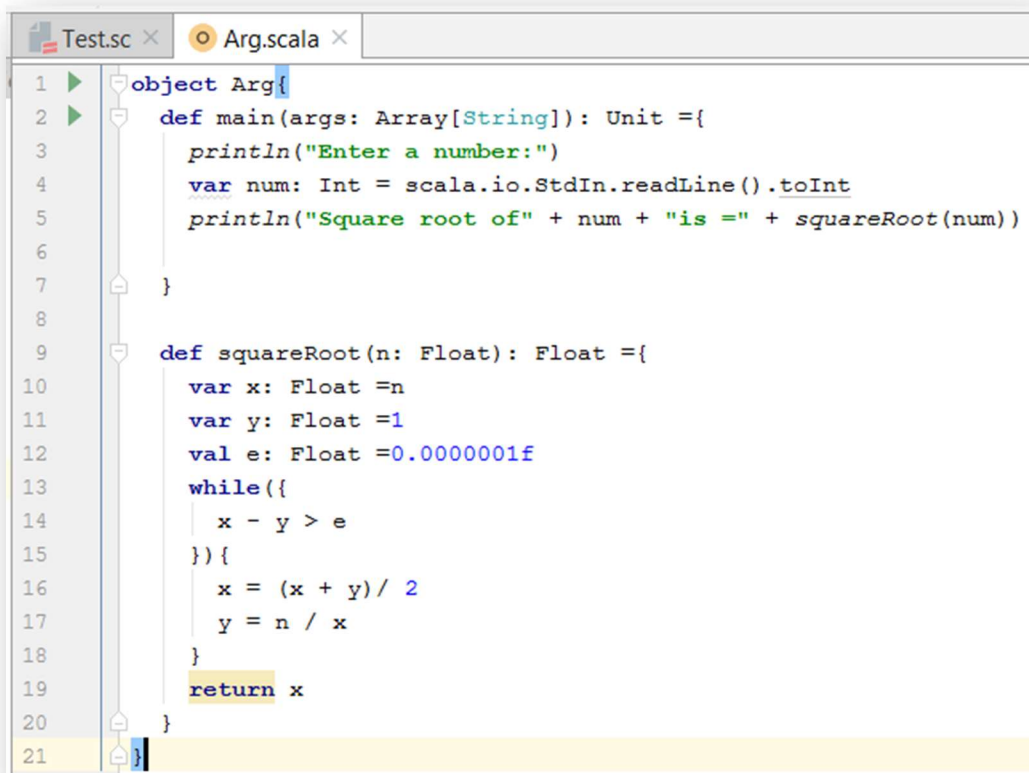


## Assignment 13.3

Find square root of number using Babylonian method.

1. Start with an arbitrary positive start value  $x$  (the closer to the root, the better).
2. Initialize  $y = 1$ .
3. Do following until desired approximation is achieved.
  - a) Get the next approximation for root using average of  $x$  and  $y$
  - b) Set  $y = n/x$

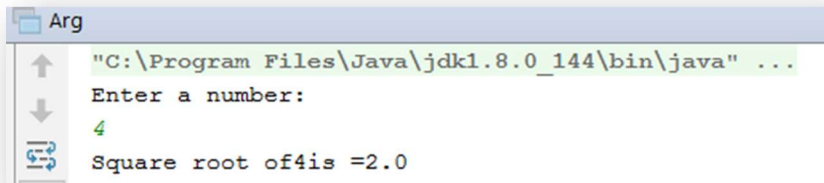
**Code:**



```
1  object Arg{
2  def main(args: Array[String]): Unit = {
3      println("Enter a number:")
4      var num: Int = scala.io.StdIn.readLine().toInt
5      println("Square root of" + num + "is =" + squareRoot(num))
6
7  }
8
9  def squareRoot(n: Float): Float = {
10     var x: Float = n
11     var y: Float = 1
12     val e: Float = 0.0000001f
13     while({
14         x - y > e
15     }) {
16         x = (x + y) / 2
17         y = n / x
18     }
19     return x
20 }
21 }
```

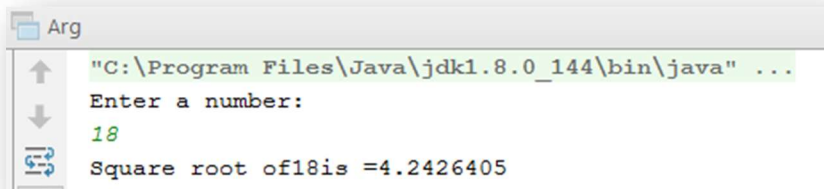
## Output:

Square of a proper number



```
Arg
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Enter a number:
4
Square root of 4 is =2.0
```

Square of an improper number



```
Arg
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Enter a number:
18
Square root of 18 is =4.2426405
```