

NAME: MUHAMMAD MUBASHIR

FATHER NAME: SAEED AKBER

COURSE: MOBILE APPLICATION

COURSE INSTRUCTOR: SALMAN BEDIYA

Concept of Dart Programming Language

Assignment 1 Question Task

1. Check whether a given year is a leap year or not?
2. Check whether a given year is a leap year or not?
3. Check the maximum number between two numbers.
4. Check if a number is positive or negative.
5. Check whether a number is divisible by 5 and 11 or not.

Assignment 2 Question Task

6. Write a program to check if a given string is a palindrome.\
7. Write a program to calculate the factorial of a given number using a function.
8. Write a program to print out the Fibonacci sequence up to a given number.
9. Write a program to calculate the distance between two points on a 2D plane using a function.
10. Write a program to convert a temperature from Fahrenheit to Celsius using a function.
11. Write a program to calculate the area of a circle using a function.
12. Write a program to print out the prime numbers between 1 and a given number.

TASK CREATING CLASS CONCEPT

13. Create a class called "BankAccount" with the following attributes:

account_number (integer)

balance (double)

account_type (string)

interest_rate (double)

And the following methods:

deposit(amount): adds the amount to the balance.

withdraw(amount): subtracts the amount from the balance. You cannot withdraw more than the current balance.

add_interest(): adds interest to the balance based on the interest rate.

display(): prints out the account number, balance, account type, and interest rate.

Then, create two instances of the BankAccount class, each with its own account number, balance, account type, and interest rate.

Finally, call the `deposit()`, `withdraw()`, `add_interest()`, and `display()` methods on each instance and confirm that the information is updated and displayed correctly.

14.(a) Create a class called "Student" with the following attributes:

name (string)

id (string)

courses (list of strings)

And the following methods:

`add_course(course)`: adds a course to the student's list of courses.

`drop_course(course)`: removes a course from the student's list of courses.

`display_courses()`: prints out the student's list of courses.

Then, create two instances of the Student class, each with their name, id, and courses.

Finally, call the `add_course()`, `drop_course()`, and `display_courses()` methods on each instance and confirm that the information is updated and displayed correctly.

Password Generate Task

15.(a) Write a program that generates a random password based on the user's specifications. The user should be able to specify the length of the password and whether it should include numbers, letters, and special characters.

(b) Write a function that takes a string input from the user and checks if the password is strong enough based on certain criteria (e.g. minimum length, use of uppercase letters, use of special characters, etc.).

If Else Condition

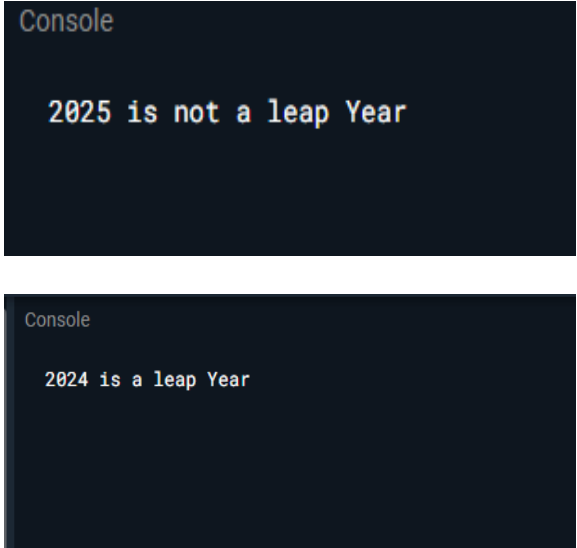
16. Write a program that takes an integer input from the user and prints out numbers from 1 to that integer, but for multiples of 3 print "Fizz" instead of the number, and for multiples of 5 print "Buzz". For numbers that are multiples of both 3 and 5, print "FizzBuzz".

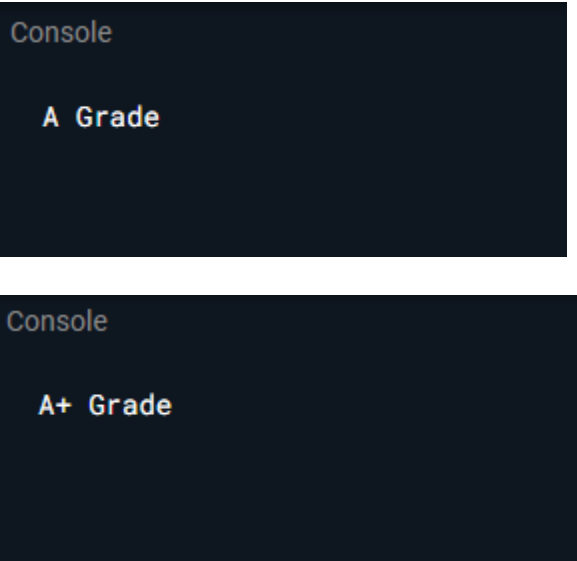
17. Write a function that takes a list of numbers as input and sorts the list in ascending order.

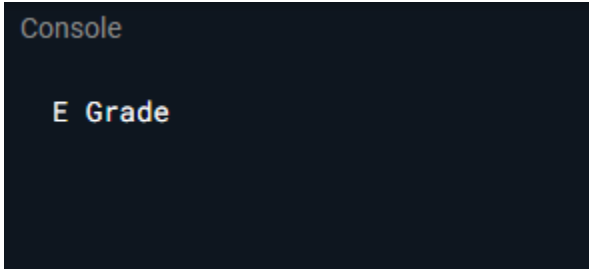
TASK INHERITANCE

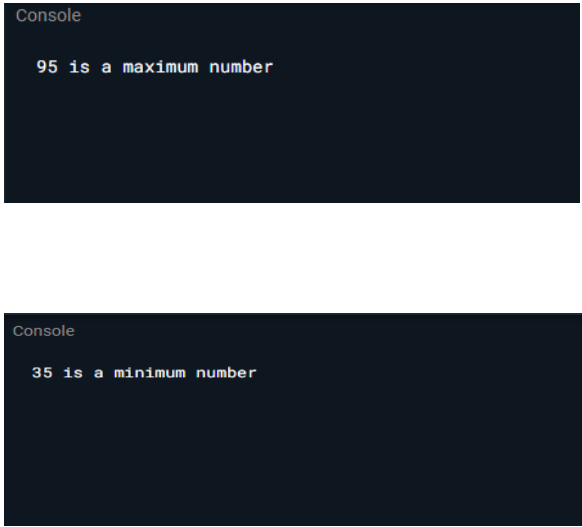
18. Calculate area of different shapes. Create a base class named "Shape" with a method to return the CalculateArea of the shape. Create a class named "Rectangle", "Circle" and "Square" derived from the base class "Shape".

