Press 3 for Update.");
  print("=> Press 4 for View.");

  stdout.write("\n* Enter your choice :- ");
  int choice = int.parse(stdin.readLineSync()!);

  switch (choice) {
    case 1:
      stdout.write("\n* Enter index number for Insert :- ");
      x = int.parse(stdin.readLineSync()!);

      stdout.write("\n* Enter element for Insert :- ");
      y = int.parse(stdin.readLineSync()!);

      a.insert(x, y);

      print("\n$a\n");
      break;
    case 2:
      stdout.write("\n* Enter position for Delete element :- ");
      x = int.parse(stdin.readLineSync()!);

      a.remove(x);

      print("\n$a\n");
      break;

    case 3:
      stdout.write("\n* Enter index number for Update element :- ");
      x = int.parse(stdin.readLineSync()!);

      stdout.write("\n* Enter element for Update element :- ");
      y = int.parse(stdin.readLineSync()!);

      a.forEach((val) {
        val = x;
        a[x] = y;
      });
      print("\n$a\n");
      break;

    case 4:
      stdout.write("\n* Enter index number for view element :- ");
      x = int.parse(stdin.readLineSync()!);
      print("\n${a[x]}\n");
      break;

    default:
      print("\n=> Invalid Choice....\n");
      break;
  }
}
```

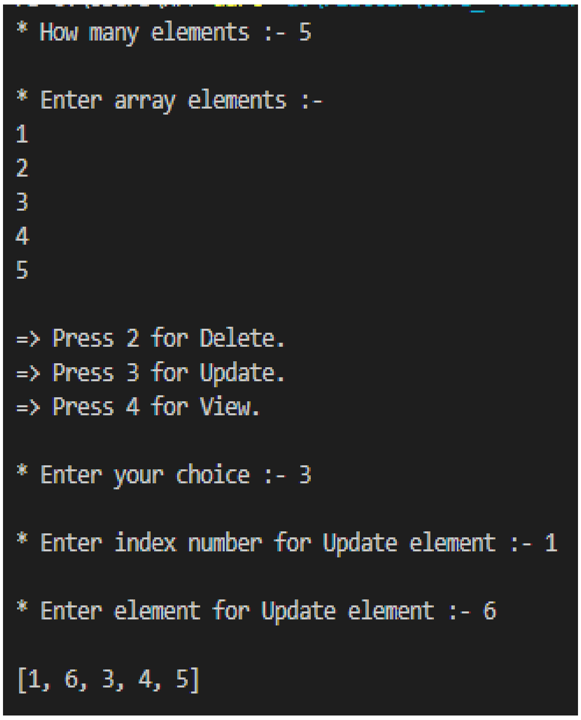
Output:



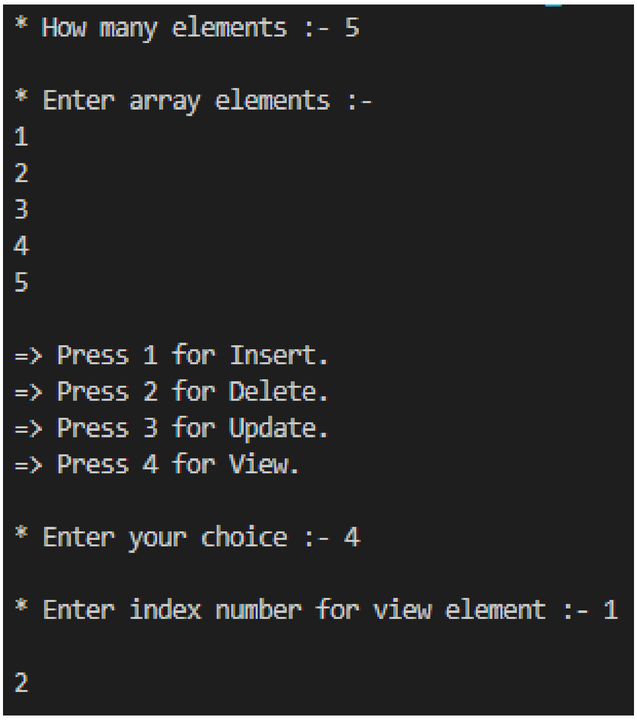
```
* How many elements :- 5

* Enter array elements :-
1
2
3
4
5

=> Press 2 for Delete.
=> Press 3 for Update.
=> Press 4 for View.

