Problem1)

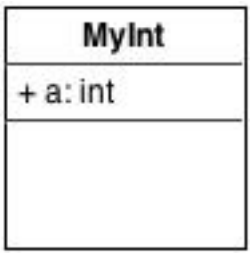
Write and run the following code:

```
int a = 5;
int b = a;
System.out.println("a is " + a + " and b is " + b);
a = a + 1;// or a++;
System.out.println("a is " + a + " and b is " + b);
```

Notice here that the value of the variable a is copied to the variable b, and each variable has its own independent value.

Now, write the following class:

```
public class MyInt
{
     public int a;
}
```



After writing the class, run the code below in a separate program:

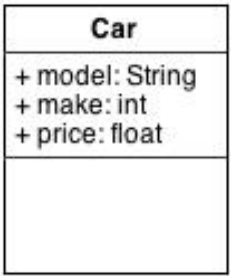
```
MyInt x = new MyInt();
x.a = 5;
MyInt y = x;
System.out.println("x.a is " + x.a + " and y.a is " + y.a);
x.a = x.a + 1;
System.out.println("x.a is " + x.a + " and y.a is " + y.a);
```

Now, can you find the difference between the 1st scenario and the 2nd one? OK. In the first scenario, the values were independent due to the fact that the value was copied to the other variable so that each has its own value. However, in the second scenario, the reference is copied and both variables are referencing the same object in the memory.

NO SOLUTION NEEDED HERE.

Problem2)

Create a class Car that has the model of the car (String), the make year (int), and the price (float).



Now, after creating the class, write the following lines of code in a separate program. However, you should add some code to it where marked, so that you can modify the object referenced by c without using the reference variable c:

```
Car c = new Car();
c.model = "GM";
c.make = 2011;
c.price = 60000.0f;
System.out.println("the car c is of type " + c.model +
" and the year of make is " + c.make + " and its price is " +
c.price);
//continue the program here ------
```

SOLUTION:

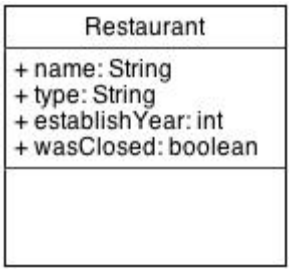
```
//the Car class is
public class Car
{
      public String model;
      public int make;
      public float price;
}

//the code to be added
Car d = c;
d.model = "Ford";
d.make = 2013;
d.price = 100000.0f;
System.out.println("the car c is of type " + c.model +
" and the year of make is " + c.make + " and its price is " +
c.price);
```

Now answer this question:
Can you do the same way of modification with variables of type `double`?

The answer:
No, it is not possible. The type double is a primitive data type, and primitive data types deal only with values, and they always kept in the stack. However, reference variables keep references and it is possible to have more than variable referencing the same object in the heap. Thus, we cannot do the same with variables of type double.

Problem3)



Create a class Restaurant that has its name (String), the cuisine type (String), the establishment year (int), and a variable to determine if it was closed before (boolean).

Now, after creating the class, write the following lines of code in a separate program. However, you should add some code to it where marked, so that you can verify if `(rest1 == rest2)` also verify if `(rest2 == rest3)` .

Hint: you don't have to use if statements.

```
Restaurant rest1 = new Restaurant();
rest1.name = "Italian nights";
rest1.type = "Italian";
rest1.establishYear = 2010;
rest1.wasClosed = true;

Restaurant rest2 = new Restaurant();
rest2.name = "Layali Najd";
rest2.type = "Saudi";
rest2.establishYear = 2008;
rest2.wasClosed = false;

Restaurant rest3 = new Restaurant();
rest3.name = "Layali Najd";
rest3.type = "Saudi";
rest3.establishYear = 2008;
rest3.wasClosed = false;

//continue the program here ------
```

SOLUTION:

```
//the Restaurant class is
public class Restaurant
{
	public String name;
	public String type;
	public int establishYear;
	public boolean wasClosed;
}


//the code to be added
boolean first = rest1 == rest2;
boolean second = rest2 == rest3;

System.out.println("the result of rest1 == rest2 is: " + first);
System.out.println("the result of rest2 == rest3 is: " + second);
```