CS1421: Object Oriented Programming Lab

LAB # 07

Instructors
Samia Bashir
Safiyah Batool

Lab 06
Object Class, Polymorphism and ArrayList

Objective
After completing this lab, the students should be able to
- Understand toString() method from object class
- Understand Polymorphism
- Understand ArrayList

Polymorphism:
polymorphism. There are two types of polymorphism in java:
1) Static Polymorphism also known as compile time polymorphism
2) Dynamic Polymorphism also known as runtime polymorphism

```java
public class VariablePolymorphsim {
   public static void main(String[] args) {
       int a=2;
       float b=4f;
       System.out.println(a+b);
       int c=2;
   }
}
```

Runtime Polymorphism (or Dynamic polymorphism)

It is also known as Dynamic Method Dispatch. Dynamic polymorphism is a process in which a call to an overridden method is resolved at runtime, that’s why it is called runtime polymorphism.

Method Overriding:

```java
public class One {
   public double calculate(int x)
   {
       return Math.pow(2,x);
   }
}
```

```java
public class Two {
    //Overriding
   public double calculate(int x)
   {
       return Math.sqrt(x);
   }
}
```

```java
public class Calculate {
   public static void main(String[] args) {
       Two t= new Two();
       One  o= new One();
       System.out.println(t.calculate(25));
       System.out.println(o.calculate(25));
   }
}
```

Compile time Polymorphism (or Static polymorphism)

Polymorphism that is resolved during compiler time is known as static polymorphism. Method overloading is an example of compile time polymorphism.

Method Overloading: This allows us to have more than one method having the same name, if the parameters of methods are different in number, sequence and data types of parameters.

```java
public class overloading {
   int add(int a,int b) {
       return a + b;
   }

   int add(int a,int b, int c){

    return a+b+c;
   }

   float add(float a,float b)
   {
       return a+b;
   }
   float add(float a,float b, float c){
       return a+b+c;
   }
}
```

```
public class TestOverloading {
   public static void main(String[] args) {
       overloading o= new overloading();
       System.out.println(o.add(3,4));
       System.out.println(o.add(4,2,5));
       System.out.println(o.add(3.33f,4.55f));

System.out.println(o.add(3.532f,5.345f,34.54f));
   }
}
```

Method Signature:

Method signature is method name along with method parameters.

Polymorphism in Static Methods:

```
public class StaticPoly {
   static void calculate(int x)
   {
       System.out.println(Math.pow(2,x));
   }
}
```

```
public class StaticPoly2 extends StaticPoly {

 static void calculate(int x)
 {
     System.out.println(Math.sqrt(x));
 }

}
```

```
public class staticPolytesst {

   public static void main(String[] args) {
       StaticPoly.calculate(2);

       StaticPoly2.calculate(2);
   }
}
```

Coercion: Automatic conversion between different data types.
Conversion: Explicit change in the data type specified by the cast operator.

```
public class ex1 {
         //Type Casting
   public static void main(String[] args) {
       //Up casting
       System.out.println("Widening");
       char a='A';
       int b=a;                       //widening,
implicit casting, up casting
       System.out.println(a);
       System.out.println(b);

       System.out.println("Narrowing");
       //Downcasting
       int c=65;
       //char d=c; //Error????
       char d=(char)c;     //narrowing, explicit
casting, down casting
```

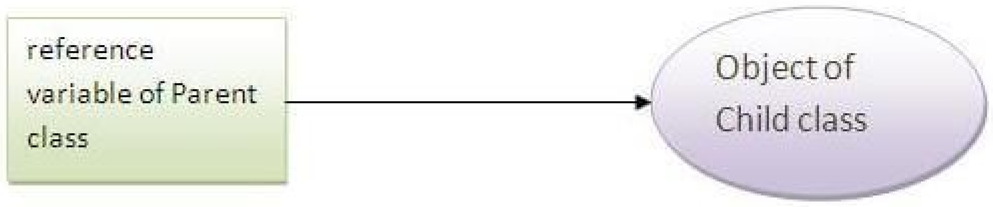
```
        System.out.println(c);
        System.out.println(d);


    }

}
```

