Name: Abdul Karim
Roll no: KWOWFL3382

Assignment 3

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

There are 8 types of operators in dart language. They are:

a. Arithmetic Operators
b. Relational Operators
c. Type Test Operators
d. Bitwise Operators
e. Assignment Operators
f. Logical Operators
g. Conditional Operator
h. Cascade Notation Operator

Arithemetic Operators:

These types of operators perform mathematical operations. They are binary and require two operands.

Operator Symbol	Operator Name	Operator Description
+	Addition	Adds two operands
–	Subtraction	Subtracts two operands
*	Multiply	Multiplies two operands
/	Division	Divides two operands
~/	Division	Divides two operands and give output in integer
%	Modulus	Gives remainder of two operands

Example:

```
void main()

{

int a = 13;

int b = 1;

var c = a + b;

print("Sum of a and b is $c");

        var d = a - b;

print("The difference between a and b is $d");

        var f = a * b;
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
print("The product of a and b is $f");

        var g = b / a;

print("The quotient of a and b is $g");

        var h = b ~ / a;

print("The quotient of a and b is $h");

var i = b % a;

print("The remainder of a and b is $i");

}
```

Relational Operators:

This type of operator related an operand to the other in a certain way.

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:

```
void main()

{
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
int a = 5;

int b = 22;

            var c = a > b;

print("a is greater than b is $c");

            var d = a < b;

print("a is smaller than b is $d");

            var e = a >= b;

print("a is greater than b is $e");

            var f = a <= b;

print("a is smaller than b is $f");

            var g = b == a;

print("a and b are equal is $g");

            var h = b != a;

print("a and b are not equal is $h");

}
```

Type Test Operators:

This types of operators are used to perform compare different operands.

Operator Symbol	Operator Name	Operator Description
is	is	Gives boolean value true as output if the object has specific type
is!	is not	Gives boolean value false as output if the object has specific type

Example:

```
void main()
{
    String a = 'abc';
    double b = 3.3;

    print(a is String);
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
     print(b is !int);
 }
```

Bitwise Operators:

This type of operators are used to perform bitwise operation on the operands.

Operator Symbol	Operator Name	Operator Description
&	Bitwise AND	Performs bitwise and operation on two operands.
|	Bitwise OR	Performs bitwise or operation on two operands.
^	Bitwise XOR	Performs bitwise XOR operation on two operands.
~	Bitwise NOT	Performs bitwise NOT operation on two operands.
<<	Left Shift	Shifts a in binary representation to b bits to left and inserting 0 from right.
>>	Right Shift	Shifts a in binary representation to b bits to left and inserting 0 from left.

Example:

```
 void main()

{

int a = 5;

int b = 7;

var c = a & b;

print(c);

var d = a | b;

print(d);

var e = a ^ b;

print(e);

var f = ~a;

print(f);
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
        var g = a << b;
print(g);
        var h = a >> b;
print(h);
}
```

Assignment Operators:
This type of operators 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
??=	Assignment operator	Assign the value only if it is null.

Example:

```
void main()
{
int a = 5;
int b = 7;
var c = a * b;
print(c);
var d;
d ? ? = a + b; // Value can be assigned as it is null
print(d);
d ? ? = a - b; // Value cannot assigned as it is not null
print(d);
}
```

Logical Operators:
This type of operators are used to logically combine two or more conditions.

Operator Symbol	Operator Name	Operator Description
&&	And Operator	Use to add two conditions and if both are true than it will return true.

Name: Abdul Karim
Roll no: KWOWFL3382

||	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:

```
void main()
{
int a = 5;
int b = 7;
        bool c = a > 10 && b < 10;
print(c);
bool d = a > 10 || b < 10;
print(d);
bool e = !(a > 10);
print(e);
}
```

Conditional Operators:
This type of operators are used to perform comparison on the operands.

Operator Symbol	Operator Name	Operator Description
condition ? expersion1 : expersion2	Conditional Operator	It is a simple version of if-else statement. If the condition is true than expersion1 is executed else expersion2 is executed.
expersion1 ?? expersion2	Conditional Operator	If expersion1 is non-null returns its value else returns expression2 value.

Example:

```
void main()
{
int a = 5;
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
int b = 7;


var c = (a < 10) ? "abc" : "def";

print(c);

int n;

var d = n ? ? "n has Null value";

print(d);

n = 10;

d = n ? ? "n has Null value";

print(d);

}
```

Cascade Notation Operators:
This type of operators perform a sequence of operation on the same element. It allows you to perform multiple methods on the same object.

