Homework 1

Benny Chen

January 31, 2023

1 Problem A

A.1

```java
//AreafromCircumference.java
//This program takes in the circumference(INPUT) and calculates the area of a circle(OUTPUT)

public class AreafromCircumference{
    public static void main(String[] args){
        //Set the value of pi to 22/7
        double pi = 22/7;

        //Set the value of circumference to the input
        double circumference = Double.parseDouble(args[0]);

        //Calculate the area of the circle
        double area = (circumference * circumference) / (4 * pi);

        //Print the area of the circle
        System.out.printf("Area of circle is: %.2f\n", area);
    }
}
```

Test Cases

```
bob@Bennys-MacBook-Pro HW1 % java AreafromCircumference.java 20
Area of circle is: 33.33
bob@Bennys-MacBook-Pro HW1 % java AreafromCircumference.java 100
Area of circle is: 833.33
bob@Bennys-MacBook-Pro HW1 % java AreafromCircumference.java 12482357
Area of circle is: 12984103022954.08
```

A.2

Can π be encoded as "static final" in the code? Why or why not? Justify your answer.
Yes, it can be encoded as "static final" in the code mainly due to the fact that we use 22/7 as our approximation of π and it will be a constant value that will not change throughout the program. We would have to write is as

```
19 static final double pi = 22.0 / 7.0;
```

2 Problem B

```
20 // VendingChange.java
21 // This program takes in the amount of an item(INPUT) and
       calculates the change of $1 in the amount of quarters, dimes
       , and nickels(OUTPUT)
22
23 public class VendingChange {
24     public static void main(String[] args){
25         //Set the value of item to the input
26         int item = Integer.parseInt(args[0]);
27
28         //Calculate the change
29         int change = 100 - item;
30
31         //Calculate the amount of quarters
32         int quarters = change / 25;
33
34         //Calculate the amount of dimes
35         int dimes = (change % 25) / 10;
36
37         //Calculate the amount of nickels
38         int nickels = ((change % 25) % 10) / 5;
39
40         //Print the change
41         System.out.printf("You bought a item for %d cents and
               gave me a dollar, so your change is \n%d quarters, \
               n%d dimes, and \n%d nickels\n", item, quarters,
               dimes, nickels);
42     }
43 }
```

Test Cases

```
bob@Bennys-MacBook-Pro HW1 % java VendingChange.java 45
You bought a item for 45 cents and gave me a dollar, so your change is
2 quarters,
0 dimes, and
1 nickels
bob@Bennys-MacBook-Pro HW1 % java VendingChange.java 25
You bought a item for 25 cents and gave me a dollar, so your change is
3 quarters,
0 dimes, and
0 nickels
bob@Bennys-MacBook-Pro HW1 % java VendingChange.java 100
You bought a item for 100 cents and gave me a dollar, so your change is
0 quarters,
0 dimes, and
0 nickels
```

3 Problem C

```java
// OperatorPrecedence.java
// This program takes in 3 numbers(INPUT) and uses them in a equation to calculate the cube root of a number(OUTPUT)

public class OperatorPrecedence {
    public static void main(String[] args){
        //Set the value of x to the first input
        double x = Double.parseDouble(args[0]);

        //Set the value of y to the second input
        double y = Double.parseDouble(args[1]);

        //Set the value of z to the third input
        double z = Double.parseDouble(args[2]);

        //Plugs in values into equation then cubes it
        double answer = Math.cbrt(Math.pow(x, 2) + Math.pow(y, 2) - Math.abs(z));

        //Print the cube root
        System.out.printf("Cube Root is: %.2f\n", answer);
    }
}
```

Test Cases

```
bob@Bennys-MacBook-Pro HW1 % java OperatorPrecedence.java 5 5 8
Cube Root is: 3.48
bob@Bennys-MacBook-Pro HW1 % java OperatorPrecedence.java 0 0 0
Cube Root is: 0.00
bob@Bennys-MacBook-Pro HW1 % java OperatorPrecedence.java 100 100 -100
Cube Root is: 27.10
```

4 Problem D

D.1

The problem the orginal code had was that only m_2 was being divided and not the whole $G * m_1 * m_2$ equation. By adding in parantheses, we can now divide the whole $G * m_1 * m_2$ equation by $r * r$.

D.2

```
66 // Force.java
67 // This program takes in the mass of 2 objects(INPUT) and
       distance between centers of the masses(INPUT) and calculates
        the force(OUTPUT)
68
69 public class Force {
70     public static void main(String[] args){
71         //Set the value of m1 to the first input of mass
72         double m1 = Double.parseDouble(args[0]);
73
74         //Set the value of m2 to the second input of mass
75         double m2 = Double.parseDouble(args[1]);
76
77         //Set the value of r to the input of distance between
               centers of the masses
78         double r = Double.parseDouble(args[2]);
79
80         //Calculate the force
81         double force = (6.67 * Math.pow(10, -11)) * ((m1 * m2)
               / Math.pow(r, 2));
82
83         //Print the force
84         System.out.printf("Force is: %.2f\n", force);
85     }
86 }
```

Test Cases

```
bob@Bennys-MacBook-Pro HW1 % java Force.java 1e+5 2e+5 1e+0
Force is: 1.33
bob@Bennys-MacBook-Pro HW1 % java Force.java 5e+0 5e+0 1e+0
Force is: 0.00
bob@Bennys-MacBook-Pro HW1 % java Force.java 5e+10 5e+2 1e+1
Force is: 16.68
```