Casting Referenced Data Types:

Upcasting: (Dynamic Polymorphism)
When reference variable of Parent class refers to the object of Child class, it is known as upcasting.



```
class A{}
class B extends A{}

A a=new B();//upcasting

Example:
class Bike{
   void run(){System.out.println("running");}
 }
class cycle extends Bike
{
 void run(){System.out.println("running slow");}
}
 class Splender extends Bike{
   void run(){System.out.println("running safely with 60km");}
```

```
  public static void main(String args[]){
    Bike a = new Bike();
    Bike c= new cycle(); //upcasting
   Bike b = new Splender();

    a.run();
    c.run();
    b.run();

  }
 }
```

Generalization and Specialization:

Example:

```
public class Billing {
        public void getBill(int units)
        {
        System.out.println("Total Bill"+units*5);
        }


        public int getrate(){
        return 5;
        }
}
```

```
public class Domestic extends Billing{
        @Override
        public void getBill(int units)
        {
        System.out.println("Total Bill"+units*2.5);
        }
}
```

```
public class testBilling {
        public static void main(String[] args) {
        Domestic april= new Domestic();


        april.getBill(5);

       //Generalization or Upcasting
        Billing may= new Domestic();
        may.getBill(5);
        System.out.println(may.getrate());

       //Specialization
      // Domestic june= (Domestic) new Billing(); //Run-time error

       //Another way of specialization

        Billing b=  new Domestic();
        Domestic june=(Domestic) b;
        june.getBill(5);
        }
}
```

ArrayList

- An ArrayList object can be used to store a list of objects. You can create an array to store objects. But, once the array is created, its size is fixed. Java provides the ArrayList class, which can be used to store an unlimited number of objects.
- ArrayList is known as a generic class with a generic type E. You can specify a concrete type to replace E when creating an ArrayList. For example, the following statement creates an ArrayList and assigns its reference to variable cities. This ArrayList object can be used to store strings.
- ArrayList<String> cities = new ArrayList<String> ();
- The following statement creates an ArrayList and assigns its reference to variable dates. This ArrayList object can be used to store dates.
- ArrayList<java.util.Date> dates = new ArrayList<java.util.Date> ();
- You cannot use the get(index) and set(index, element) methods if the element is not in the list. It is easy to add, insert, and remove elements in a list, but it is rather complex to add, insert, and remove elements in an array. You have to write code to manipulate the array in order to perform these operations. Note that you can sort an array using the

java.util. Arrays.sort(array) method. To sort an arraylist, use the java.util.Collections. sort(arraylist) method. Suppose you want to create an ArrayList for storing integers. Can you use the following code to create a list?

- ArrayList<int> list = new ArrayList<>();
- No. This will not work because the elements stored in an ArrayList must be of an object type. You cannot use a primitive data type such as int to replace a generic type. However, you can create an ArrayList for storing Integer objects as follows:
- ArrayList<Integer> list = new ArrayList<>();

To create an Arraylist from Array:

```
String[] array = {"red", "green", "blue"};

ArrayList<String> list = new ArrayList<>(Arrays.asList(array));
```

To create an array using Arraylist:

```
String[] array1 = new String[list.size()];

 list.toArray(array1);
```

Object Class:

If no inheritance is specified when a class is defined, the superclass of the class is Object by default.

toString() method in the Object class
The signature of the toString() method is:

public String toString()

Invoking toString() on an object returns a string that describes the object. By default, it returns a string consisting of a class name of which the object is an instance, an at sign (@), and the object's memory address in hexadecimal.