Operator Symbol	Operator Name	Operator Description
..	cascading Method	It is used to perform multiple methods on the same object.

Example:

```
class Abc {

var a;

var b;


void set(x, y)

{

this.a = x;

this.b = y;

}


void add()
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
{

var z = this.a + this.b;

print(z);

}

}


void main()

{

Abc abc1 = new ABC();

Abc geek2 = new ABC;

abc1.set(1, 2);

abc1.add();

abc2..set(3, 4)

..add();

}
```

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?

Name: Abdul Karim
Roll no: KWOWFL3382

Q3. 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.

There are 5 data types in dart. They are

a. Number
b. Strings
c. Booleans
d. Lists
e. maps

Numbers:

A numeric variable in dart holds a numerical value. The three types of numbers dart support are

int - that holds & represents a whole numbers

double - that holds & represents a decimal numbers of 64-bits

num - a generic numeric datatype that can store numbers of any of the types mentioned above

Name: Abdul Karim
Roll no: KWOWFL3382

```
int a = 64;

double b = 5.5;

num c = 78;

num d = 9.66;
```

Strings:

This type of variables can store a sequence of characters. They are embedded between a single or double inverted commas. It is a UTF-16 code units sequence. The keyword String is used in dart for this.

```
String 'AbdulKarim';

String "Flutter & Dart;
```

Boolean:

This type of variables can either have a true or a false but nothing else. A keyword bool used to declare this type of variables.

```
void main() {

int a = 2;

int b = 4;

bool x = (a==b);

print (bool);

}
```

List:

List is identical to what we call an array in different languages. A lists holds and represents an ordered collection of objects.

```
  List abc = new List(3);

  abc[0] = 'Flutter';
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
   abc[1] = 'and';

   abc[2] = 'Dart';
```

Map:

A map contains a pair of a key and a value. Keys and maps are free to be of any data type.

```
Map abc = new Map();

  abc['First'] = 'Flutter';

  abc['Second'] = 'and';

  abc['Third'] = 'Dart';
```

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, S, 6, 7, 8, 9, 10];

var data = Map<int, int>.fromIterables(arr2, arri);

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

```
import 'dart:io';
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
void main() {
  var email= "abd@flutter.com", js = "100100";
  print("Enter Your Email: ");
  email = stdin.readLineSync();
  if (a == name) {
    print("Your Email is Correct!");
    print("Enter Your Password: ");
    b = stdin.readLineSync();
    if (js == password) {
      print("Your Password is Correct!");
      print("Welcome!");
    } else {
      print("Incorrect Password!");
    }
  }
  else {
    print("Invalid 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 = ["Abdul", "Karim", "Mustafa"];
  var score = [368, 422, 317];
  var ass = [500, 500, 500], abc = [];
  for (int i = 0; i <= 2; i++) {
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
    abc[i] = (score[i] / ass[i]) * 100;
  }
  var data = Map<String, num>.fromIterables(student, abc);
  print(data);
}
```

Q9. Declare 5 legal & 5 illegal variable names.

```
void main ()
{
var 1abc, Abc, a-bc, print, a bc;
var abc, aBc, a12, _abc, abc_123;
}
```

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

```
Void main() {
  String city = ”Hyderabad”;
  String change = city.replaceAll(’Hyder’, ’Islam’);
  Print(change);
}
```

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');
  int inputDate = int.parse(stdin.readLineSync()!);
  if (inputDate < 16) {
```

Name: Abdul Karim
Roll no: KWOWFL3382

```
    print('First fifteen days of the month');

  } else if (inputDate < 31) {

    print('Last fifteen days of the month');

  }

}
```

Q13. Find 5 new methods of List and String.

String

1. toLowerCase(): Converts all characters in this string to lower case.
2. toUpperCase(): Converts all characters in this string to upper case.
3. trim(): Returns the string without any leading and trailing whitespace.
4. compareTo(): Compares this object to another.
5. replaceAll(): Replaces all substrings that match the specified pattern with a given value.

List

1. First: Returns the first element in the list.
2. isEmpty: Returns true if the collection has no elements.
3. isNotEmpty: Returns true if the collection has at least one element.
4. Length: Returns the size of the list.
5. Last:Returns the last element in the list.