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:

Test.sc Arg.scala

```scala
object Arg{
  def main(args: Array[String]): Unit ={
    println("Enter a number:")
    var num: Int = scala.io.StdIn.readLine().toInt
    println("Square root of" + num + "is =" + squareRoot(num))

  }

  def squareRoot(n: Float): Float ={
    var x: Float =n
    var y: Float =1
    val e: Float =0.0000001f
    while({
      x - y > e
    }){
      x = (x + y)/ 2
      y = n / x
    }
    return x
  }
}
```

Output:

Square of a proper number

```
Arg
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Enter a number:
4
Square root of4is =2.0
```

Square of an improper number

```
Arg
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Enter a number:
18
Square root of18is =4.2426405
```