Part 1
Functions

The goal is to write such function that the example code below compiles and gives expected results (on the right). Reminder, use if external materials is prohibited.

```
println( pi ) // no extra () inside
println( pi(pi) ) //multiplication by pi
println( pi(pi(pi)) ) // and one more

printarg(arg=“hello") // named parameter,
printarg()
```

```
Expected results:
3.1415
9.86902225 // multiplication
31.003533398375
hello
-
32
```

```
println(repN(5,  (x: Int) => 2*x , 1 )) // repeat function application N times i.e. f(f(f(f(f(1))))), 1 the argument of the first invocation
```

pi - 2 points

printarg - 1 point

repN - 3points

Part 2
Tail recursion

- The second exercise is about using tail-recursive function to sum fractions 1/(2^n): 1, 1/2, 1/4, 1/8, 1/16 etc...
  The summation should stop when the value of the fraction to be added in the next cycle is less than the precision parameter passed as an argument to sumfrac.

- example invocation:

```
val sum1 = sumfrac( 1e-3 )

val sum2 = sumfrac( 1e-9 )

val sum3 = sumfrac( 1e-16 )
```

- Both techniques, either using buffer parameters or inner function can be used.