Name: Abdullah Soomro

ID: KWOWFL3381

ASSIGNMENT #3

Q1. What are the various types of operators in dart? Explain with Examples.

<u>OPERATORS IN DART:</u>

The operators are some special symbols that we used to perform different operations on the operands. In Dart, we have several built-in operators which we can used to perform different functions.

<u>DIFFERENT TYPES OF OPERATORS IN DART:</u>

- Arithmetic Operators
- Relational Operators
- Assignment Operators
- Logical Operators
- Conditional Operator

<u>Arithmetic Operators:</u>

In this we have those operators which are used to perform simple arithmetic operation on the operands. They act on two operands.

Operator Symbol	Operator Name	Operator Description
+	Addition	Use to add two operands
–	Subtraction	Use to subtract two operands
*	Multiply	Use to multiply two operands
/	Division	Use to divide two operands
~/	Division	Use two divide two operands but give output in integer
%	Modulus	Use to give remainder of two operands
++	Increment	Use to increment of one in an operand
--	Decrement	Use to decrement of one in an operand

```
Example: Using Arithmetic Operators in the program
void main()
{
    int a = 2;
    int b = 3;
    // Addition
    var c = a + b;
    print("Sum : $c");
    // Subtraction
    var d = a - b;
    print("Subtraction :$d");
    // Multiplication
    var e = a * b;
    print("Multiplication:$e");
    // Division
    var f = b / a;
    print("The quotient:$f");
    // Using ~/ to divide
    var g = b ~ / a;
    print("The quotient:$g");
    // Remainder of a and b
    var h = b % a;
    print("The remainder :$h");
    // Increment
    a++;
    print("Increment of a: $a");
    // decrement
    a--;
    print("Decrement of a: $a");
}

OUTPUT
Sum: 5
Subtraction: -1
Multiplication: 6
The quotient: 1.5
The quotient: 1
The remainder: 1
Increment of a: 4
decrement of a: 2
```

Relational Operators:

In this we have those operators which are used to perform relational operation on the operands.

Operator Symbol	Operator Name	Operator Description
>	Greater than	Check which operand is bigger and give result as boolean expression.
<	Less than	Check which operand is smaller and give result as boolean expression.

>=	Greater than or equal to	Check which operand is greater or equal to each other and give result as boolean expression.
<=	less than equal to	Check which operand is less than or equal to each other and give result as boolean expression.
==	Equal to	Check whether the operand are equal to each other or not and give result as boolean expression.
!=	Not Equal to	Check whether the operand are not equal to each other or not and give result as boolean expression.

```
Example: Using Relational Operators in the program
void main()
{
    int a = 2;
    int b = 3;
     // Greater between a and b
    var c = a > b;
    print("a is greater than b is $c");
    // Smaller between a and b
    var d = a < b;
    print("a is smaller than b is $d");
   // Greater than or equal to between a and b
    var e = a >= b;
    print("a is greater than b is $e");
     // Less than or equal to between a and b
    var f = a <= b;
    print("a is smaller than b is $f");
    // Equality between a and b
    var g = b == a;
    print("a and b are equal is $g");
    // Unequality between a and b
    var h = b != a;
    print("a and b are not equal is $h");
}
Output:
a is greater than b is false
a is smaller than b is true
a is greater than b is false
a is smaller than b is true
a and b are equal is false
a and b are not equal is true
```

Logical Operators:

In this we have those operators which are used to logically combine two or more conditions of the operands.

Operator Symbol	Operator Name	Operator Description
&&	And Operator	Use to add two conditions and if both are true than it will return true.
||	Or Operator	Use to add two conditions and if even one of them is true than it will return true.
!	Not Operator	It is use to reverse the result.

```
Example: Using Logical Operators in the program
void main()
{
    int a = 5;
    int b = 7;
    // Using And Operator
    bool c = a > 10 && b < 10;
    print(c);
    // Using Or Operator
    bool d = a > 10 || b < 10;
    print(d);
    // Using Not Operator
    bool e = !(a > 10);
    print(e);
}
Output:
false
true
true
```

Assignment Operators:

In this we have those operators which are used to assign value to the operands.

