* **Control structures and functions**
  + **Conditional Expressions:**

Every expression in scala has some value

Val a = if(x>1) 1 else -1;

Any: This is a datatype which can take any datatype

Unit: This is like void it has no value unit is denoted as ()

Void = empty wallet. Unit: Wallet with a ssbill labelled no Dollor

Scala has no switch statements

:paste 🡪 This is used to type in multiple lines of code in REPL

* + **Statement termination**

Semi colon (;) is optional in scala. It is mandatory only when multiple statements are there in same line.

We can use java style of semicolon also if we are uncomfortable of not putting semi colon

* + **Block Expressions and Assignments:**

In java or c++ code block is a list of statements in curly braces

In Scala, a {} block is list of expressions and result is also a expression

The value of the block is valu of the last statement

scala> val addition = {val x = 10l; val y = 10; x+y}

addition: Long = 20

x = y = 1 // No

* + **Input and Output**

print("Answer: ")

println(42)

You can read a line of input from the console with the readLine method of the

scala.io.StdIn class. To read a numeric, Boolean, or character value, use readInt,

readDouble, readByte, readShort, readLong, readFloat, readBoolean, or readChar. The readLine

method, but not the other ones, takes a prompt string

import scala.io.StdIn

val name = StdIn.readLine("Enter you name:" )

println("Enter your age")

val age = StdIn.readInt()

print(s"Hi ${name} you age in next one year will be ${age + 1}")

* + **Loops**

Scala has same while and do while loop.

Scala has no direct analog of the for (initialize; test; update) loop. If you need such

a loop, you have two choices. You can use a while loop. Or, you can use a for

statement like this:

for (i <- 1 to n)

r = r \* i

scala> :pas

// Entering paste mode (ctrl-D to finish)

for (i <- 1 to 10)

println(i)

// Exiting paste mode, now interpreting.

1

2

3

4

5

6

7

8

9

10

In Scala there is no break or continue keyword to break out of loop

* + **Advanced for loop**

# Functions