Lab Assignment 09



Inspiring Excellence

Course Code:	CSE111
Course Title:	Programming Language II
Topic:	Polymorphism
Number of Tasks:	10

1.a. Write the Pokemon class so that the following code generates the output below:

Driver Code	Output
<pre>public class PokemonTester{ public static void main(String[] args){ Pokemon pikachu = new Pokemon("Pikachu"); pikachu.attack(); pikachu.attack("Thunderbolt"); pikachu.attack("Iron Tail", 90); } }</pre>	Pikachu attacks with a basic move! Pikachu uses Thunderbolt! Pokemon uses Iron Tail with power 90!

1.b. What type of polymorphism is depicted in the code above?

Task 2

Write the **Mango** and the **Jackfruit** classes derived from Fruit class so that the following code generates the output below:

```
public class Fruit{
  private boolean formalin = false;
  private String name = "";
  public Fruit(boolean formalin, String name){
    this.formalin = formalin;
    this.name = name;
  }
  public String getName(){
    return name;
  }
  public boolean hasFormalin(){
    return formalin;
  }
}
```

Driver Code	Output
<pre>public class FruitTester{ public static void testFruit(Fruit f){ System.out.println("Printing Detail"); if(f.hasFormalin()){ System.out.println("Do not eat the "+f.getName()+"."); System.out.println(f); }else{ System.out.println("Eat the "+f.getName()+"."); System.out.println(f); } } public static void main(String [] args){ Mango m = new Mango(); testFruit(m); Jackfruit j = new Jackfruit(); testFruit(j);</pre>	Printing Detail Do not eat the Mango. Mangos are bad for youPrinting Detail Eat the Jackfruit. Jackfruits are good for you

```
}
```

Write the **CSEStudent** and **CSE111Student** classes derived from **Student** class so that the following code generates the output below:

```
Parent Class
public class Student{
 public String msg = "I love BU";
 public String shout(){
    return msg;
 }
}
                    Driver Code
                                                                          Output
public class StudentTester{
                                                      I love BU
 public static void printShout(Student s){
                                                      I want to transfer to CSE
    System.out.println("----");
                                                      I love Java Programming
    System.out.println(s.msg);
    System.out.println(s.shout());
                                                      I love BU
                                                      I love BU
 public static void main(String [] args){
                                                       -----
    Student s = new Student();
                                                      I love BU
    CSEStudent cs = new CSEStudent();
                                                      I want to transfer to CSE
   CSE111Student cs111 = new CSE111Student();
    System.out.println(s.msg);
                                                      I love BU
   System.out.println(cs.msg);
                                                      I love Java Programming
    System.out.println(cs111.msg);
    printShout(s);
   printShout(cs);
   printShout(cs111);
 }
```

Task 4

Write the **PlatinumCard** and **SignatureCard** classes derived from **CreditCard** class so that the following code generates the output below.

}

Note: Platinum card users initially have 100 reward points and will get 2 reward points for spending 100 taka each. Signature card users initially have 200 reward points and will get 4 reward points for spending 100 taka each. Signature card users are allowed to bring upto 5 companions at lounges.

```
public class CreditCard {
   public String cardHolder;
   public String accountNo;
   public int rewardPoints;
   public CreditCard(String cardHolder, String accountNo, int rewardPoints){
      this.cardHolder = cardHolder;
      this.accountNo = accountNo;
      this.rewardPoints = rewardPoints;
   }
   public void cardDetails(){
      System.out.println("Card Holder Name: " + cardHolder);
      System.out.println("Account Number: " + accountNo);
      System.out.println("Reward point gained: " + rewardPoints);
```