Operator Symbol	Operator Name	Operator Description
=	Equal to	Use to assign values to the expression or variable

```
Example: Using Assignment Operators in the program
void main()
{
    int a = 5;
    int b = 7;
     // Assigning value to variable c
    var c = a * b;
    print(c);
 }
Output:
35
```

Conditional Operators:

Decision-making statements are those statements which allow the programmers to decide which statement should run in different conditions. There are four ways to achieve this:

if Statement:

This type of statements simply checks the condition and if it is true the statements within it is executed but if it in is not then the statements are simply ignored in the code.

Syntax:

```
if ( condition ){
  // body of if
}
void main()
{
    int gfg = 10;
    // Condition is true
    if (gfg > 3) {
      // This will be printed
        print("Condition is true");
    }
}
Output:
Condition is true
```

if…else Statement:

This type of statement simply checks the condition and if it is true, the statements within is executed but if not then else statements are executed.

Syntax:

```
if ( condition ){
  // body of if
}
else {
  // body of else
}
void main()
{
    int gfg = 10;

    // Condition is false
    if (gfg > 30) {
      // This will not be printed
        print("Condition is true");
    }
    else {
      // This will be printed
        print("Condition id false");
    }
}
Output:
Condition is false
```

Q2. Cost of one movie ticket is 600 PKR. Write a script to store ticket price in a variable & calculate the cost of buying 5 tickets to a movie.

```
void main()
{
  var ticketPrice = 600;
  print('The price of one ticket is $ticketPrice');
  print('The price of 5 tickets would be ${ticketPrice*5}');
}
```

Q3. How to get difference of lists in Dart? Problem: Consider you have two lists [1,2,3,4,5,6,7] and [3,5,6,7,9,10]. How would you get the difference as output? E.g. [1, 2, 4].

```
void main() {
  List<int> first = [1, 2, 3, 4, 5, 6, 7];
  List<int> second = [3, 5, 6, 7, 9, 10];
  List<int> difference = first.toSet().difference(second.toSet()).toList();
  print(difference);
}
```

Q4. What is a difference between these operators "?? And?"

The ?? operator followed by an = sign in dart makes a syntax where it is used to assign values to variable only when they are null. Once assigned the value cannot be changed.

i.e. a ??= 3;

Where as ? is concatenated to the datatype to tell the compiler that this variable can be null as well as can hold a value.

i.e. int? a = null;

Q5. What are the data types supported in Dart? Explain with Examples.

Data Types

Each variable has an associated data type. In Dart language, there is the type of values that can be represented and manipulated in a programming language. The data type classification is as given below:

Data Type	Keyword	Description
Number	int, double, num	Numbers in Dart are used to represent numeric literals
Strings	String	Strings represent a sequence of characters
Booleans	bool	It represents Boolean values true and false
Lists	List	It is an ordered group of objects
Maps	Map	It represents a set of values as key-value pairs

Number: The number in Dart Programming is the data type that is used to hold the numeric value. Dart numbers can be classified as:

The int data type is used to represent whole numbers.

The double data type is used to represent 64-bit floating-point numbers.

The num type is an inherited data type of the int and double types.

```
void main() {
   // declare an integer
   int num1 = 2;
   // declare a double value
   double num2 = 1.5;
   var c1 = a1+b1;
}
```

String: It used to represent a sequence of characters. It is a sequence of UTF-16 code units. The keyword string is used to represent string literals. String values are embedded in either single or double-quotes.

```
   String string = 'Ab''du''llah';
   String str = 'Ahmed';
```

Boolean: It represents Boolean values true and false. The keyword bool is used to represent a Boolean literal in DART.

```
void main() {
  String str = 'Coding is ';
  String str1 = 'Fun';

  bool val = (str==str1);
  print (val);
}
Output:
false
```