* Enter your choice :- 1

* Enter index number for Insert :- 0

* Enter element for Insert :- 0

[0, 1, 2, 3, 4, 5]
```



```
* How many elements :- 5

* Enter array elements :-
1
2
3
4
5

=> Press 2 for Delete.
=> Press 3 for Update.
=> Press 4 for View.

* Enter your choice :- 2

* Enter position for Delete element :- 4

[1, 2, 3, 5]
```



```
* How many elements :- 5

* Enter array elements :-
1
2
3
4
5

=> Press 2 for Delete.
=> Press 3 for Update.
=> Press 4 for View.

* Enter your choice :- 3

* Enter index number for Update element :- 1

* Enter element for Update element :- 6

[1, 6, 3, 4, 5]
```



```
* How many elements :- 5

* Enter array elements :-
1
2
3
4
5

=> Press 1 for Insert.
=> Press 2 for Delete.
=> Press 3 for Update.
=> Press 4 for View.

* Enter your choice :- 4

* Enter index number for view element :- 1

2
```

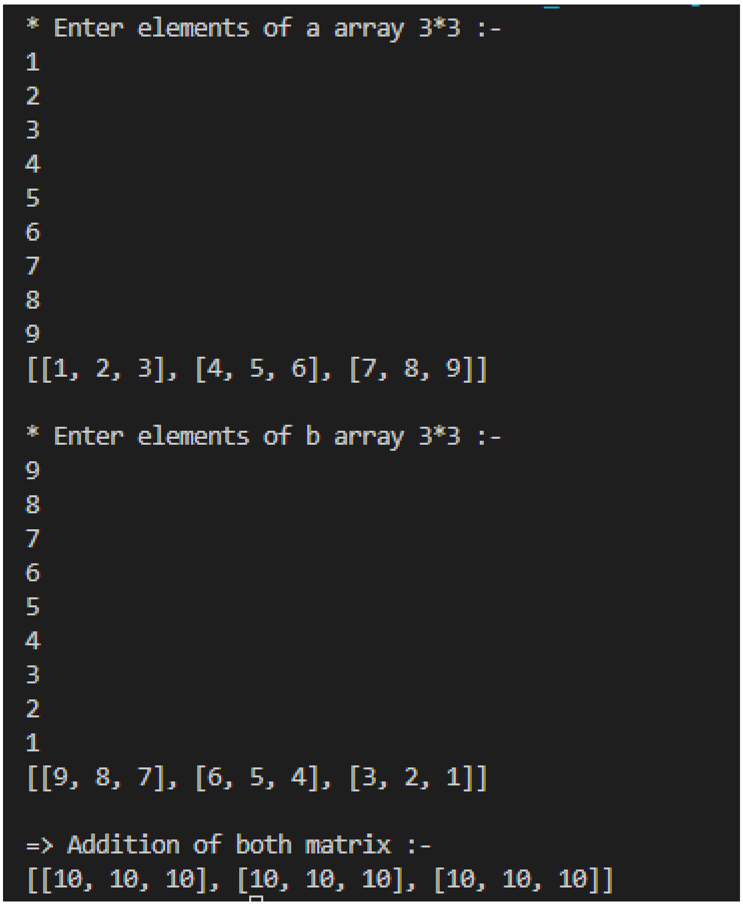
Project - 4

Aim: Write a Dart program to addition of two matrices of dimension 3*3.

Program:

```
import 'dart:io';

void main() {
  int n = 3;
  int m = 3;

  print("* Enter elements of a array 3*3 :- ");
  var a = List.generate(
      n, (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
  print(a);
  print("\n* Enter elements of b array 3*3 :- ");
  var b = List.generate(
      n, (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
  print(b);
  print("\n=> Addition of both matrix :- ");
  var c = List.generate(n, (i) => List.generate(m, (j) => a[i][j] + b[i][j]));
  print(c);
}
```

Output:



```
* Enter elements of a array 3*3 :-
1
2
3
4
5
6
7
8
9
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]

* Enter elements of b array 3*3 :-
9
8
7
6
5
4
3
2
1
[[9, 8, 7], [6, 5, 4], [3, 2, 1]]

