Yanbu University College
BACHELOR OF SCIENCE IN
Applied Computer Science (ACS)
Introduction to Computer Science (CS 102)
Third Semester – Academic Year 2011 – 2012

Lab Exercise – 6

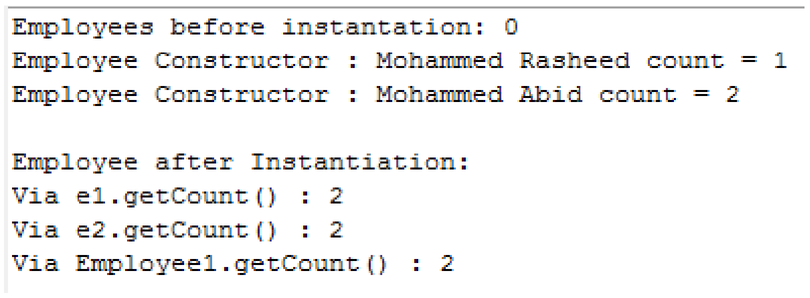
Course Instructor: Mohammed Rafi / Rasheed Ricardo

1. Write a class Country with 3 instance variables. country_name , capital_City , president_name are of type strings.
    a. Provide get and set methods for each of the instance varaibales.
    b. Write a toString() method to print the object in the following format:

President	Country	Capital
ABCCC DDDDD	YZ	SSS

    Write a test application named CountryApp which performs the following tasks.
    a. Create an object the class Country
    b. Read couny name, president and country capital.
    c. print the above given format.

2. Tracking the numbers of Employee objects that have been created, for the given class.
    Create a class Employee which three instance variables, firstName (type string), lastName (type string) and count (type static int).
    Provide a constructor that initializes two instance variables fistName and lastName.
    Provide get and set methods for the Instance variables.
    Write a test application named EmployeeTest that demonstrates class Employee's capabilities. Create two Employee objects displays Employee name and numbers of objects it has created.
    Output:



```
Employees before instantation: 0
Employee Constructor : Mohammed Rasheed count = 1
Employee Constructor : Mohammed Abid count = 2

Employee after Instantiation:
Via e1.getCount() : 2
Via e2.getCount() : 2
Via Employee1.getCount() : 2
```

3. Create a class Author which as three instance variables – name (type string), email (type string) and gender (type char) which performs the following tasks.
    a. Provide a constructor that initializes the three instance variables.
    b. Provide set and get methods for the instance variables.
    c. Provide toString method which the returns the string in the specified format as shown:

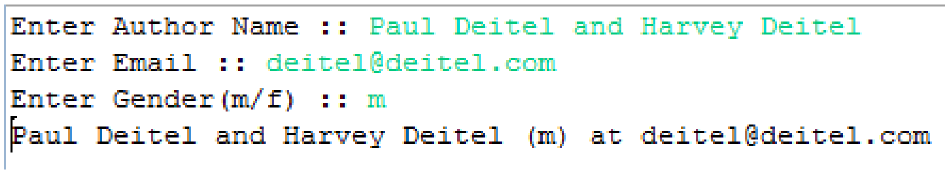
    Paul Deitel and Harvey Deitel (m) at deitel@deitel.com

    Write a test application AuthorTest which performs the following tasks.
    a. Take the input from user for authorName, email and gender.

b. Create an object for the Author class.

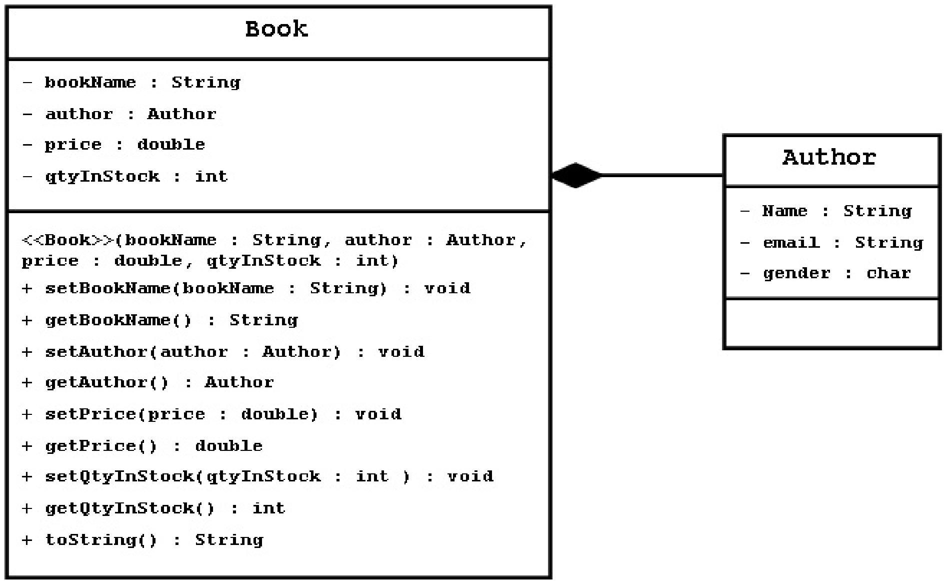
c. Display the information.