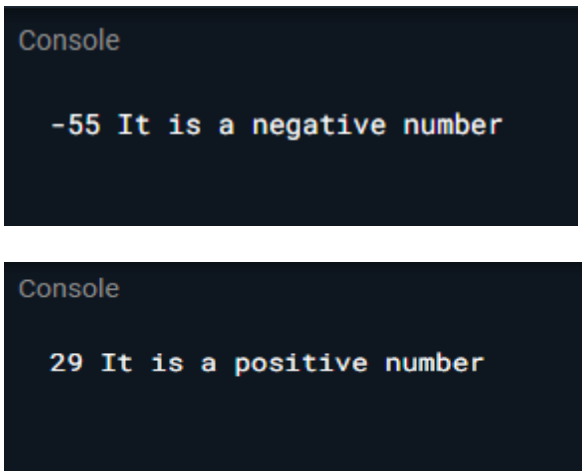
TASK 1 to 5

TASK 1	OUTPUT
<pre>//Check whether a given year is a leap year or not? Void main(){ int year = 2024 // int year = 2025; if ((year % 4 == 0 && year % 100 !=0) year % 400 == 0){ print('\$year is a leap Year'); } else{ print('\$year is not a leap Year'); } }</pre>	 <p>Console</p> <p>2025 is not a leap Year</p> <p>Console</p> <p>2024 is a leap Year</p>

TASK 2	OUTPUT
<pre>// Program to check the grade of a student from total marks with the criteria like A = >80, B = >70, etc. void main(){ // int a = 39; // int a = 89; int a = 79; if(a>80 && a<100){ print("A+ Grade"); } else if(a>70 && a<80){ print("A Grade"); } }</pre>	 <p>Console</p> <p>A Grade</p> <p>Console</p> <p>A+ Grade</p>

<pre>else if(a>60 && a<70){ print("B Grade"); } else if(a>50 && a<60){ print("C Grade"); } else if(a>40 && a<50){ print("D Grade"); } else if(a>33 && a<40){ print("E Grade"); } else if(a>0 && a<33){ print("F Grade"); } else{ print("Correct the valid input"); } }</pre>	 A screenshot of a dark-themed console window. The word "Console" is at the top in a light blue font. Below it, the text "E Grade" is displayed in a light yellow font.
---	---

TASK 3	OUTPUT
<pre>// Check the maximum number between two numbers void main(){ var a, b; a = 95; b = 37; // a = 35; // b = 87; if(a>b){ print("\$a is a maximum number"); } else if(a<b){ print("\$a is a minimum number"); } }</pre>	 <p>The output consists of two separate console windows. The first window displays the text '95 is a maximum number' in white on a dark background. The second window displays the text '35 is a minimum number' in white on a dark background.</p>

TASK 4	OUTPUT
<pre>// Check if a number is positive or negative void main() { var b = 0; var num1 = 29; // var num1 = -55; if(num1>b){ print("\$num1 It is a positive number"); } else if(num1<b){ print("\$num1 It is a negative number"); } }</pre>	 <p>The output consists of two separate console windows. The first window displays the text '-55 It is a negative number' in white on a dark background. The second window displays the text '29 It is a positive number' in white on a dark background.</p>

TASK 5	OUTPUT
<pre>// Check whether a number is divisible by 5 and 11 or not. void main() { int a, b ,c; a=5; b=11; // c = 89; c = 55; if(c % a == 0 && c % b == 0) { print("\$c Divisible by 5 and 11"); } else{ print("\$c Not Divisible by 5 and 11"); } }</pre>	<div data-bbox="812 415 1385 657"> <p>Console</p> <p>55 Divisible by 5 and 11</p> </div> <div data-bbox="812 695 1385 989"> <p>Console</p> <p>89 Not Divisible by 5 and 11</p> </div>

Assignment 2

1. Write a program to check if a given string is a palindrome.

```
void main(){
```

```
  checkpalindrome("civic")? print("its is palindrome word") :
  print("its is not palindrome word");
```

```
  checkpalindrome("hello")? print("its is palindrome word") :
```

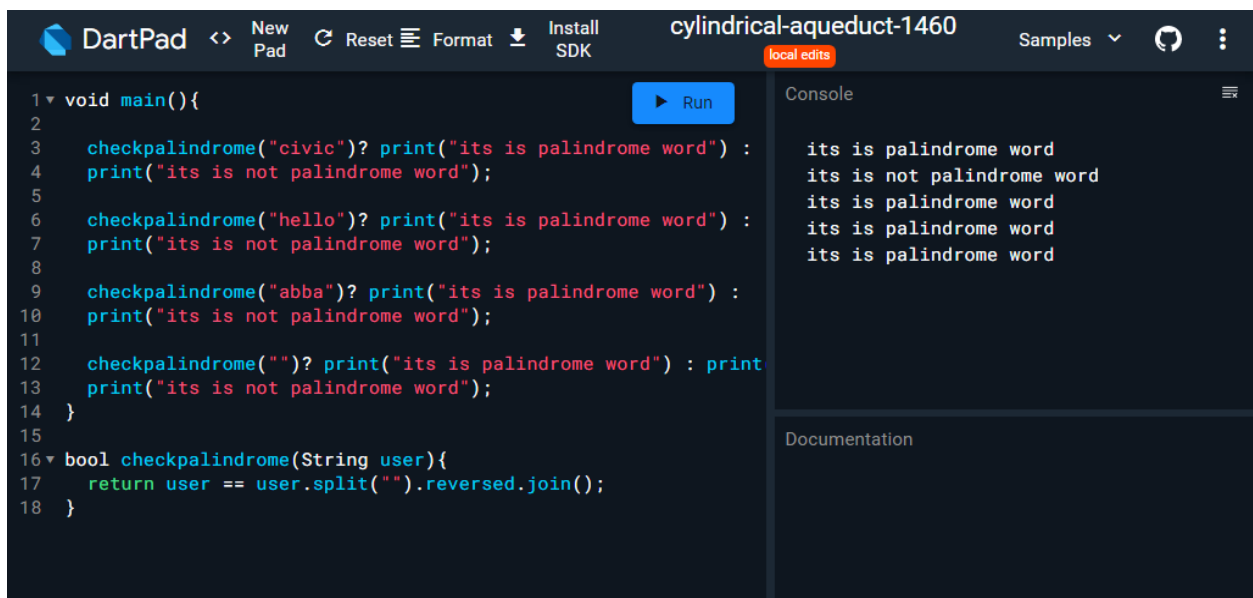
```
print("its is not palindrome word");
```

```
checkpalindrome("abba")? print("its is palindrome word") :
```

```
print("its is not palindrome word");
```

```
checkpalindrome(")? print("its is palindrome word") : print("its is not  
palindrome word"); checkpalindrome("amma")? print("its is palindrome word") :  
print("its is not palindrome word");  
}
```

```
bool checkpalindrome(String user){  
  return user == user.split("").reversed.join();  
}
```