=> Addition of both matrix :-
[[10, 10, 10], [10, 10, 10], [10, 10, 10]]
```

Project - 5

Aim: Write a Dart program to find below mentioned operations on the 3x3 array.
using a menu-driven programming approach.
Perform below all mentioned by user choice:

- **Sum of all elements**
- **Sum of specific Row**
- **Sum of specific Column**
- **Sum of diagonal elements**
- **Sum of antidiagonal elements**
- **Press 0 for exit**

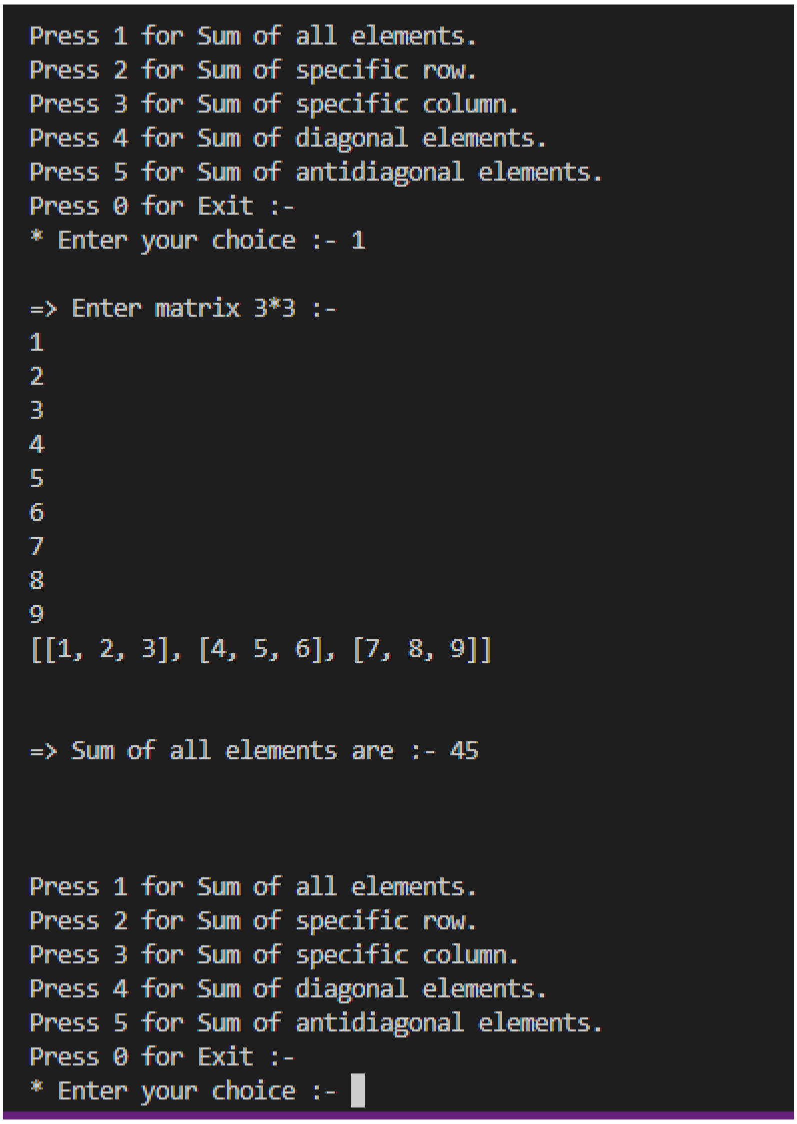
Program:

```
import 'dart:io';

void main() {
  int ch;
  int n = 3;
  int m = 3;
  int sum = 0;
  int r = 0;

  do {
    stdout.write("""\n\n\n
Press 1 for Sum of all elements.
Press 2 for Sum of specific row.
Press 3 for Sum of specific column.
Press 4 for Sum of diagonal elements.
Press 5 for Sum of antidiagonal elements.
Press 0 for Exit :- """);

    stdout.write("\n* Enter your choice :- ");
    ch = int.parse(stdin.readLineSync()!);

    switch (ch) {
      case 0:
        break;
      case 1:
```

```
      print("\n=> Enter matrix 3*3 :- ");

      var ascList = List.generate(n,
          (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
      print(ascList);

      stdout.write("\n\n=> Sum of all elements are :- ");

      var bList = List.generate(
          n, (i) => List.generate(m, (j) => sum = sum + ascList[i][j]));

      stdout.write(sum);
      break;

    case 2:
      stdout.write("\n=> Enter matrix 3*3 :-");

      var ascList = List.generate(n,
          (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
      print(ascList);

      stdout.write("\n\nsum of rows ::");
      int i = int.parse(stdin.readLineSync()!);
      var bList = List.generate(
          n, (i) => List.generate(m, (j) => r += ascList[i][j]));
      for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
          sum += ascList[i][j];
          r += ascList[i][j];
        }
        print("sum of ${i + 1} row is $r ");
      }

      break;