}

```
Driver Code
                                                                             Output
public class CardTester {
                                                              Previous Reward Points: 100
 public static void main(String[] args) {
                                                              Reward points after spending 200
   CreditCard card1 = new PlatinumCard("Ali", "345 127");
                                                              taka: 104
   CreditCard card2 = new SignatureCard("Rahul", "514 123");
                                                              _____
   CreditCard card3 = new SignatureCard("Rohan", "147 965");
                                                              Card Holder Name: Ali
   CreditCard [] cards = {card1, card2, card3};
                                                              Account Number: 345 127
   for (int i = 0; i < cards.length; i++)</pre>
                                                              Reward point gained: 104
                                                              ===========
                                                              Previous Reward Points: 200
      System.out.println("=======");
                                                              Reward points after spending 500
     if (cards[i] instanceof SignatureCard)
                                                              taka: 220
                                                              ===========
       SignatureCard new_card = (SignatureCard) cards[i];
                                                              Card Holder Name: Rahul
       new_card.spendCash(500);
                                                              Account Number: 514 123
                                                              Reward point gained: 220
                                                              Possible Number of Companions for
     else if (cards[i] instanceof PlatinumCard)
                                                              Lounge: 5
                                                              ===========
       PlatinumCard new_card = (PlatinumCard) cards[i];
                                                              Previous Reward Points: 200
       new_card.spendCash(200);
                                                              Reward points after spending 500
                                                              taka: 220
     System.out.println("=======");
                                                              ===========
      cards[i].cardDetails();
                                                              Card Holder Name: Rohan
                                                              Account Number: 147 965
   }
                                                              Reward point gained: 220
 }
                                                              Possible Number of Companions for
}
                                                              Lounge: 5
```

Design a set of classes for a Fantasy Game Character System. There is a parent class called **Character**. From it, there are two different child classes: **Warrior** and **Mage**. Additionally, there is a subclass called **Paladin** that extends Warrior. Each character has:

- name (String)
- level (int)

Lastly, you need to Override the .equals() method inside the parent class

```
public class Character {
  public String name;
  public int level;

public Character(String name, int level) {
    this.name = name;
    this.level = level;
  }

public void specialMove() {
    System.out.println("Character uses a generic move.");
  }

// Override the .equals() method
}
```

Driver Code	Output
<pre>public class GameTester { public static void main(String[] args) { Character c1 = new Paladin("Arthur", 10); Character c2 = new Mage("Merlin", 12); Character c3 = new Warrior("Leon", 10); c1.specialMove(); c2.specialMove(); c3.specialMove(); if (c1 instanceof Paladin) { Paladin p = (Paladin) c1; p.specialMove(); }</pre>	Arthur unleashes a holy strike! Merlin casts a powerful fireball! Leon performs a heavy sword slash! Arthur unleashes a holy strike! c3 equals w1? true c2 equals m1? false
<pre>Warrior w1 = new Warrior("Leon", 10); System.out.println("c3 equals w1? " + c3.equals(w1));</pre>	
<pre>Mage m1 = new Mage("Merlin", 15); System.out.println("c2 equals m1? " + c2.equals(m1)); } </pre>	

Write the Garage, Bike and Car class. **Car, Bike** are child classes of **Vehicle** class. But **Garage** is neither a parent nor a child class. The Garage class has **two arrays as instance variables** called *cars* and *bikes* that can store **Car and Bike objects**.

Hint: In this task you'll need to use the instanceof keyword and downcasting.

```
Vehicle Class
public class Vehicle {
    private String brand;
    private int year, wheels;
    public Vehicle(String b, int y){
        this.brand = b;
        this.year = y;
    }
    public String getBrand(){
        return this.brand;
    public int getYear(){
        return this.year;
    }
    public void setWheels( int w ){
        this.wheels = w;
    }
    public int getWheels(){
        return this.wheels;
    }
    public String toString(){
        return "Brand: "+this.brand+", Year: "+this.year+", Wheels: "+this.wheels;
}
```

```
public class Gandalf {
      public void method1(){
        System.out.println("Gandalf 1");
3
4
5
6
      public void method2(){
        System.out.println("Gandalf 2");
7
        method1();
8
9
      }
10
    public class Bilbo extends Gandalf{
11
      public void method1(){
12
13
        System.out.println("Bilbo 1");
14
15
    public class Gollum extends Gandalf{
16
17
      public void method3(){
        System.out.println("Gollum 3");
18
19
20
    public class Frodo extends Bilbo{
21
      public void method1(){
22
        System.out.println("Frodo 1");
23
        super.method1():
24
25
26
27
      public void method3(){
        System.out.println("Frodo 3");
28
29
30
```

Assuming the following variables have been defined:

```
Gandalf var1 = new Frodo();
Gandalf var2 = new Bilbo();
Gandalf var3 = new Gandalf();
Object var4 = new Bilbo();
Bilbo var5 = new Frodo();
Object var6 = new Gollum();
```

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be Detected.