The screenshot shows the DartPad IDE interface. The top bar includes the DartPad logo, navigation icons, and a title bar with the name 'cylindrical-aqueduct-1460'. Below the title bar, there are buttons for 'New Pad', 'Reset', 'Format', and 'Install SDK'. The main editor area contains the following Dart code:

```
1 void main(){  
2  
3   checkpalindrome("civic")? print("its is palindrome word") :  
4   print("its is not palindrome word");  
5  
6   checkpalindrome("hello")? print("its is palindrome word") :  
7   print("its is not palindrome word");  
8  
9   checkpalindrome("abba")? print("its is palindrome word") :  
10  print("its is not palindrome word");  
11  
12  checkpalindrome(")? print("its is palindrome word") : print  
13  print("its is not palindrome word");  
14 }  
15  
16 bool checkpalindrome(String user){  
17   return user == user.split("").reversed.join();  
18 }
```

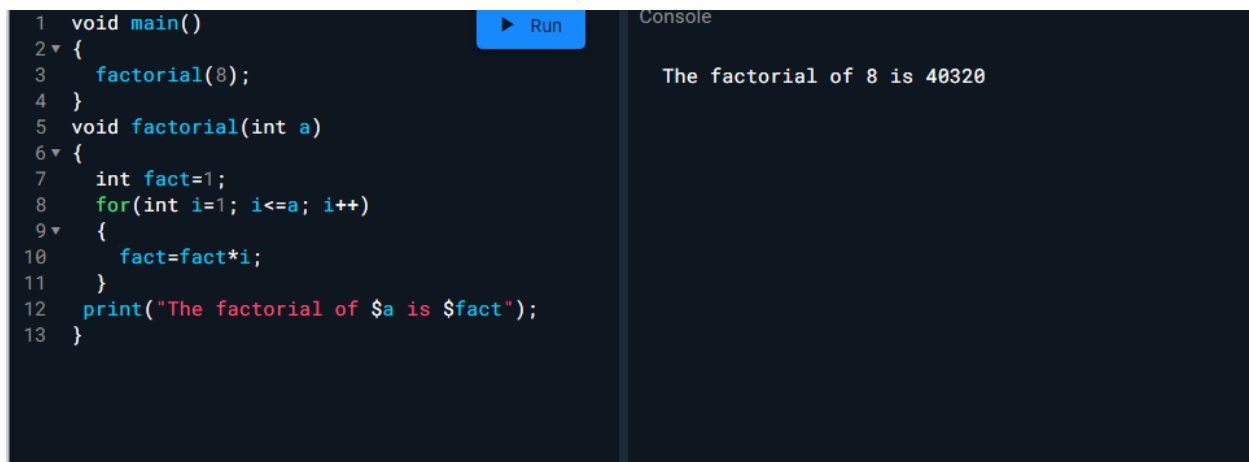
The console output on the right shows the results of the program execution:

```
its is palindrome word  
its is not palindrome word  
its is palindrome word  
its is palindrome word  
its is palindrome word
```


2. Write a program to calculate the factorial of a given number using a function.

```
void main()
{
    factorial(8);
}

void factorial(int a)
{
    int fact=1;
    for(int i=1; i<=a; i++){
        fact=fact*i;
    }
    print("The factorial of $a is $fact");
}
```



The screenshot shows a code editor with the following Dart code:

```
1 void main()
2 {
3     factorial(8);
4 }
5 void factorial(int a)
6 {
7     int fact=1;
8     for(int i=1; i<=a; i++){
9     {
10        fact=fact*i;
11    }
12    print("The factorial of $a is $fact");
13 }
```

A blue "Run" button is visible next to the code. To the right, the "Console" panel displays the output:

```
The factorial of 8 is 40320
```

3. Write a program to print out the Fibonacci sequence up to a given number.

```
void main() {  
  
    fibonacci(55);  
}  
  
void fibonacci(int a){  
    int b = 0;  
    int c = 1;  
    int d;  
  
    print('Fibonacci sequence up to $a:');  
    print(b);  
    print(c);  
  
    for (int i = 2; i <=a; i++) {  
        d = b + c;  
        if (c > a) {  
            break;  
        }  
        print(c);  
        b = c;  
        c = d;  
    }  
}
```

```
1 void main() {  
2  
3   fibonacci(55);  
4 }  
5  
6 void fibonacci(int a){  
7   int b = 0;  
8   int c = 1;  
9   int d;  
10  
11   print('Fibonacci sequence up to $a:');  
12   print(b);  
13   print(c);  
14  
15   for (int i = 2; i <=a; i++) {  
16     d = b + c;  
17     if (c > a) {  
18       break;  
19     }  
20     print(c);  
21     b = c;  
22     c = d;  
23   }  
24 }
```

Run

Console

Fibonacci sequence up to 55:

