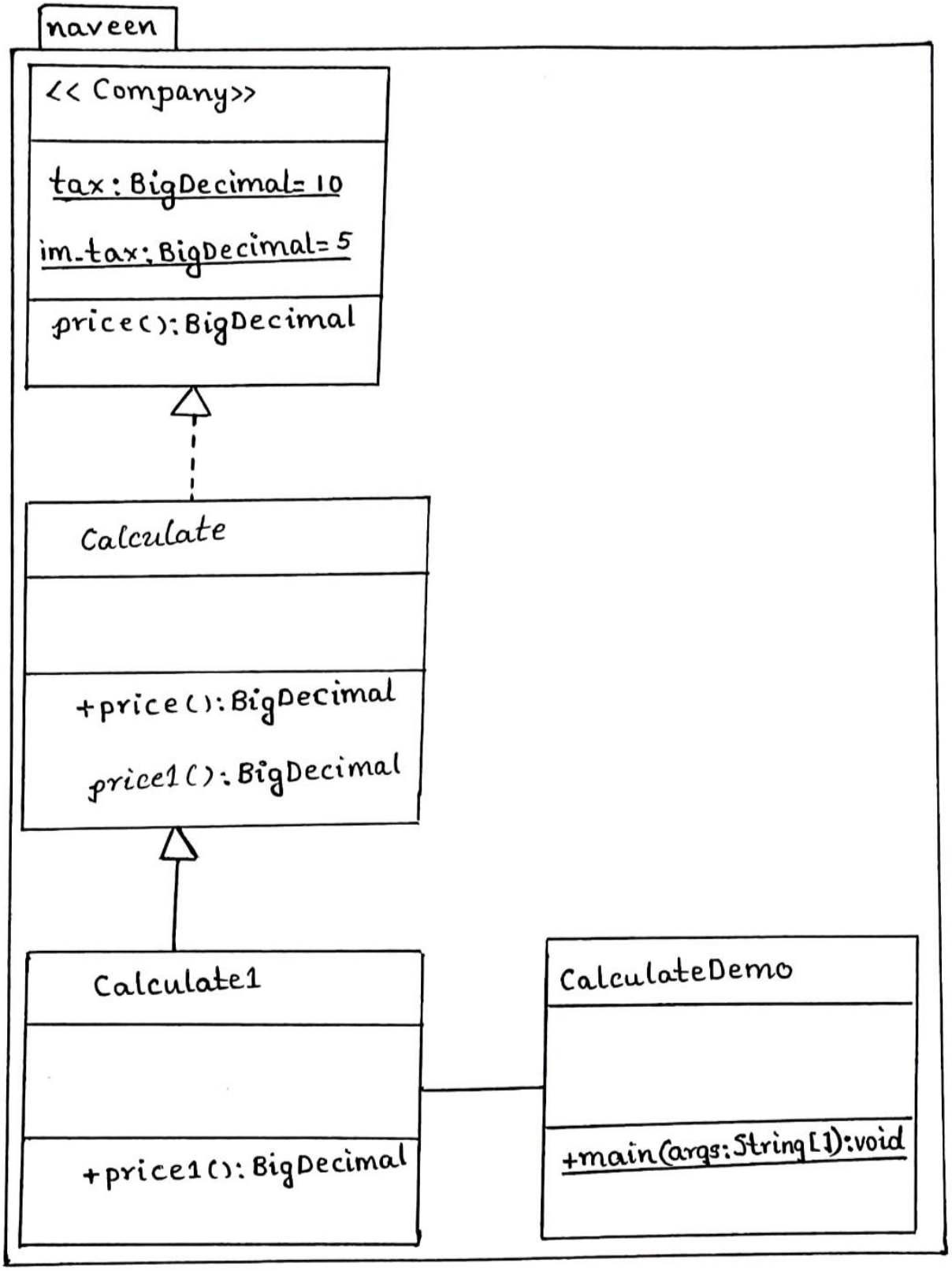
SKILLING EXERCISE-5

Name: Badisa Naveen

Reg.no: 2000031509

CLASS DIAGRAM:



Interface:

1. Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, no body). It specify what a class must do and not how. It is the blueprint of the class.
2. To declare an interface, use **interface** keyword. It is used to provide total abstraction and all the members are final and static
3. To implement interface, we use **implements** keyword and denoted by

Why do we use interface ?

- Since java does not support multiple inheritance in case of class, but by using interface it can achieve multiple inheritance
- It is also used to achieve loose coupling

Abstract Class:

1. A class which is declared with the abstract keyword is known as an abstract class . It can have abstract and non-abstract methods (method with the body).
2. For any abstract java class we are not allowed to create an object i.e., for abstract class instantiation is not possible.

Note:

In interface are abstract methods where as in abstract class at least one abstract method.

BigDecimal:

1. The BigDecimal class provides operations on double numbers for arithmetic, scale handling, rounding, comparison, format conversion and hashing
2. It can handle very large and very small floating point numbers with great precision i.e Big Decimal provides us with the exact answer.