List: List data type is similar to arrays in other programming languages. A list is used to represent a collection of objects. It is an ordered group of objects.

```
void main()
```

```
{
    List gfg = new List(3);
    g[0] = 'Abdullah';
    g[1] = 'Ahmed';
    g[2] = 'Soomro';
    print(g);
    print(g[0]);
}
Output:
[Abdullah, Ahmed, Soomro]
Abdullah
```

Map: The Map object is a key and value pair. Keys and values on a map may be of any type. It is a dynamic collection.

```
void main() {
  Map gfg = new Map();
  gfg['First'] = 'A';
  gfg['Second'] = 'B';
  gfg['Third'] = 'C';
  print(gfg);
}
Output:
{First: A, Second: B, Third: C}
```

Q6. Solve:

a. First declare an array and assign the numbers of the table of 7.

b. Second declare another array and assign the numbers 1-10

c. Now write down the table of 7 using map.fromiterables method.

```
void main() {
  var arr1 = [7, 14, 21, 28, 35, 42, 49, 56, 63, 70];
  var arr2 = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

  var data = Map<int, int>.fromIterables(arr2, arr1);
  print(data);
}
```

Q7. Write a program that

a. Store correct password in a JS variable.

b. Asks user to enter his/her password

c. Validate the two passwords:

d. Check if user has entered password. If not, then give message "Please enter your password"

e. Check if both passwords are same. If they are same, show message "Correct! The password you

f. entered matches the original password". Show "Incorrect password" otherwise.

```
import 'dart:io';

void main() {
  var name = "abcd", js = "0123";

  print("Enter Your Email: ");
  var a = stdin.readLineSync();

  if (a == name) {
    print("Your Email is Correct!");
    print("Enter Your Password: ");
    var b = stdin.readLineSync();
    if (b == password) {
      print("Your Password is Correct!");
      print("Welcome!");
    } else {
      print("Wrong Password!");
    }
  }
  else {
    print("Wrong Email!");
  }
}
```

Q8. Write a program to store 3 student names in an array. Take another array to store score of these three students. Assume that total marks are 500 for each student, display the scores & percentages of students.

```
void main() {
  var student = ["Abdullah", "Ahmed", "Sameer"];
  var score = [400, 300, 200];
  var a = [500, 500, 500], abc = [];
  for (int i = 0; i <= 2; i++) {
    abc[i] = (score[i] / a[i]) * 100;
  }
  var data = Map<String, num>.fromIterables(student, abc);
  print(data);
}
```

Q9. Declare 5 legal & 5 illegal variable names.

```
Code:
void main() {
  //Illegal
 var 4asd, main, a-sda, ae sd, Asd;
  //Legal
 var abc, aBcd, _asd, kream, asd_123;
}
```

Q10. Write a program to replace the "Hyder" to "Islam" in the word "Hyderabad" and display the result.

```
void main() {
  String str = "Hyderabad";
  String result = str.replaceAll('Hyder', 'Islam');
  print(result);
}
```

Q12. Write a program that shows the message "First fifteen days of the month" if the date is less than 16th of the month else shows "Last days of the month".

```
import 'dart:io';
void main() {
  print("Enter a Date");
  var a = int.parse(stdin.readLineSync()!);
  if (a <= 15) {
    print("First fifteen days of the month");
  } else {
    print("Last days of the month");
  }
}
```

Q13. Find 5 new methods of List and String.

LIST:

sublist(): Returns a new list containing the elements between start and end

shuffle(): Randomizes the order of the elements in the array

asMap(): Returns an unmodifiable Map view of this**.**

whereType():

getRange(): Returns an Iterable that iterates over the objects in the range start inclusive to end exclusive.

String

trim(): returns a new string without any whitespace characters at the beginning or end. There are two variants of this methods trimLeft() and trimRight().

padLeft(): returns a right-justified string with given minimum width.

padRight(): returns a left-justified string with given minimum width.

replaceRange(): returns new string which is substring from start to end index with string to be replaced.

isEmpty(): returns true if string is empty.

isNonEmpty(): returns true if string is not empty.