6/5/18

REPL : read - eval - print loop

PAGE :
DATE : / /

lab-8

Scala

```
scala > 8 * 5 + 2
res0 : Int = 42

scala > 0.5 * res0
res1 : Double = 21.0

scala > "Hello," + res0
res2 : java.lang.String = Hello, 42

scala > val answer = 8 * 5 + 2
answer : Int = 42
```

can use these names in subsequent expression

```
scala > 0.5 * answer
res3 : Double = 21.0
```

value declared with val is actually a constant - you can't changes its contents:

```
scala > answer = 0
<console>:6: error: reassignment to val

var counter = 0
counter = 1

val greeting : String = null
val greeting : Any = "Hello"

val xmax, ymax = 100
var greeting, message : String = null
```

```
1.toString()
1.to(10)     // yields Range(1,2,3,4,5,6,7,8,9,10)

"Hello".intersect("world")   // yields "lo"

sqrt(2)
pow(2, 4)
min(3, pi)
scala.math.sqrt(2)

val s = "Hello"
s(4)

 print("Answer:")
 println(42)

o/p println("Answer: " + 42)
```

→ // Read a line of input from the console with the readLine method of the scala.io.StdIn class.

```
import scala.io
val name = StdIn.readLine("your name:")
print("Your age:")
val age = StdIn.readInt()
println(s"Hello, ${name}! Next year, you will be
                ${age + 1}.")

for(i <- 1 to 3; j <- 1 to 3) print(f"${10 * i + j}%3d")
 // prints 11  12  13  21  22  23  31  32  33
```

```
for (i<-1 to 3; j<-1 to 3 if i!=j) print
  (f"${10 * i+j}%3d")
 // Prints 12 13 21 23 31 32
```

```
for (i<-1 to 3; from= 4 -i; j<-from to 3) print
  (f"${10 * i+j}%3d")
  // print 13 22 23 31 32 33
```

```
for (i<- 1 to 10) yield i%3
// yields vector(1, 2, 0, 1, 2, 0, 1, 2, 0, 1)
```

```
for (c <-"Hello"; i<-0 to 1) yield (c+i).toChar
 // yields "HIeflmlmop"
```

```
for (i<-0 to 1; c<-"Hello") yield (c+i).toChar
// yields Vector('H', 'e', 'l', 'l', 'o', 'I', 'f', 'n', 'm', 'p')
```

Fixed-length Arrays

```
val nums= new Array[Int](10)
// An array of ten integers, all initialized
with zero
```

```
val a = new Array[String](10)
// A string array with ten elements, all initialized
with null
```

```
val s = Array("Hello", "World")
// An Array[String] of length 2 - the type is inferred
```

Variable-Length Arrays : Array Buffers

```
import scala.collection.mutable.ArrayBuffer
val b = ArrayBuffer[Int]()
// or new ArrayBuffer[Int] // An empty array buffer, ready to
                                                   hold integers

b+=1          // ArrayBuffer(1)

b+=(1,2,3,5) // Add an element at the end with +=
                  // ArrayBuffer(1,1,2,3,5)
// Add multiple elements at the end by enclosing them in
b++=Array(8,13,21)          // ArrayBuffer(1,1,2,3,5,8,13,21)    parentheses

b.insert(2,6)          // ArrayBuffer(1,1,6,2) [insert before index 2]

b.insert(2,7,8,9)  // ArrayBuffer(1,1,7,8,9,6,2)
b.remove(2,3)          // ArrayBuffer(1,1,2)
```

Constructing a Map

```
val scores = Map("Alice" -> 10, "Bob" -> 3, "Cindy" -> 8)
```

23/5

```
val scores = scala.collection.mutable.Map("Alice" -> 10, "Bob" -> 3,
      "Cindy" -> 8)
val scores = scala.collection.mutable.Map[String, Int]()
```