TP06

Java Class and Objects (part 2)

Remark

1. Overloading methods – Are methods which has the same name but different arguments. Example:

```
public class Compare {
   int max(int a, int b) {
      if (a > b)
         return a;
      else
         return b;
   }

   String max(String a, String b) {
      if (a.compareTo(b) > 0)
         return a;
      else
         return b;
   }
}
```

Method `max()` is called overloading method, because there are 2 methods with the same name `max`.

2. Class Variable – a variable is common to all instance of this class. It means that all object instance of this class share this variable only one address memory. In Java, we use static keyword place in front of variable that we want it to become class variable.

Example:

```
public class User {
   static int user_count = 0;
   String username;
   public User(String username){
      this.username = username;
      user_count++;
   }
}
```

```
public class UserTest {
   public static void main(String[] args) {
      System.out.println("user_count = "+User.user_count);
      User u1 = new User("Dara");
      System.out.println("user_count = "+User.user_count);
      User u2 = new User("Sotha");
      System.out.println("user_count = "+User.user_count);
      User u3 = new User("Sothea");
      System.out.println("user_count = "+User.user_count);
      User u5 = new User("Nida");
      System.out.println("user_count = "+User.user_count);
   }
}
```

Every time, we create a User object, the user_count variable is increased by 1. The variable user_count is created and initialized only one time. Output of program above:

```
user_count = 0
user_count = 1
user_count = 2
user_count = 3
user_count = 4
```

3. Class Method – method that can be called without instantiating object of that class. In Java, we use static keyword in method declaration. Example: Math.abs(int), Math.sqrt(double)
4. Constants – variable that is not changeable at runtime. Example: Math.PI
5. Object Destruction – a special method named `finalize()` will be called automatically before an object is removed from memory.

To facilitate the properties of a class, we draw as graphical representation instead:

<ClassName>
<VisibilitySign><VariableName>:<DataType>=<DefaultValue>
<VisibilitySign><MethodName>(<ParamName1>:<ParamType1>,…):<ReturnType>

Where:

1. <ClassName>: Name of the class
2. <VisibilitySign>: Encapsulation of field. The signs include:

Sign	Keyword	Own class	Same package		Another package	
			Child classes	Other classes	Child classes	Other classes
+	public	✓	✓	✓	✓	✓
-#	protected	✓	✓	✗	✓	✗
~	-	✓	✓	✓	✗	✗
_	private	✓	✓ x	✗	✗	✗

✓ means accessible ✗ means inaccessible

3. <VariableName>: name of variable
4. <MethodName>: name of method
5. <DataType>: can be basic types(int, short, long, char, …) or class types (Point, Student, …)
6. <ParamName1>: name of first parameter
7. <ParamType1>: Datatype of first parameter
8. <ReturnType>: Datatype of value return from the method

Example:

Computer
-serialNumber: String +purchasedDate: Date #isGood: boolean = true ~room: Room
+Computer(serial:String, date:Date, room:Room) +updateStatus(b:boolean):void +checkCondition():Boolean

Dynamic arrays

Dynamic arrays are the array that can grow and shrink upon insertion and deletion of elements in array. We can called it unsized array.

In Java, there are built-in dynamic arrays including LinkedList<E>, ArrayList<E>, Stack<E>, and Vector<E>. The <E> is datatype that the array will store. For example, ArrayList<Integer> is dynamic array that store int values. ArrayList is a class that need to create an instance before use:

```
ArrayList<Integer> arr = new ArrayList<Integer>();
```

or

```
ArrayList<Integer> arr = new ArrayList<>();
```

Some useful methods and fields of ArrayList object:

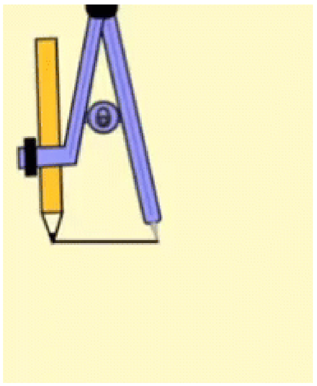
Method	**Description**
arr.add(<value>)	Add <value> to end of array (increase in size of array).
arr.add(<index>,<value>)	Insert <value> at specified <index> (cause other elements shifted to right).
arr.clear()	Remove all elements (size() = 0).
arr.contains(<value>) arr.indexOf(<value>)	Search for <value> in the list.
arr.get(<index>)	Get element at specified <index>.
arr.set(<index>,<value>)	Replace element at <index> with <value>.
arr.remove(<index>) arr.remove(<value>)	Remove an element from the list.
arr.size()	Get size or length of array.

TP06.1. Circle

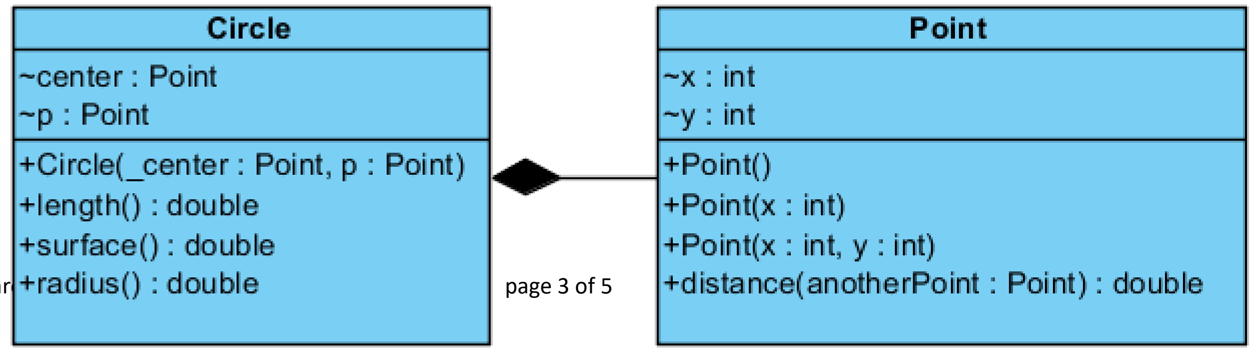
Create class Point represent a point in coordinate system. Point contains attribute x and y as integer (shown in figure below).

Create a class name Circle that have:

- An explicit constructor that 2 points as parameters.
  The first Point is center point,
  and the second point is a point on the circle
- A method to calculate the length of the circle
- A method to calculate the radius of the circle
- A method to calculate the surface of the circle



Then create another class TestCircle as Java application to test all methods in Circle.



TP06.2. Student

Create a Java class represents Student in Department DICE. You can suggest:

- Attributes (student ID and Name are a must)
- Operations/Methods
- Constructor (s)

Then write a program that contains ArrayList<Student>. The program will display a menu:

1. Add new student
2. List students
3. Remove student by name
4. Update student information by id
5. Quit

TP06.3. PC in DICE's Lab

Create a class in Java that represents PC at room 306F in DICE.

Then, write a program in Java that contains an array of all PCs in this ROOM.
The program will show a menu:

1. List all PCs
2. List all damages PCs
3. List all good PCs
4. Mark a PC as damaged
5. Mark a PC as not damaged
6. Quit

TP06.4. Courses

Create a class Course represents all courses in DICE.

Then create a program with a menu:

1. List all courses
2. Find courses by name
3. Add new course
4. Quit

TP06.5. New Year Gift Shop (Object mode)

Create a class name **Product** that represents products to sell in the gift shop.

Then, write a program in Java to help a shop to manage the products to sell to customers. The program will provide a menu to enable admin to:

1. List all products in shop with product number, name, price, and amount in stock
2. Add new product to the list

3. Remove product from list by index
4. Update product in list
5. Exit program