0
1
1
1
2
3
5
8
13
21
34
55

Documentation

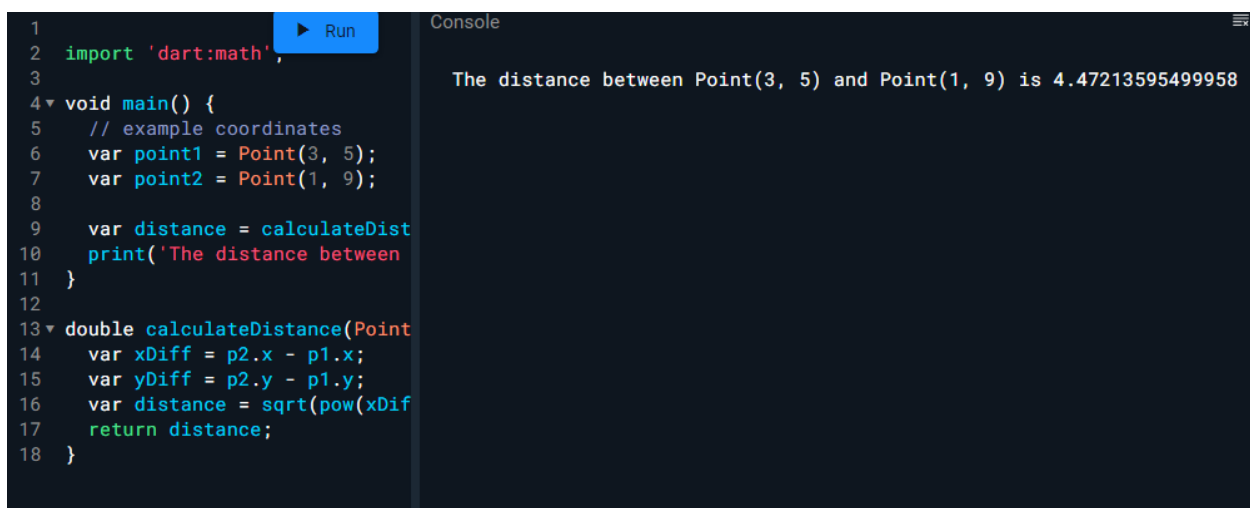
4. Write a program to calculate the distance between two points on a 2D plane using a function.

```
import 'dart:math';

void main() {
  // example coordinates
  var point1 = Point(3, 5);
  var point2 = Point(1, 9);

  var distance = calculateDistance(point1, point2);
  print('The distance between $point1 and $point2 is $distance');
}

double calculateDistance(Point p1, Point p2) {
  var xDiff = p2.x - p1.x;
  var yDiff = p2.y - p1.y;
  var distance = sqrt(pow(xDiff, 2) + pow(yDiff, 2));
  return distance;
}
```

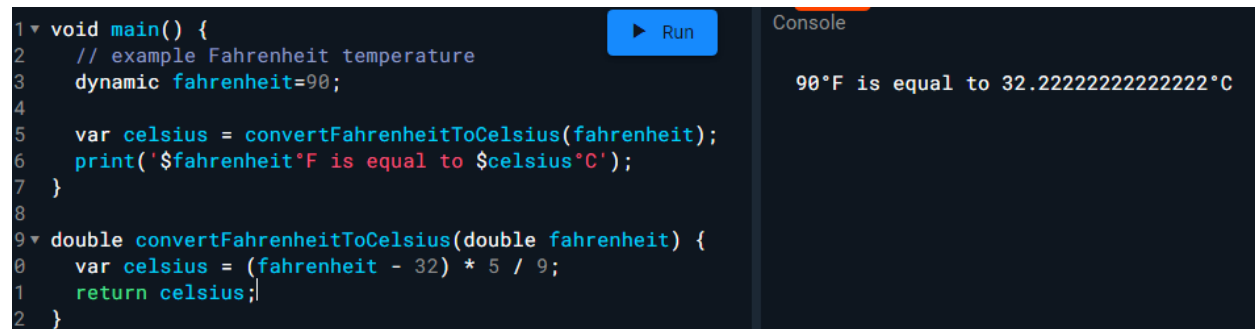


The screenshot shows a Dart IDE with a code editor on the left and a console on the right. The code editor contains the same Dart code as shown in the previous block. A blue 'Run' button is visible above the code. The console on the right displays the output of the program: 'The distance between Point(3, 5) and Point(1, 9) is 4.47213595499958'.

5. Write a program to convert a temperature from Fahrenheit to Celsius using a function

```
void main() {  
    // example Fahrenheit temperature  
    dynamic fahrenheit=90;  
  
    var celsius = convertFahrenheitToCelsius(fahrenheit);  
    print('$fahrenheit°F is equal to $celsius°C');  
}
```

```
double convertFahrenheitToCelsius(double fahrenheit) {  
    var celsius = (fahrenheit - 32) * 5 / 9;  
    return celsius;  
}
```



The screenshot shows a Dart IDE with a code editor on the left and a console on the right. The code in the editor is the same as the one shown in the previous blocks. A blue 'Run' button is visible next to the code. The console on the right displays the output of the program: '90°F is equal to 32.22222222222222°C'.

```
1 void main() {  
2     // example Fahrenheit temperature  
3     dynamic fahrenheit=90;  
4  
5     var celsius = convertFahrenheitToCelsius(fahrenheit);  
6     print('$fahrenheit°F is equal to $celsius°C');  
7 }  
8  
9 double convertFahrenheitToCelsius(double fahrenheit) {  
0     var celsius = (fahrenheit - 32) * 5 / 9;  
1     return celsius;  
2 }
```

