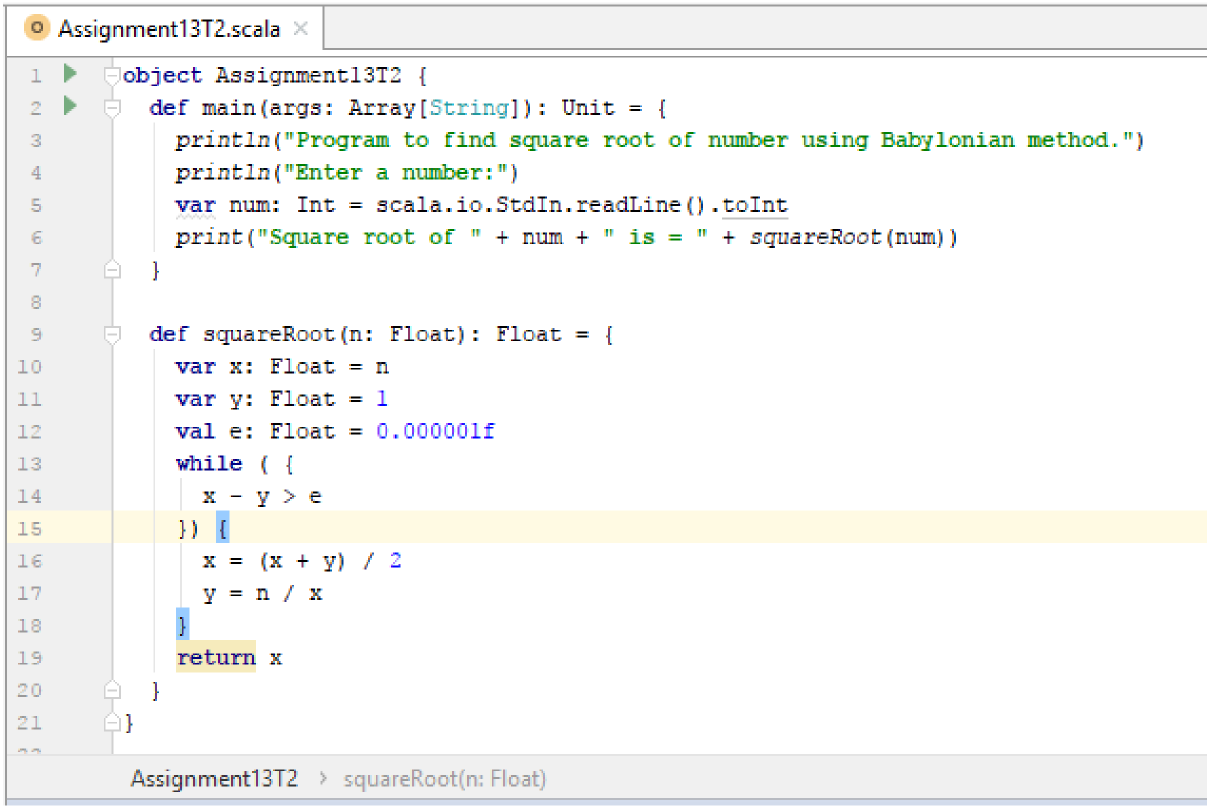
Assignment 13.2:

Problem Statement:

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

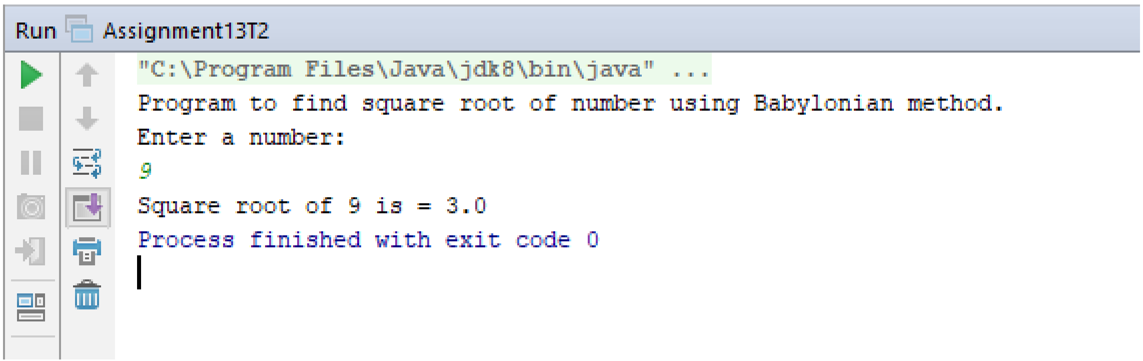
Program:



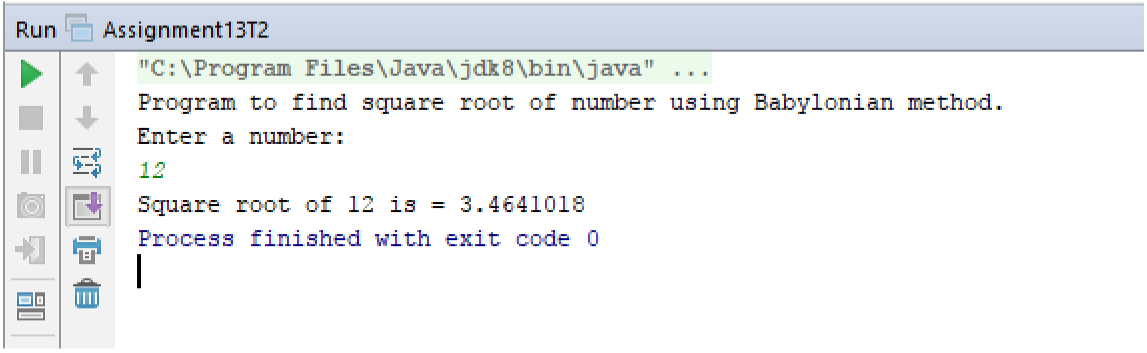
```scala
object Assignment13T2 {
  def main(args: Array[String]): Unit = {
    println("Program to find square root of number using Babylonian method.")
    println("Enter a number:")
    var num: Int = scala.io.StdIn.readLine().toInt
    print("Square root of " + num + " is = " + squareRoot(num))
  }

  def squareRoot(n: Float): Float = {
    var x: Float = n
    var y: Float = 1
    val e: Float = 0.000001f
    while ( {
      x - y > e
    }) {
      x = (x + y) / 2
      y = n / x
    }
    return x
  }
}
```

Output:



```
"C:\Program Files\Java\jdk8\bin\java" ...
Program to find square root of number using Babylonian method.
Enter a number:
9
Square root of 9 is = 3.0
Process finished with exit code 0
```



```
"C:\Program Files\Java\jdk8\bin\java" ...
Program to find square root of number using Babylonian method.
Enter a number:
12
Square root of 12 is = 3.4641018
Process finished with exit code 0
```