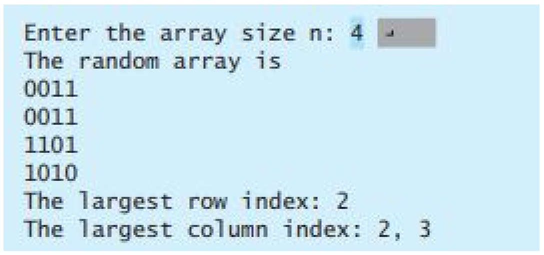
Practice Tasks

Task 1:
Create a SavingAccount class
•Use a static data member annualInterestRate of type double to store interest rate.
•Use a double data member of the class savingBalance.
•Provide a method calculateMonthlyInterest that calculates interest by:
savingBalance = savingBalance + (savingBalance*annualInterestRate)/12

•Provide a method printBalance that will display the saving balance.
•Provide a static method modifyInterestRate that sets the static annualInterestRate to a new value.

Write a driver/client program TestSavingAccount to test class SavingAccount
•Create two objects saver1 and saver2 with the balance of 2000 and 3000 respectively.
•Set the annualInterestRate to 3 percent i.e. 3/100 = 0.03 and calculate monthlyInterest and display the balance of both savers.
•Now, set the annualInterestRate to 4 percent i.e. 4/100 = 0.04 and calculate monthlyInterest and display the balance of both savers.

Task 2
Write a program that randomly fills in 0s and 1s into an n-by-n matrix, prints the matrix, and finds the rows and columns with the most 1s. (Hint: Use two ArrayLists to store the row and column indices with the most 1s.) Here is a sample run of the program:



```
Enter the array size n: 4
The random array is
0011
0011
1101
1010
The largest row index: 2
The largest column index: 2, 3
```

Task 3
Write a program using ArrayList that prompts the user to enter a sequence of numbers and displays the distinct numbers in the sequence. Assume that the input ends with 0 and 0 is not counted as a number in the sequence.

Task 4
Suppose you need to process course information. Each course has a name and has students enrolled. You should be able to add/drop a student to/from the course. Use an ArrayList to store

students. A Course object can be created using the constructor Course(String name) by passing a course name. You can add students to the course using the addStudent(String student) method, drop a student from the course using the dropStudent(String student) method, and return all the students in the course using the getStudents() method. Gives a test class that creates two courses and adds students to them.

Task 5

How do you do the following?

a. Create an ArrayList for storing double values?

b. Append an object to a list?

c. Insert an object at the beginning of a list?

d. Find the number of objects in a list?

e. Remove a given object from a list?

f. Remove the last object from the list?

g. Check whether a given object is in a list?

h. Retrieve an object at a specified index from a list?

Task 5

Spectrum Publishers (SP) is a fast growing publishing house in Pakistan. They were confined to educational books up till now but now they plan to expand their company and include magazines as well. With the expansion in operations, they also wish to go online. You as current OOP students have been hired to design the basic structure for the automation of their tasks. To help you with that, your instructors have provided a few instructions.

●Each Publication is either published or not(true/false). If the publication is not published, display “Not yet Published”.

●Moreover, all Publication have a publicationDate, noOfPages, title and price. Every Publication can be copyrighted, add an additional method of copyright(currentYear, copyrightYear) to give rights to the publication.

●Copyright year must be between 10 years before current year and 1 year after current year, then only a book gets copyrighted.

Note. Make sure that no of pages are greater than 0. There are two special types of Publication, Book and Magazine.

●Each Book additionally has ISBN, author and quantity. A user can also orderBook(noOfBooks) which takes number of books to be ordered from user and multiply by the quantity to calculate the total price, if price exceeds Rs.10000, the customer is eligible for 10% discount. Don’t forget to subtract the number of ordered books from the total quantity of books.

●Each magazine additionally has category, issue and noOfCopies. A user can orderMagazines(noOfMagazines) which takes number of magazines to be ordered from user and multiply by the quantity to calculate the total price, if price exceeds Rs.2500, the customer is eligible for 5% discount. Don’t forget to subtract the number of ordered magazines from the total quantity of magazines.