    case 3:
      stdout.write("enter matrix 3*3 ::");

      var ascList = List.generate(n,
          (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
      print(ascList);

      stdout.write("\n\nsum of coloumns ::");
      int i = int.parse(stdin.readLineSync()!);
    var bList = List.generate(
        n, (i) => List.generate(m, (j) => r += ascList[i][j]));
    for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
        sum += ascList[i][j];
        r += ascList[j][i];
      }
      print("sum of ${i + 1} column is $r ");
    }
    break;

  case 4:
    stdout.write("enter matrix 3*3 ::");

    var ascList = List.generate(n,
        (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
    print(ascList);

    stdout.write("\n\nSum of diagonal elements are :: ");

    var bList = List.generate(
        n,
        (i) => List.generate(
            m, (j) => (i == j) ? {sum += ascList[i][j]}! : {" "}));

    for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
        if (i == j) {
          sum += ascList[i][j];
        }
      }
    }
    stdout.write(sum);
    break;

  case 5:
    stdout.write("enter matrix 3*3 ::");

    var ascList = List.generate(n,
        (i) => List.generate(m, (j) => int.parse(stdin.readLineSync()!)));
    print(ascList);

    stdout.write("\n\nSum of diagonal elements are :: ");
```

```
        var bList = List.generate(
            n,
            (i) => List.generate(
                m, (j) => (i == j) ? {sum += ascList[i][j]}! : {" "}));

        for (int i = 0; i < n; i++) {
          for (int j = 0; j < n; j++) {
            if (i + j == 2) {
              sum += ascList[i][j];
            }
          }
        }
        stdout.write(sum);

        break;
      default:
        stdout.write("Put valid choice :::");
    }
  } while (ch != 0);
}
```

Output:



```
Press 1 for Sum of all elements.
Press 2 for Sum of specific row.
Press 3 for Sum of specific column.
Press 4 for Sum of diagonal elements.
Press 5 for Sum of antidiagonal elements.
Press 0 for Exit :-
* Enter your choice :- 1

=> Enter matrix 3*3 :-
1
2
3
4
5
6
7
8
9
[[1, 2, 3], [4, 5, 6], [7, 8, 9]]


=> Sum of all elements are :- 45



Press 1 for Sum of all elements.
Press 2 for Sum of specific row.
Press 3 for Sum of specific column.
Press 4 for Sum of diagonal elements.
Press 5 for Sum of antidiagonal elements.
Press 0 for Exit :-
* Enter your choice :-
```

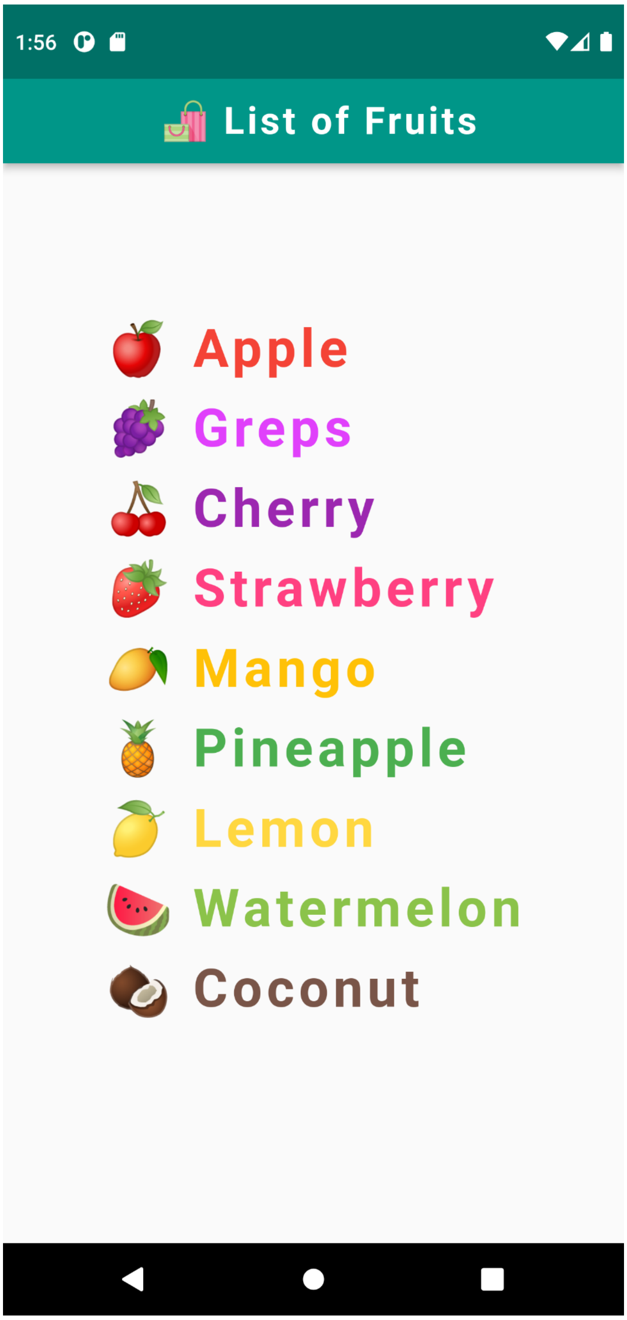
Task- 1

Aim: Create an app as shown in below image.

Program:

```
import 'package:flutter/material.dart';

void main() {
  runApp(
    MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          title: const Text(" 🛍️ List of Fruits"),
          centerTitle: true,
          backgroundColor: Colors.teal,
          titleTextStyle: const TextStyle(
            fontSize: 25,
            fontWeight: FontWeight.bold,
            letterSpacing: 2,
          ),
        ),
        body: Center(
          child: RichText(
            text: const TextSpan(
                text: "🍎 Apple\n",
                style: TextStyle(
                  fontSize: 35,
                  color: Colors.red,
                  height: 1.5,
                  fontWeight: FontWeight.bold,
                  letterSpacing: 3,
                ),
                children: [
                  TextSpan(
                    text: "🍇 Greps\n",
                    style: TextStyle(
                      color: Colors.purpleAccent,
                    ),
                  ),
        TextSpan(
          text: "🍒 Cherry\n",
          style: TextStyle(
            color: Colors.purple,
          ),
        ),
        TextSpan(
          text: "🍓 Strawberry\n",
          style: TextStyle(
            color: Colors.pinkAccent,
          ),
        ),
        TextSpan(
          text: "🥭 Mango\n",
          style: TextStyle(
            color: Colors.amber,
          ),
        ),
        TextSpan(
          text: "🍍 Pineapple\n",
          style: TextStyle(
            color: Colors.green,
          ),
        ),
        TextSpan(
          text: "🍋 Lemon\n",
          style: TextStyle(
            color: Colors.amberAccent,
          ),
        ),
        TextSpan(
          text: "🍉 Watermelon\n",
          style: TextStyle(
            color: Colors.lightGreen,
          ),
        ),
        TextSpan(
          text: "🥥 Coconut\n",
          style: TextStyle(
            color: Colors.brown,
          ),
        ),
      ]),
),
          ),
        ),
      ),
    );
}
```

Output:



1:56

🛍️ List of Fruits

🍎 Apple

🍇 Greps

🍒 Cherry

🍓 Strawberry

🥭 Mango

🍍 Pineapple

🍋 Lemon

🍉 Watermelon

🥥 Coconut

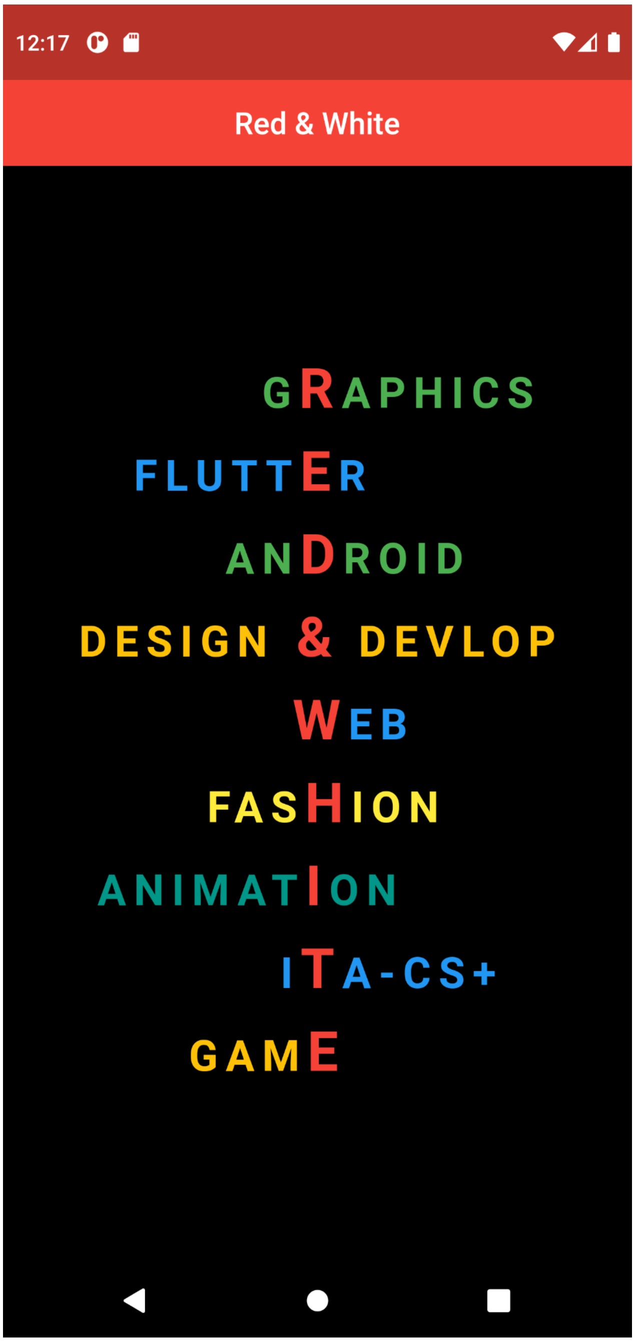
Task- 2

Aim: Create an app as shown in below image.

Program:

```
import 'package:flutter/material.dart';

void main() {
  runApp(
    MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        backgroundColor: Colors.black,
        appBar: AppBar(
          title: const Text("Red & White"),
          centerTitle: true,
          backgroundColor: Colors.red,
        ),
        body: Center(
          child: RichText(
            text: const TextSpan(
              text: "        G",
              style: TextStyle(
                fontSize: 28,
                color: Colors.green,
                fontWeight: FontWeight.bold,
                letterSpacing: 5,
                height: 1.5,
              ),
              children: [
                TextSpan(
                  text: "R",
                  style: TextStyle(
                    fontSize: 36,
                    color: Colors.red,
                  ),
                ),
                TextSpan(
                  text: "APHICS\n",
                  style: TextStyle(
    fontSize: 28,
    color: Colors.green,
  ),
),
TextSpan(
  text: "   FLUTT",
  style: TextStyle(
    fontSize: 28,
    color: Colors.blue,
  ),
),
TextSpan(
  text: "E",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: "R\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.blue,
  ),
),
TextSpan(
  text: "        AN",
  style: TextStyle(
    fontSize: 28,
    color: Colors.green,
  ),
),
TextSpan(
  text: "D",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: "ROID\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.green,
  ),
),
TextSpan(
  text: "DESIGN",
  style: TextStyle(
    fontSize: 28,
    color: Colors.amber,
  ),
),
TextSpan(
  text: " &",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: " DEVLOP\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.amber,
  ),
),
TextSpan(
  text: "          W",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: "EB\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.blue,
  ),
),
TextSpan(
  text: "       FAS",
  style: TextStyle(
    fontSize: 28,
    color: Colors.yellow,
  ),
),
TextSpan(
  text: "H",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: "ION\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.yellow,
  ),
),
TextSpan(
  text: " ANIMAT",
  style: TextStyle(
    fontSize: 28,
    color: Colors.teal,
  ),
),
TextSpan(
  text: "I",
  style: TextStyle(
    fontSize: 36,
    color: Colors.red,
  ),
),
TextSpan(
  text: "ON\n",
  style: TextStyle(
    fontSize: 28,
    color: Colors.teal,
  ),
),
TextSpan(
  text: "            I",
  style: TextStyle(
    fontSize: 28,
    color: Colors.blue,
  ),
),
TextSpan(
  text: "T",
                  style: TextStyle(
                    fontSize: 36,
                    color: Colors.red,
                  ),
                ),
                TextSpan(
                  text: "A-CS+\n",
                  style: TextStyle(
                    fontSize: 28,
                    color: Colors.blue,
                  ),
                ),
                TextSpan(
                  text: "     GAM",
                  style: TextStyle(
                    fontSize: 28,
                    color: Colors.amber,
                  ),
                ),
                TextSpan(
                  text: "E",
                  style: TextStyle(
                    fontSize: 36,
                    color: Colors.red,
                  ),
                ),
              ],
            ),
          ),
        ),
      ),
    ),
  );
}
```

Output:



12:17

Red & White

GRAPHICS
FLUTTER
ANDROID
DESIGN & DEVLOP
WEB
FASHION
ANIMATION
ITA-CS+
GAME