Sample Output:



```
Enter Author Name :: Paul Deitel and Harvey Deitel
Enter Email :: deitel@deitel.com
Enter Gender(m/f) :: m
Paul Deitel and Harvey Deitel (m) at deitel@deitel.com
```

4.



```
Book
- bookName : String
- author : Author
- price : double
- qtyInStock : int
<<Book>>(bookName : String, author : Author,
price : double, qtyInStock : int)
+ setBookName(bookName : String) : void
+ getBookName() : String
+ setAuthor(author : Author) : void
+ getAuthor() : Author
+ setPrice(price : double) : void
+ getPrice() : double
+ setQtyInStock(qtyInStock : int ) : void
+ getQtyInStock() : int
+ toString() : String

Author
- Name : String
- email : String
- gender : char
```

Use the above UML, to write java application for the book which as composition of Author class. Write a test application BookTest which performs the following tasks.

a. Initialize the book object with the following values as given

bookName = “Java How to Program”

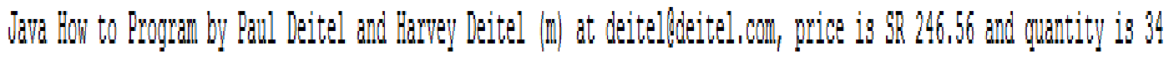
author = Instance of the object Author

price = 246.56

qtyInStock = 10

b. Display the information as shown in the output.

Sample Ouput:



```
Java How to Program by Paul Deitel and Harvey Deitel (m) at deitel@deitel.com, price is SR 246.56 and quantity is 34
```

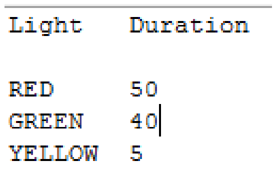
5. Write a enum type TrafficLight, whose constants (RED, GREEN YELLOW) provide instance variable – duration of type int and modifier final.

a. Provide a constructor which take one parameter – the duration of light and initialize the instance variable.

b. Provide the get method for the instance variable which returns the duration of light.

Write a program to test the TrafficLight enum so that it displays the enum constants and their durations.

Sample Output:



```
Light    Duration

RED      50
GREEN    40
YELLOW   5
```

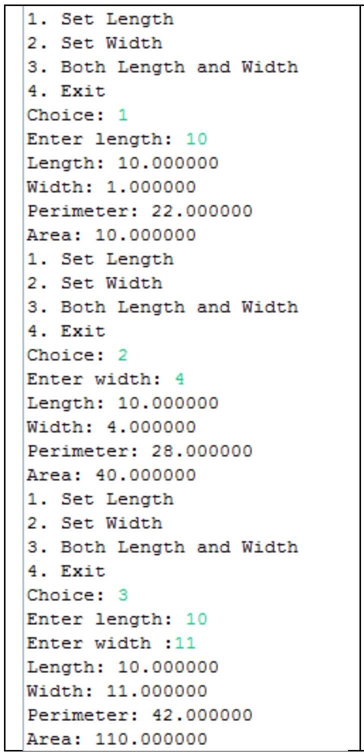
6. Create a class Rectangle with two instance variables – length of type double and width of type double.
   a. Provide a parameterless constructor to the instance variables and set the values to 1.0
   b. Provide a Parameterized constructor to the instance variables. Using overloaded and chaining constructor.
   c. Provide set and get methods for each of the instance of variable. The set method should verify that length and width each larger than 0.0 and less than 20.0 otherwise set to 1.
   d. Provide methods calculate the rectangle's area and perimeter. It returns the calculated area and perimeter using toString method.

