Michael Xavier Canonizado

BSCS 2A – Object Oriented Programming

```java
public class HelloWorld {
    public static void main(String args[]) {
        float radicand = Float.parseFloat(args[0]);
        int iterations = 10;

        double x = radicand;

        // Newton-Raphson Algorithm
        /* The more iterations the
        more accurate the result will be*/
        /* Source:
        https://surajregmi.medium.com/
        how-to-calculate-the-square-root-
        of-a-number-newton-raphson-method-
        f8007714f64*/
        System.out.print("\n");
        for (int i = 1; i <= iterations; i++) {
            x = (x + (radicand / x)) / 2;
            System.out.println(i + " iteration" + ": " + x);
        }
        System.out.print("\n");
        System.out.println("Result: " + x);
    }
}
```

```
Student@DESKTOP-3SVV1VQ MINGW64 /c/
ANONIZADO/Lab 2/CmdSquareRoot
$ java HelloWorld 64

1 iteration: 32.5
2 iteration: 17.234615384615385
3 iteration: 10.474036101145005
4 iteration: 8.292191785986859
5 iteration: 8.005147977880979
6 iteration: 8.000001655289593
7 iteration: 8.00000000000017
8 iteration: 8.0
9 iteration: 8.0
10 iteration: 8.0

Result: 8.0
```

```
Student@DESKTOP-3SVV1VQ MINGW64 /
ANONIZADO/Lab 2/CmdSquareRoot
$ java HelloWorld 1200

1 iteration: 600.5
2 iteration: 301.2491673605329
3 iteration: 152.6162904300545
4 iteration: 80.23957349381237
5 iteration: 47.59739379134379
6 iteration: 36.404429105092106
7 iteration: 34.68372558703923
8 iteration: 34.64104244751303
9 iteration: 34.641016151387525
10 iteration: 34.64101615137754

Result: 34.64101615137754
```