	Statement	Output
1	var1.method1();	
2	var2.method1();	
3	<pre>var4.method1();</pre>	
4	<pre>var6.method1();</pre>	
5	<pre>var1.method2();</pre>	
6	<pre>var3.method2();</pre>	
7	<pre>var4.method2();</pre>	
8	<pre>var5.method2();</pre>	
9	<pre>var6.method2();</pre>	
10	((Frodo)var4).method3();	
11	((Frodo)var6).method2();	
12	((Gollum)var1).method3();	
13	((Gollum)var4).method1();	
14	<pre>((Gandalf)var1).method2();</pre>	
15	((Frodo)var4).method1();	
16	((Gollum)var6).method2();	
17	((Gandalf)var2).method1();	
18	((Bilbo)var6).method2();	
19	((Frodo)var1).method3();	
20	((Gandalf)var5).method3();	

1	public class Sue {
2	void method1() {
3	System.out.println("sue 1");
4	}
5	<pre>void method3() {</pre>
6	System.out.println("sue 3");
7	}
8	}
9	
10	public class Blue {
11	<pre>void method1() {</pre>
12	System.out.println("blue 1");
13	method3();
14	}
15	void method3() {
16	System.out.println("blue 3");
17	}
18	}
19	
20	public class Moo extends Blue {
21	<pre>void method2() {</pre>
22	<pre>super.method3();</pre>
23	System.out.println("moo 2");
24	this.method3();
25	}
26	void method3() {
27	System.out.println("moo 3");
28	}
29]}
30	
31	public class Crew extends Moo {
32	<pre>void method1() {</pre>
33	System.out.println("crew 1");
34	}
35	void method3() {
36	System.out.println("crew 3");
37	}
38	}

Assuming the following variables have been defined:

```
Moo var1 = new Crew();
Blue var2 = new Moo();
Object var3 = new Sue();
Sue var4 = new Sue();
Blue var5 = new Crew();
Blue var6 = new Blue();
```

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

	Statement	Output
1	var1.method1();	
2	var2.method1();	
3	var3.method1();	
4	var4.method1();	
5	<pre>var5.method1();</pre>	
6	<pre>var6.method1();</pre>	
7	<pre>var1.method3();</pre>	
8	<pre>var2.method3();</pre>	
9	<pre>var3.method3();</pre>	
10	((Blue)var1).method1();	
11	((Crew)var1).method2();	
12	((Sue)var1).method3();	
13	((Blue)var3).method1();	
14	((Crew)var3).method1();	
15	((Sue)var3).method3();	
16	((Moo)var2).method2();	
17	((Crew)var3).method2();	
18	((Moo)var5).method2();	
19	((Moo)var6).method2();	
20	((Moo)var2).method1();	

1	public class Foo {
2	String name = "foo";
3	public void call1() {
4	System.out.println("Foo 1");
5	}
6	<pre>public void call2() {</pre>
7	call1();
8	System.out.println("Foo 2");
9	}
10	}
11	
12	public class Bar extends Foo {
13	<pre>public void call2() {</pre>
14	System.out.println("Bar 2");
15	}
16	<pre>public void call3() {</pre>
17	System.out.println("Bar 3");
18	}
19	}
20	
21	public class Buzz extends Bar {
22	String name = "Buzz";
23	<pre>public void call1() {</pre>
24	<pre>System.out.println("Buzz 1");</pre>
25	}
26	<pre>public void call4() {</pre>
27	call3();
28	<pre>System.out.println("Buzz 4");</pre>
29	}
30	}
31	public class Bux extends Foo {
32	String name = "Bux";
33	<pre>public void call1() {</pre>
34	System.out.println("Bux 1");
35	}
36	<pre>public void call3() {</pre>
37	<pre>System.out.println("Bux 3");</pre>
38	}
39	}

Assuming the following variables have been defined:

```
Foo foo1 = new Foo();
Bar bar1 = new Bar();
Bux bux1 = new Bux();
Foo foo2 = new Buzz();
Bar bar2 = new Buzz();
Object obj1 = new Foo();
```

