Part 1

6 points, no help other than scala console and instructor

- 3 points) Generate numbers that are not squares of some numbers up to a value N. For N == 11 result should be:
  1 2 3 5 6 7 8 10 11 (4 and 9 should not be printed)

  - The **N** should be passed **from command line**, if nothing is passed then only 50 first natural numbers need to printed

    - in IJ, to set cmd line args go to:
      Run -> Edit Configurations -> Program arguments, and enter some number there

- 3 points) We have 2D array:
  val x = Array.ofDim[Int](4,5);
  for (i <- 0 until 4; j <- 0 until 5) x(i)(j) = i+j

  - print this array in the following form:

```
|0 1 2 3 4|
|1 2 3 4 5|
|2 3 4 5 6|
|3 4 5 6 7|
```

    - merge two arrays x and y of identical sizes (here 4x5) so that **each element** in the resulting array **is larger of the corresponding element in x and y**

Part 2

4 points, all help you need

- 1point) Write a function that takes any number of integers, sums them up, and returns that sum
- 1point) Write function that repetitively applies another function. I.e.:
  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 - expected result is 32
- 2points) Write closure that can be used to accumulate numbers like in the example:
  val ac1 = accumuator()
  val ac2 = accumulator()
  println(ac1(1)) // prints 1
  println(ac1(7)) // prints 8
  println(ac2(3)) // prints 3
  println(ac1(7)) // prints 15