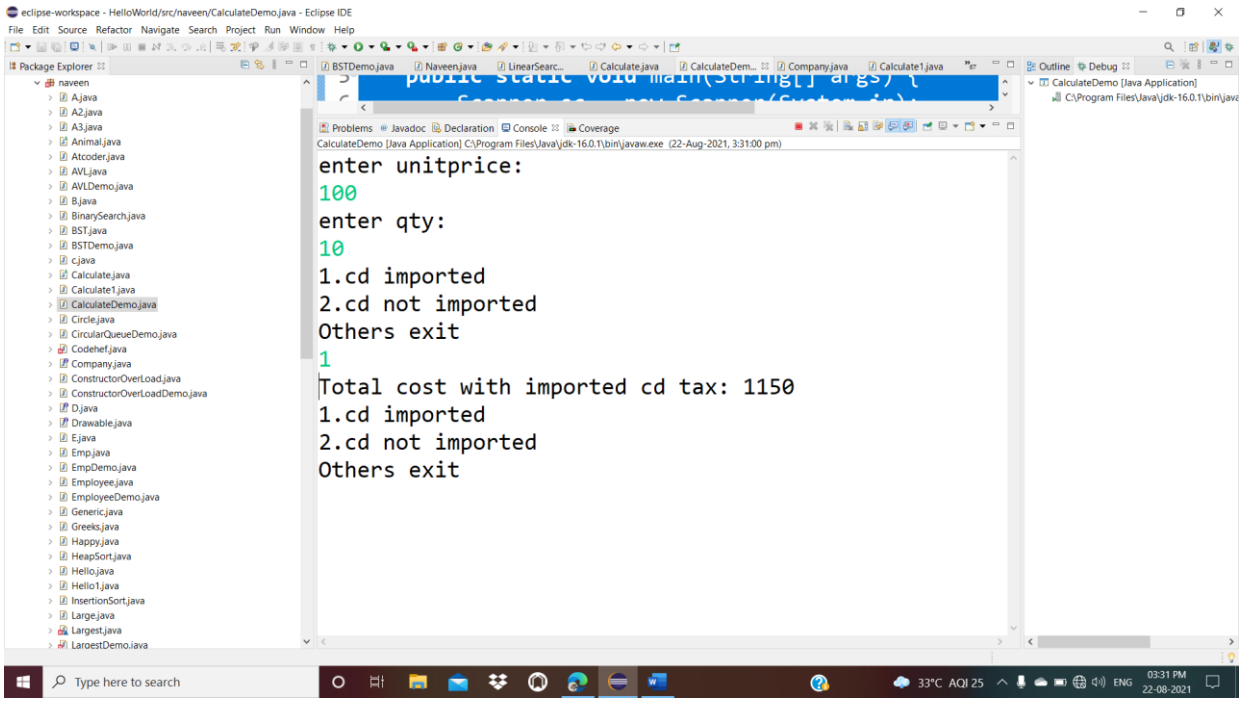
Code:

```
package naveen;
import java.math.BigDecimal;
public interface Company {
        final static BigDecimal tax = new BigDecimal("10");
         final static BigDecimal imported_tax = new BigDecimal("5");
        public abstract BigDecimal price();
}
package naveen;
import java.math.BigDecimal;
public abstract class Calculate implements Company {
       public BigDecimal price() {
              return Company.tax;
       }
   public abstract BigDecimal price1();
}
package naveen;
import java.math.BigDecimal;
public class Calculate1 extends Calculate {
  public BigDecimal price1()
  {
         return imported_tax;
  }
}
 package naveen;
 import java.math.BigDecimal;
 import java.util.Scanner;
 public class CalculateDemo {
       public static void main(String[] args) {
              Scanner sc = new Scanner(System.in);
              Calculate1 c = new Calculate1();
              System.out.println("enter unitprice: ");
              BigDecimal unitPrice =sc.nextBigDecimal();
              System.out.println("enter qty: ");
              BigDecimal qty =sc.nextBigDecimal();
              boolean repeat = true;
```

```
            while(repeat)
            {
                System.out.println("1.cd imported\n2.cd not
imported\nOthers exit");
                switch(sc.nextInt())
                {
                case 1:
                    BigDecimal a = c.price();
                    a = a.add(unitPrice);
                    a = a.add(c.price1());
                    a = a.multiply(qty);
                    System.out.println("Total cost with imported cd tax:
"+a);
                    break;
                case 2:
                    BigDecimal b = c.price();
                    b = b.add(unitPrice);
                    b = b.multiply(qty);
                    System.out.println("Total cost without imported cd
tax: "+b);
                    break;
                    default:
                        System.out.println("Incorrect input , so program
terminated ");
                        repeat=false;

                    }
            }
            sc.close();

      }

}
}
```

ScreenShots:

Total cost if cd imported:



Total cost if cd is not imported: