LAB EXERCISE # 9[1]

REMEMBER THE RULES FOR SUBMISSION.

- Students must try to complete this exercise during class on 11/06 and have it approved (demo it to a TA) no later than the next lab session.
- You can demo it to a TA during the class or during TA office hrs.
- Must be shown to a TA or instructor and be reviewed for correctness before it gets submitted.
- Upon approval by TA, you can submit it on Blackboard as single zip file. Name the file <lastname>_<firstname>_lab-ex-9.zip Watch for a checkmark in the blackboard grade center next to your name.
- Lack of check mark will be interpreted as an unauthorized submission.

Using abstract methods and polymorphism/Method Overriding.

- Write the code for a class A.
    - This class has instance variables :
      public int a
      private double aa
    - Write the constructors/accessor/mutator methods and the toString method. The non default constructor takes two arguments (used to initialize the two instance variables). Include a System.out.println statement to indicate that the particular constructor of the class A is executing now.
    - It has a method m2 implemented twice, each time with different arguments.

```
public  int  m2(char a)
{
        int  x=(int)a;
        System.out.println("m2 of A is executing now");
        return x;
}
public int m2(int  x1)
```

[1] Acknowledge material provided by George Koutsogiannakis

```
{
    int y=10+x1; //get the returned value from the first version and add x1
   System.out.println("second version of m2 in A is executing now");
   return y;
}
```

- It has a method m3

```
public void m3()
{
        System.out.println("m3 of A is executing now");
}
```

- This class has an abstract method:

```
public abstract int m1();
```

- Write the code for a class B <u>which inherits A</u>
  - B has the instance variable
    ```
    public  String s
    ```
  - Write the constructors/accessor/mutator methods. The constructors utilize the superclass's constructors and then initialize the specialized variable of this class. For the default constructor, initialize the string s to "test". Include a System.out.println statement to indicate that the particular constructor of this class is executing now. Make sure that the correct number of arguments appear in the non default constructor.
  - Write the toString method which includes the toString of the super class.
  - It implements the abstract method m1 as follows (you may need to modify):

```
public int m1()
{
    int  i1= 5*a+(int)aa;
   System.out.println("m1 implementation of B is executing now");
   return i1;
}
```

- Write the code for a class C <u>which inherits A</u>
  - C has the instance variable
    ```
    public  double  c
    ```
  - Write the constructors/accessor/mutator methods. The constructors utilize the super constructors then they initialize the specialized variable. Include a System.out.println statement to indicate that the particular constructor of class

C is executing now. Make sure that the correct number of arguments appear in the non default constructor.

- Write the toString method which includes the toString of the superclass.
- It implementst he abstract method m1 as follows:

```
public int m1()
{
        int  i2= a+(int)(c/2);
        System.out.println("m1 implementation of C is executing now");

        return i2;
}
```

- Write the code for a **<u>class D which inherits C.</u>**
  D has the instance variable: public String str.
  - Write the constructors/accessor/mutator methods. The constructors utilize the super constructors then they initialize the specialized variable. Include a System.out.println statement to indicate that the particular constructor of class D is executing now. Make sure that the correct number of arguments appear in the non default constructor.
  - It has a method m3 which is an overridden version of m3 from class A.

```
public void m3()
{
        //place the code here to execute the code from method m3 of class A
first
        System.out.println("I am executing m3 as implemented in  class D");
}
```

  - It also overrides the first version of method m2 of A

```
public  int m2(char c)
{
```

  If the argument c is the character 'e' the method invokes the method m2 of the super class and passes the character 'e' as argument and then displays: "The m2(char c) version of D is executing now with returned value= "+x
  Otherwise the method invokes the method m2 of the super class and passes the argument 100 and then displays : "The m2(int x1) version of D is executing now with returned value= "+x
  Finally the method returns the value returned by the proper m2 version of the super class A.

}

- Write the code for a class Client.
    - FIRST OUTPUT
      Create objects of B and C using the default constrcutors and then use polymorphism to invoke the two versions of method m1. In doing so you would need a reference of A and you will be using the A reference to invoke m1 twice. Invoke first the version from class B and then the version from class C. Output the returned value each time.
    - SECOND OUTPUT
      Create an object of D. Next invoke (again using polymorphism) the overridden method m3 from class D
    - THIRD OUTPUT
      In the next two invocations use the object of D directly to make the invocations (do not use polymorphism)
      1. Invoking the overridden method m2 of D, by passing it the character e as argument
      2. Invoke the second version of m2 from A, by passing an appropriate character as argument.

      SAMPLE OUTPUT
      >java client
      First Output

      Default constructor of Class A
      Default constructor of Class B
      Default constructor of Class A
      Default constructor of Class C
      m1 implementation of B is executing now
      m1 implementation of C is executing now

      Second Output

      Default constructor of Class A
      Default constructor of Class C
      Default constructor of Class D
      m1 implementation of C is executing now

      Third Output

m2 of A is executing now
The m2(char c) version of D is executing now with returned value= 101
second version of m2 in A is executing now
The m2(int x1) version of D is executing now with returned value= 110