Console
90°F is equal to 32.22222222222222°C

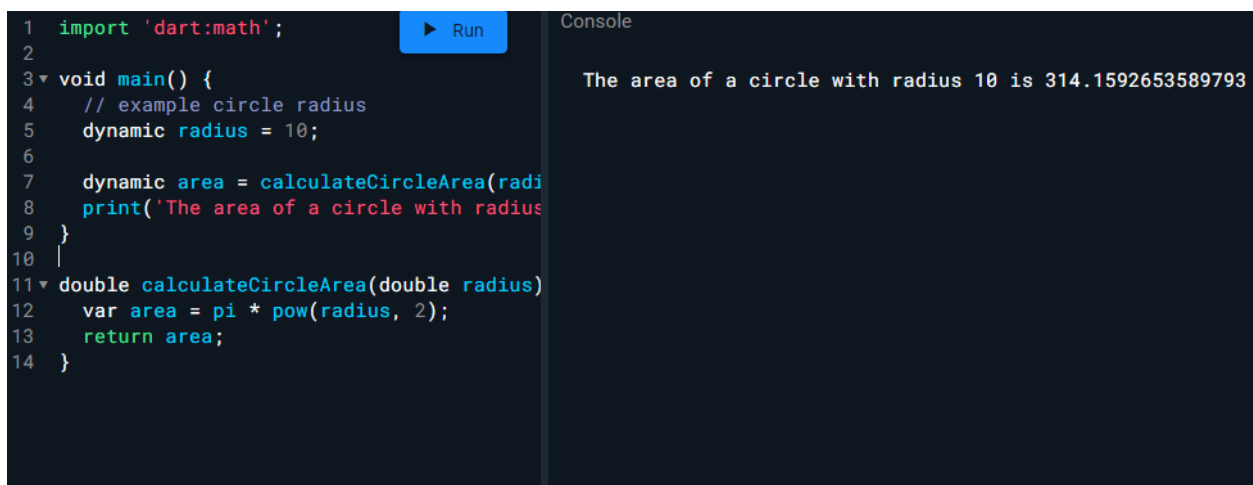
6. Write a program to calculate the area of a circle using a function.

```
import 'dart:math';

void main() {
  // example circle radius
  dynamic radius = 10;

  var area = calculateCircleArea(radius);
  print('The area of a circle with radius $radius is $area');
}

double calculateCircleArea(double radius) {
  var area = pi * pow(radius, 2);
  return area;
}
```



The screenshot shows a code editor with the Dart code from the previous block. A blue 'Run' button is visible next to the code. To the right, a 'Console' window displays the output of the program: 'The area of a circle with radius 10 is 314.1592653589793'.

7. Write a program to print out the prime numbers between 1 and a given number.

```
import 'dart:math';

void main() {
  // example upper limit
  var limit = 30;

  print('The prime numbers between 1 and $limit are:');
  for (var i = 2; i <= limit; i++) {
    if (isPrime(i)) {
      print(i);
    }
  }
}

bool isPrime(int number) {
  if (number <= 1) {
    return false;
  }

  for (var i = 2; i <= sqrt(number); i++) {
    if (number % i == 0) {
      return false;
    }
  }
}
```

```
}  
  
return true;  
  
}
```

Question:1

Create a class called "BankAccount" with the following attributes:

account_number (integer)

balance (double)

account_type (string)

interest_rate (double)

And the following methods:

deposit(amount): adds the amount to the balance.

withdraw(amount): subtracts the amount from the balance. You cannot withdraw more than the current balance.

add_interest(): adds interest to the balance based on the interest rate.

display(): prints out the account number, balance, account type, and interest rate.

Then, create two instances of the BankAccount class, each with its own account number, balance, account type, and interest rate.

Finally, call the deposit(), withdraw(), add_interest(), and display() methods on each instance and confirm that the information is updated and displayed correctly.

```
void main()  
{  
  
    BankAccount account1 = BankAccount("Saving",49516,10000,1.9);  
    BankAccount account2 = BankAccount("Saving",30293,50000,4.9);  
  
    print("First account information");  
    account1.deposit(7000);  
    account1.withdraw(2500);  
    account1.addInterest();  
    account1.display();  
  
    print("Second account information");  
    account2.deposit(9000);  
    account2.withdraw(3500);  
}
```



```
account2.addInterest();
account2.display();

}

class BankAccount{

    String accountType;
    int accountNumber;
    double balance;
    double interestRate;

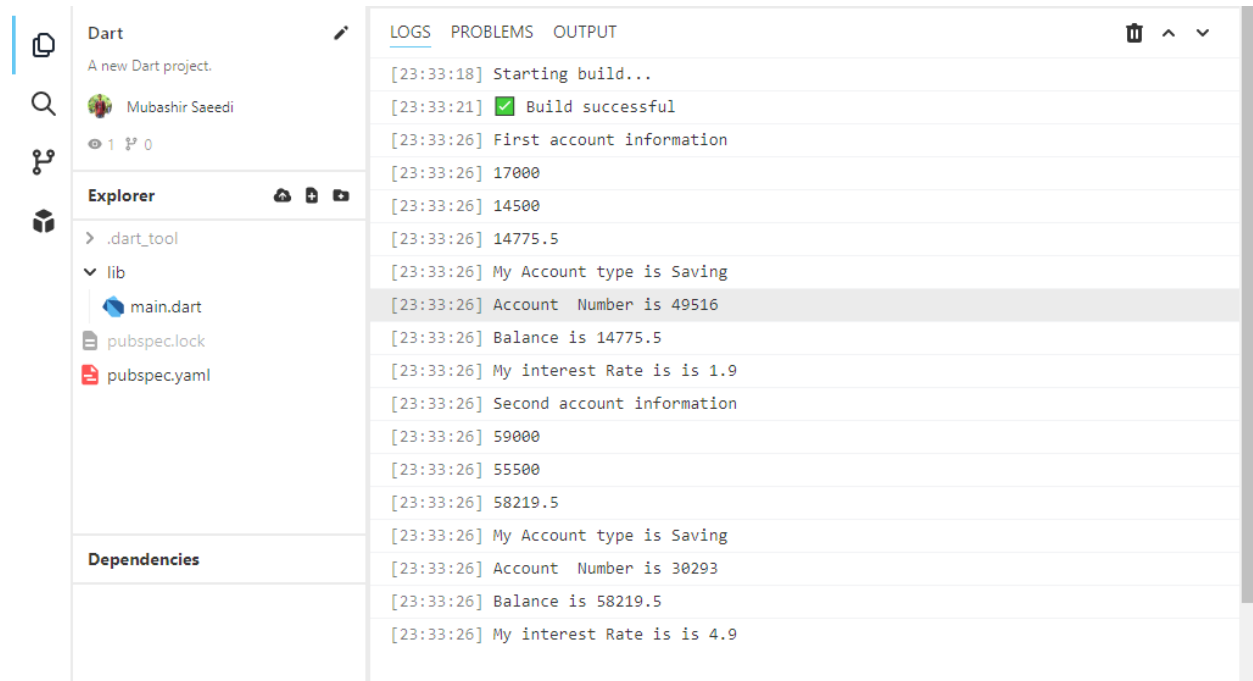
    BankAccount(this.accountType,this.accountNumber,this.balance,this.interestRate);

    void deposit(double amount)
    {
        balance = balance + amount;
        print(balance);
    }

    void withdraw(double amount)
    {
        if(amount <= balance){
            balance = balance - amount;
            print(balance);
        }
        else{
            print("You cannot withdraw more than the current balance.");
        }
    }

    void addInterest()
    {
        double interest = ( balance * (interestRate/100));
        balance = balance + interest;
        print(balance);
    }
    void display(){
        print("My Account type is $accountType");
        print("Account Number is $accountNumber");
        print("Balance is $balance");
        print("My interest Rate is is $interestRate");
    }
}
```

}



Question:2

Create a class called "Student" with the following attributes:

name (string)

id (string)

courses (list of strings)

And the following methods:

add_course(course): adds a course to the student's list of courses.

drop_course(course): removes a course from the student's list of courses.

display_courses(): prints out the student's list of courses.

Then, create two instances of the Student class, each with their name, id, and courses.

Finally, call the add_course(), drop_course(), and display_courses() methods on each instance and confirm that the information is updated and displayed correctly.

```
void main(List<String> args)
{
  print("First Student Information");

  Student stud1 = Student("Asad", "201A-F22-005", ["computer, Urdu"]);
  stud1.add_course("Physics");
  stud1.drop_course("Chemistry");
  stud1.display_courses();

  print("Second Student Information");
  Student stud2 = Student("Bawany", "BSE-22S-082", ["English, Urdu"]);
  stud2.add_course("ICT");
  stud2.drop_course("DSA");
  stud2.display_courses();
}

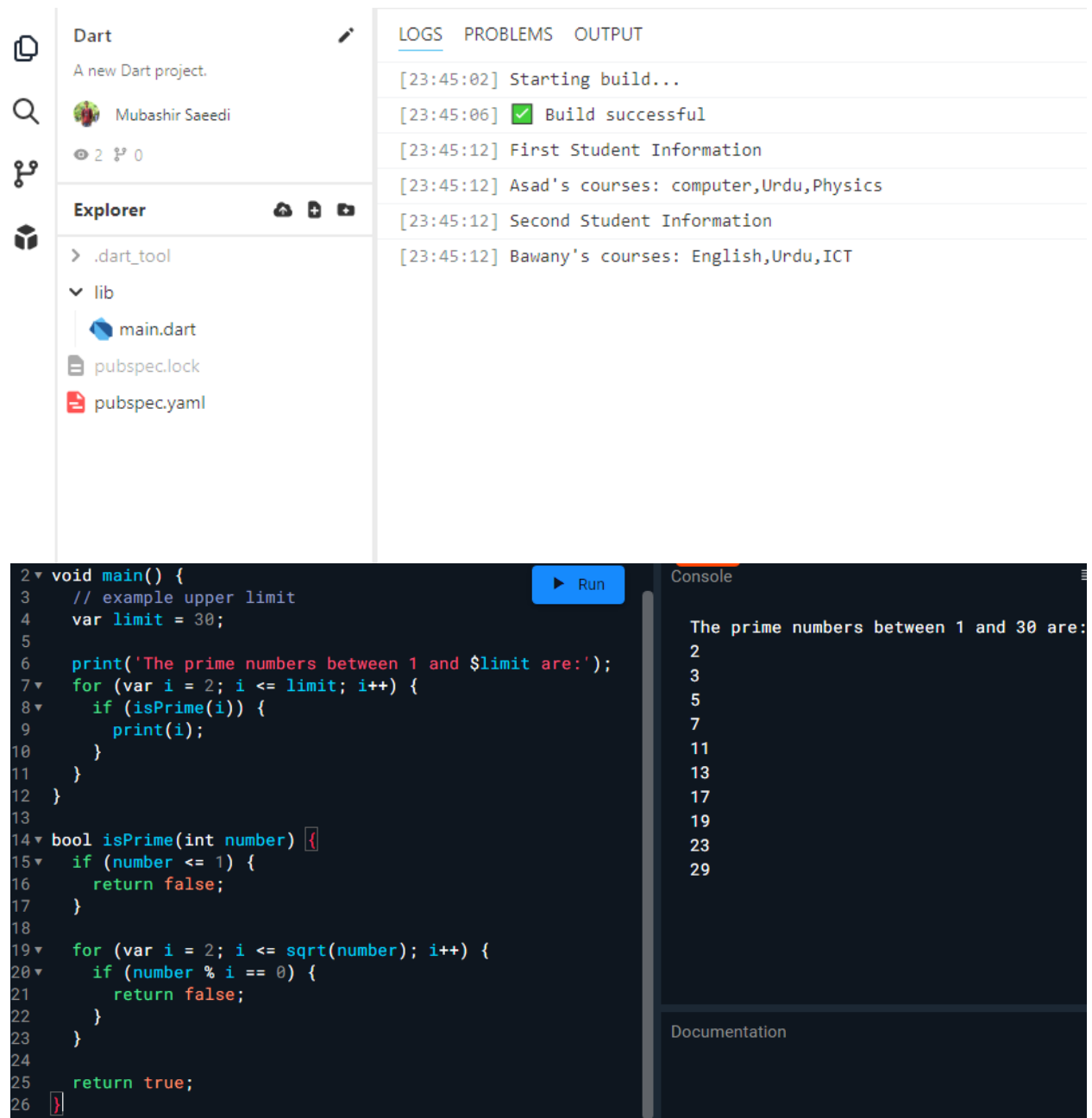
class Student
{
  String name;
  String id;
  List <String> courses;

  Student(this.name, this.id, this.courses);

  void add_course(String course)
  {
    courses.add(course);
  }

  void drop_course(String course)
  {
    courses.remove(course);
  }

  void display_courses()
  {
    print('${name}\'s courses: ${courses.join(", ")}');
  }
}
```



The image shows a screenshot of an IDE interface for a Dart project. The top panel displays the Explorer view on the left, showing a new Dart project named "Dart" by Mubashir Saeedi. The Explorer view lists files: .dart_tool, lib, main.dart, pubspec.lock, and pubspec.yaml. The right panel shows the LOGS, PROBLEMS, and OUTPUT tabs. The LOGS tab is active, displaying the following log entries:

- [23:45:02] Starting build...
- [23:45:06] ✓ Build successful
- [23:45:12] First Student Information
- [23:45:12] Asad's courses: computer,Urdu,Physics
- [23:45:12] Second Student Information
- [23:45:12] Bawany's courses: English,Urdu,ICT

The bottom panel shows the code editor with the following Dart code:

```
2 void main() {  
3   // example upper limit  
4   var limit = 30;  
5  
6   print('The prime numbers between 1 and $limit are:');  
7   for (var i = 2; i <= limit; i++) {  
8     if (isPrime(i)) {  
9       print(i);  
10    }  
11  }  
12 }  
13  
14 bool isPrime(int number) {  
15   if (number <= 1) {  
16     return false;  
17   }  
18  
19   for (var i = 2; i <= sqrt(number); i++) {  
20     if (number % i == 0) {  
21       return false;  
22     }  
23   }  
24  
25   return true;  
26 }
```

The right panel shows the Console output, which displays the prime numbers between 1 and 30:

```
The prime numbers between 1 and 30 are:  
2  
3  
5  
7  
11  
13  
17  
19  
23  
29
```

The bottom panel also shows a Documentation tab, which is currently empty.

Class Quiz # 01

Q#1:

(a) Write a program that generates a random password based on the user's specifications. The user should be able to specify the length of the password and whether it should include numbers, letters, and special characters.

```
import 'dart:math';

void main(){
  int length = 11;
  bool should_Allow_Letr = true;
  bool should_Allow_Nmbr = true;
  bool should_Allow_Spec = true;
  print(gen_Pas(length,should_Allow_Letr,should_Allow_Nmbr,should_Allow_Spec));
}

String gen_Pas(int length,bool should_Allow_Letr,bool should_Allow_Nmbr,bool should_Allow_Spec){
  String alpha = "abcdefghijklmnopqrstuvwxyz";
  String numb = "0123456789";
  String special = ",./:;][<>?!@#%^&*()_+=-|~'";
  String user_specif = "";

  if(should_Allow_Letr == true){
    user_specif = user_specif + alpha;
  }
  if(should_Allow_Nmbr == true){
    user_specif = user_specif + numb;
  }
  if(should_Allow_Spec == true){
    user_specif = user_specif + special;
  }
}
```

```

}

String password = "";

for(int i = 0; i<length; i++){

    int index = Random().nextInt(user_specif.length);

    print(user_specif[index]);

}

return password;

}

```

The screenshot shows an IDE with Dart code on the left and a console on the right. The code defines a `main` function that sets a password length of 11 and three boolean flags: `should_Allow_Letr`, `should_Allow_Nmbr`, and `should_Allow_Spec`, all set to `true`. It then calls `gen_Pas` with these parameters. The `gen_Pas` function is partially visible, showing it takes `length`, `should_Allow_Letr`, and `should_Allow_Nmbr` as arguments. It defines three strings: `alpha` (lowercase letters), `numb` (digits), and `special` (special characters). It then starts building the password by adding characters from `alpha` if `should_Allow_Letr` is true.

```

1  import 'dart:math';
2  void main(){
3      int length = 11;
4      bool should_Allow_Letr = true;
5      bool should_Allow_Nmbr = true;
6      bool should_Allow_Spec = true;
7      print(gen_Pas(length, should_Allow_Letr, should_Allow_Nmbr, should_Allow_Spec));
8  }
9
10 String gen_Pas(int length, bool should_Allow_Letr, bool should_Allow_Nmbr, bool should_Allow_Spec){
11     String alpha = "abcdefghijklmnopqrstuvwxyz";
12     String numb = "0123456789";
13     String special = ",./';][<>?!@#%^&*()_+~|\\~' ";
14     String user_specif = "";
15
16     if(should_Allow_Letr == true){
17         user_specif = user_specif + alpha;
18     }
19     if(should_Allow_Nmbr == true){

```

The console on the right shows the output of the program, which is a 11-character password: `y - t , x 0 c % e g`.

(b)

Write a function that takes a string input from the user and checks if the password is strong enough based on certain criteria (e.g. minimum length, use of uppercase letters, use of special characters, etc.).

Q#2:

Write a program that takes an integer input from the user and prints out numbers from 1 to that integer, but for multiples of 3 print "Fizz" instead of the number, and for multiples of 5 print "Buzz". For numbers that are multiples of both 3 and 5, print "FizzBuzz".

```

void main(){

    int a = 15;

    for(int i = 0; i<=55; i++){

```

```
if(a%3 == 0 && a%5 == 0)
{
  print("Fizz Buzz");
}

else if(a%3 == 0)
{
  print("Fizz");
}

else if(a%5 == 0)
{
  print("Buzz");
}

}
```

```
main.dart x |
lib > main.dart

1
  Run Application
2  ✓ void main(){
3
4    int a = 15;
5
6    for(int i = 0; i<=55; i++){
7
8      if(a%3 == 0 && a%5 == 0)
9      {
10     print("Fizz Buzz");
11   }
12
13   else if(a%3 == 0)
14   {
15     print("Fizz");
```

```
LOGS  PROBLEMS  OUTPUT
-      -
[22:49:43] Fizz Buzz
[22:49:43] Fizz Buzz
[22:49:43] Fizz Buzz
[22:49:43] Fizz Buzz
[22:49:43] Fizz Buzz
[22:49:43] Fizz Buzz
```

Q#3:

Write a function that takes a list of numbers as input and sorts the list in ascending order.

```
void main(){
```

```
List<int> Sequence = [4,6,1,2,8];
print("Before Sorting $Sequence");
```

```
Sequence.sort();
print("After Sorting $Sequence");
```



```
}
```


main.dart ×

lib > main.dart

```
2 void main(){
3
4 List<int> Sequence = [4,6,1,2,8];
5 print("Before Sorting $Sequence");
6
7 Sequence.sort();
8 print("After Sorting $Sequence");
9 }
```

LOGS PROBLEMS OUTPUT

[22:56:52] Starting build...

[22:56:56]  Build successful

[22:57:06] Before Sorting [4, 6, 1, 2, 8]

[22:57:06] After Sorting [1, 2, 4, 6, 8]

TASK INHERITANCE

Calculate area of different shapes

```
void main()
{
  Square sq = Square(4);
  sq.CalculateArea();

  Circle cr = Circle(7);
  cr.CalculateArea();

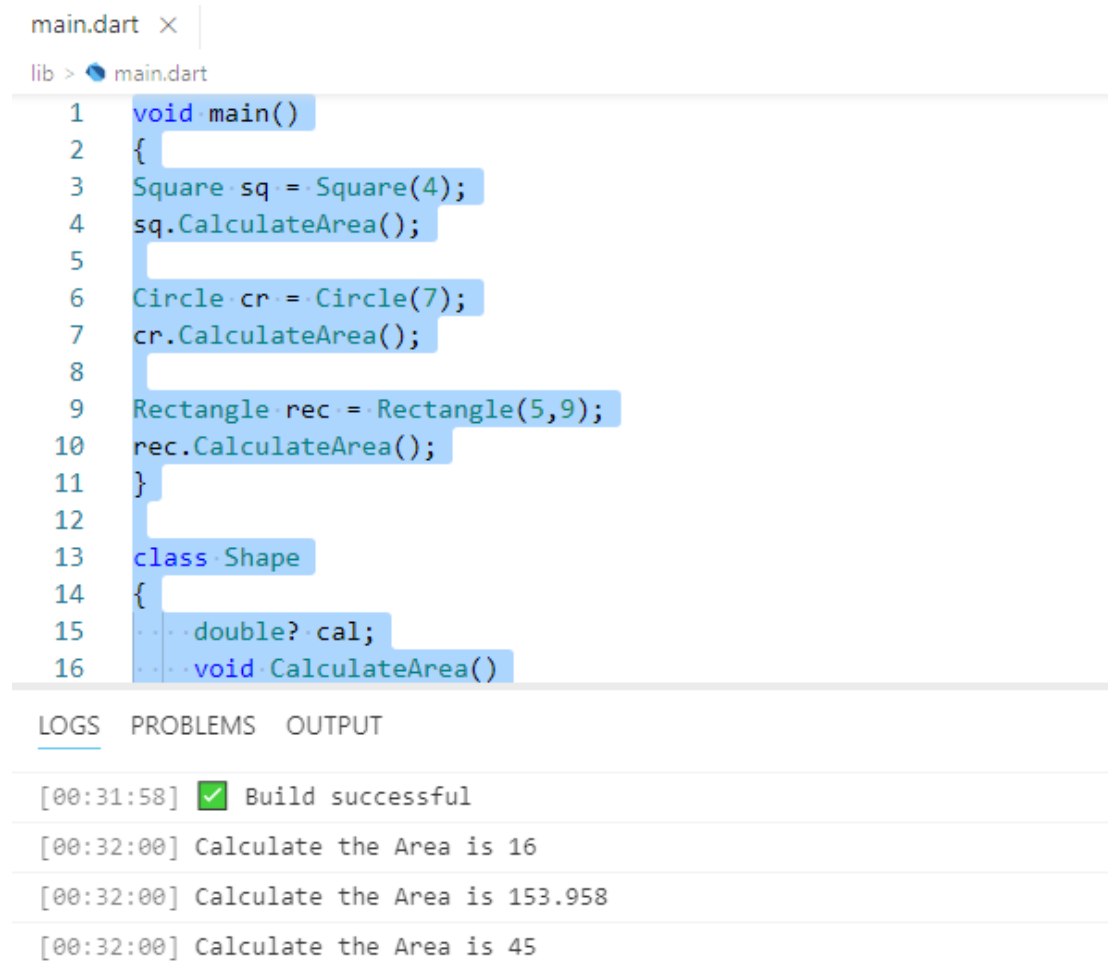
  Rectangle rec = Rectangle(5,9);
  rec.CalculateArea();
}

class Shape
{
  double? cal;
  void CalculateArea()
  {
    print("Calculate the Area is $cal");
  }
}

class Square extends Shape
{
  double sqr;
```

```
    Square(this.sqr);  
    @override  
    void CalculateArea()  
    {  
        cal = sqr*sqr;  
        super.CalculateArea();  
    }  
}  
  
class Circle extends Shape  
{  
    int cr;  
    Circle(this.cr);  
    @override  
    void CalculateArea()  
    {  
        cal = 3.142*cr*cr;  
        super.CalculateArea();  
    }  
}  
  
class Rectangle extends Shape  
{  
    double len;  
    double bre;  
    Rectangle(this.len,this.bre);  
    @override  
    void CalculateArea()
```

```
{  
  cal = (len*bre);  
  super.CalculateArea();  
}  
}
```



The screenshot shows an IDE window titled 'main.dart'. The code defines a `Shape` class with a `CalculateArea` method and a `main` function that creates instances of `Square`, `Circle`, and `Rectangle`, each calling `CalculateArea`. The output pane shows a successful build and three lines of output: 'Calculate the Area is 16', 'Calculate the Area is 153.958', and 'Calculate the Area is 45'.

```
main.dart ×  
lib > main.dart  
1 void main()  
2 {  
3   Square sq = Square(4);  
4   sq.CalculateArea();  
5  
6   Circle cr = Circle(7);  
7   cr.CalculateArea();  
8  
9   Rectangle rec = Rectangle(5,9);  
10  rec.CalculateArea();  
11 }  
12  
13 class Shape  
14 {  
15   double? cal;  
16   void CalculateArea()  
17 {  
18   }  
19 }  
20  
21 }  
22 }  
23 }  
24 }  
25 }  
26 }  
27 }  
28 }  
29 }  
30 }  
31 }  
32 }  
33 }  
34 }  
35 }  
36 }  
37 }  
38 }  
39 }  
40 }  
41 }  
42 }  
43 }  
44 }  
45 }  
46 }  
47 }  
48 }  
49 }  
50 }  
51 }  
52 }  
53 }  
54 }  
55 }  
56 }  
57 }  
58 }  
59 }  
60 }  
61 }  
62 }  
63 }  
64 }  
65 }  
66 }  
67 }  
68 }  
69 }  
70 }  
71 }  
72 }  
73 }  
74 }  
75 }  
76 }  
77 }  
78 }  
79 }  
80 }  
81 }  
82 }  
83 }  
84 }  
85 }  
86 }  
87 }  
88 }  
89 }  
90 }  
91 }  
92 }  
93 }  
94 }  
95 }  
96 }  
97 }  
98 }  
99 }  
100 }
```

LOGS PROBLEMS OUTPUT

[00:31:58] ✓ Build successful

[00:32:00] Calculate the Area is 16

[00:32:00] Calculate the Area is 153.958

[00:32:00] Calculate the Area is 45


```
    print(roll);
}

}
```

