Scala
Session-2

Scala session - 2

1. what is a pure function

3 characteristics:

a. A pure function is only dependent on its input parameter. No external variable impacts the result.

b. A pure function should not change its input parameter.

c. A pure function should not have any side effects.

2. is there any easy way to test the purity of a function.

Referential transparency:

A function is referentialy transparent if replacing the function with a value do not impact the result.

sqrt(4) sqrt(9) - referentially transparent / **pure**

2 3

dollarsToRS(40) - not referentially transparent / **not pure**

3. First class function

whatever we were able to do with values in traditional programming, same thing we should be able to do with the functions as well.

we should be able to treat functions like values.

3 characteristics:

a. we should be able to assign a function to a variable

```
var a: Int = 5

def doubler(i: Int): Int = {
      i * 2
}

val a = doubler(_)

a(2)
```

b. we should be able to pass a function as parameter to the function

```
def tripler(i: Int): Int = { i * 3 }

def func(i: Int , f:Int => Int ) = {
     f(i)
}

func(5,tripler)
```

c. we should be able to return a function from a function

```
def func = {
x:Int => x * x
}
```

4. Higher order function

A function which either takes a function as input parameter or returns another function as its output.

map

if we have n input rows, then we will get n output rows also.

```
var a = 1 to 10

def doubler(i: Int): Int = {i * 2}

a.map(doubler)
```

5. Anonymous function

A function without a name.

```
var a = 1 to 10

def doubler(i: Int):Int = {
      i * 2
}

a.map(doubler)

a.map(x => x * 2)
```

in python the same thing is called as lambdas

6. Immutability

we cannot change the value.

but immutability is more preferred.

```
var a = 5
a = 6
```

```
val b = 7
b = 8
```

val is preferred over var.

7. Loop vs recursion vs tail recursion

find factorial of a number

loop

```
def factorial(input: Int): Int = {
var result: Int = 1
for(i <- 1 to input)
{
      result = result * i
}
result
}
```

recursion

```
def factorial(input: Int): Int = {
      if (input == 1) 1
      else input * factorial(input-1)
}

factorial(5)
```

```
5 * factorial(4)
5 * 4 * factorial(3)
5 * 4 * 3 * factorial(2)
5 * 4 * 3 * 2 * factorial(1)
5 * 4 * 3 * 2 * 1
```

tail recursion

```
def factorial(input: Int , result: Int): Int = {
      if(input == 1) result
      else factorial(input-1,result*input)
}

factorial(1,24)
```

what tail recursion says is that the recursive call should be the last statement in the function.

13. Statement vs Expression

each line in a code block is a statement.

Expression is a line of code that returns something.

in scala we do not have statements and we have only expressions.

that means each statement returns something.

```
val a = println("hello world")

Unit ()

var x

val a = 5

val x = if (a==5) 2 else 7


print (x)
```

Google REVIEWS

5 Star Google Rated Big Data Course

LEARN FROM THE EXPERT

9108179578

Call for more details

Follow US

Trainer	Mr. Sumit Mittal
LinkedIn	https://www.linkedin.com/in/bigdatabysumit/
Website	https://trendytech.in/courses/big-data-online-training/
Phone	9108179578
Email	trendytech.sumit@gmail.com
Youtube	TrendyTech
Twitter	@BigdataBySumit
Instagram	bigdatabysumit
Facebook	https://www.facebook.com/trendytech.in/