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

	Statement	Output
1	bar1.call1();	
2	foo2.call1();	
3	foo2.call2();	
4	bar2.call3();	
5	<pre>System.out.println(bar1.name);</pre>	
6	<pre>System.out.println(bar2.name);</pre>	
7	<pre>System.out.println(((Buzz)bar2).name);</pre>	
8	((Buzz)bar1).call4();	
9	((Bar)foo1).call3();	
10	((Foo)bux1).call1();	
11	((Bux)foo1).call1();	
12	bux1.call1();	
13	bux1.call2();	
14	((Foo)foo2).call2();	
15	((Buzz)obj1).call3();	
16	((Buzz)obj1).call2();	
17	((Bux)foo2).call2();	
18	((Buzz)obj1).call1();	
19	<pre>System.out.println(foo2.name);</pre>	
20	<pre>System.out.println(((Bux)foo2).name);</pre>	

```
public class Caramel extends SilkOreo{
    String texture = "Softy";
3
    public void method1() {
4
       System.out.println("Caramel m1");
5
    public void method4() {
    System.out.println("Caramel m4");
7
8
    public String toString(){
9
10
      method2();
      return "Caramel is "+ texture;
11
12
13 |}
14 public class Chocolate{
    String texture = "Chocolaty";
    public void method1() {
16
17
      method2();
      System.out.println("Chocolate m1");
18
19
    public void method2() {
20
21
      System.out.println("Chocolate m2");
22
    public String toString(){
23
      method2();
24
25
      return "Chocolate is "+ texture;
26
27
28 public class DairyMilk extends Chocolate{
    String texture = "Yummy";
29
30
    public void method2() {
       System.out.println(this.texture);
31
      System.out.println("DairyMilk m2");
32
33
34
    public void method3() {
       System.out.println("DairyMilk m3");
35
36
37 |}
38 public class KitKat extends Chocolate{
   String texture = "Crunchy";
39
    public void method1() {
```

```
System.out.println("KitKat m1");
42
43
     public void method4() {
44
       System.out.println("KitKat m4");
45
46
     public String toString(){
47
       method2();
48
       return "KitKat is "+ texture;
49
50 |}
51 public class SilkOreo extends DairyMilk{
     String texture = "Silky";
52
    public void method1() {
53
54
       super.method1();
55
       System.out.println("SilkOreo m1");
56
57
    public void method3() {
       System.out.println("SilkOreo m3");
58
       System.out.println(this);
59
60
61 |}
```

Assuming the following variables have been defined:

```
Chocolate choco1 = new Chocolate();
KitKat kit = new KitKat();
DairyMilk dairyMilk1 = new DairyMilk();
DairyMilk dairyMilk2 = new SilkOreo();
Object obj1 = new DairyMilk();
Object obj2 = new KitKat();
Chocolate caramel1 = new Caramel();
```

- The output produced by the statement in the left-hand column, should be written in the right-hand column
- If the statement produces more than one line of output, indicate the line breaks with slashes as in "a/b/c" to indicate three lines of output with "a" followed by "b" followed by "c".
- If the statement causes an error, fill in the right-hand column with either the phrase "compiler error" or "runtime error" to indicate when the error would be detected.

	Statement	Output
1	<pre>choco1.method1();</pre>	
2	<pre>dairyMilk1.method1();</pre>	
3	<pre>dairyMilk2.method4();</pre>	
4	<pre>caramel1.method1();</pre>	
5	<pre>System.out.println(caramel1);</pre>	
6	<pre>System.out.println(caramel1.texture);</pre>	
7	<pre>((Chocolate)kit).method2();</pre>	
8	<pre>((SilkOreo)dairyMilk2).method3();</pre>	
9	((DairyMilk)kit).method2();	
10	<pre>((Chocolate)kit).method3();</pre>	
11	<pre>((Chocolate)dairyMilk2).method1();</pre>	
12	((Chocolate)obj1).method2();	
13	((Caramel)obj1).method2();	
14	((SilkOreo)obj2).method3();	
15	<pre>System.out.println(((Object)choco1).toString());</pre>	
16	<pre>System.out.println(((Chocolate)kit).texture);</pre>	