**Problem Statement-** **Find square root of number using Babylonian method.**

**Solution-**

Below is the code used to find the square root of number using Babylonian method –

Here we are trying to find the square root of 2 and chosen the value of iterations as 5.

Now as the method suggests we are guessing 1 as the square root of number 2. So as approximation we are calculating using average as shown below-

```
package main.scala

object Sqrt {

  def squareRoot(n: BigDecimal): Stream[BigDecimal] =
  {
    def squareRoot(guess: BigDecimal, n: BigDecimal): Stream[BigDecimal] = {
      Stream.cons(guess, squareRoot(0.5 * (guess + n / guess), n))
    }
    squareRoot(1, n) // best guess, let's say square root of 2 is 1
  }

  def main(args: Array[String]): Unit = {
    println(squareRoot(2))
    val iterations = 5
    println(squareRoot(2)(iterations - 1))
    println(squareRoot(2).take(iterations).toList)

  }
}
```

Sqrt.scala

```
1   package main.scala
2
3   object Sqrt {
4
5     def squareRoot(n: BigDecimal): Stream[BigDecimal] =
6     {
7       def squareRoot(guess: BigDecimal, n: BigDecimal): Stream[BigDecimal] = {
8         Stream.cons(guess, squareRoot(0.5 * (guess + n / guess), n))
9       }
10      squareRoot(1, n) // best guess, let's say square root of 2 is 1
11    }
12
13    def main(args: Array[String]): Unit = {
14      println(squareRoot(2))
15      val iterations = 5
16      println(squareRoot(2)(iterations - 1))
17      println(squareRoot(2).take(iterations).toList)
18
19    }
20  }
```

Below screenshot shows the solution after running above code-

```
Sqrt
"C:\Program Files\Java\jdk1.8.0_131\bin\java" ...
Stream(1, ?)
1.41421356237468991062629557889013 5
List(1, 1.5, 1.41666666666666666666666666666666, 1.41421568627450980392156862745098, 1.41421356237468991062629557889013 5)

Process finished with exit code 0
```