   Write a test application named RectangleApp which perform the following tasks.
   a. Read the length and width from the user.
   b. Pass the values to using object
   c. And display the area and parameter.

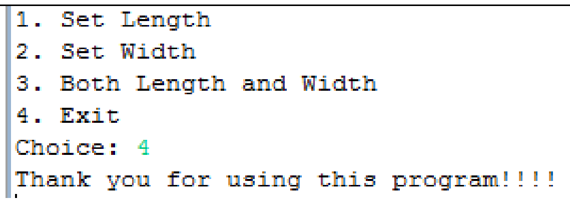
   Note: **Area = Length * width**

   Perimeter = 4 * Length

Sample Output:



```
1. Set Length
2. Set Width
3. Both Length and Width
4. Exit
Choice: 1
Enter length: 10
Length: 10.000000
Width: 1.000000
Perimeter: 22.000000
Area: 10.000000
1. Set Length
2. Set Width
3. Both Length and Width
4. Exit
Choice: 2
Enter width: 4
Length: 10.000000
Width: 4.000000
Perimeter: 28.000000
Area: 40.000000
1. Set Length
2. Set Width
3. Both Length and Width
4. Exit
Choice: 3
Enter length: 10
Enter width :11
Length: 10.000000
Width: 11.000000
Perimeter: 42.000000
Area: 110.000000
```



```
1. Set Length
2. Set Width
3. Both Length and Width
4. Exit
Choice: 4
Thank you for using this program!!!!
```

7. A Class called MyPoint, which models a 2D point with x and y coordinates, is designed as shown in the class diagram.

```
MyPoint
-x:int = 0
-y:int = 0
+MyPoint()
+MyPoint(x:int, y:int)
+getX():int
+setX(x:int):void
+getY():int
+setY(y:int):void
+setXY(x:int, y:int):void
+toString():String
+distance(x:int, y:int):double
+distance(another:MyPoint):double
```

It contains

a. Two instance variables x (int) and y (int).

b. A "no-argument" (or "no-arg") constructor that construct a point at (0, 0).

c. A constructor that constructs a point with the given x and y coordinates.

d. Getter and setter for the instance variables x and y.

e. A method setXY() to set both x and y.

f. A toString() method that returns a string description of the instance in the format "(x, y)".

g. A method called distance(int x, int y) that returns the distance from this point to another point at the given (x, y) coordinates.

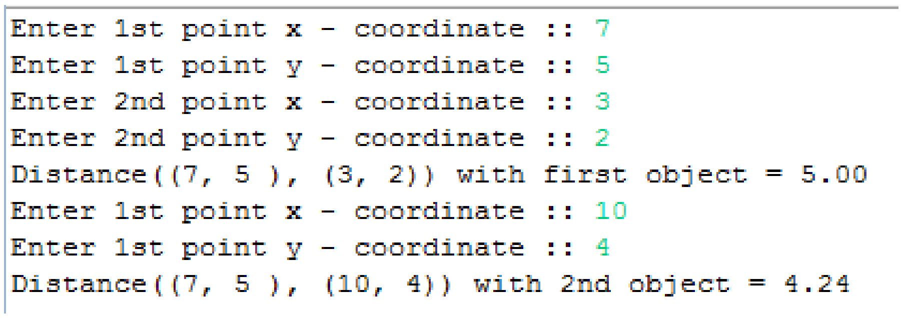
h. An overloaded distance(MyPoint another) that returns the distance from this point to the given MyPoint instance another.

Write a test application MyPointTest and perform the followings.

a. Create two objects for the class MyPoint.

b. Test the overloaded distance methods and display the distance between the two points..

Sample Output:



```
Enter 1st point x - coordinate :: 7
Enter 1st point y - coordinate :: 5
Enter 2nd point x - coordinate :: 3
Enter 2nd point y - coordinate :: 2
Distance((7, 5 ), (3, 2)) with first object = 5.00
Enter 1st point x - coordinate :: 10
Enter 1st point y - coordinate :: 4
Distance((7, 5 ), (10, 4)) with 2nd object = 4.24
```

8. Create a class called Date which has three instance variables – month (type integer), day (type integer) and year (type integer).

a. The constructor receives the three parameters, invokes the utility method checkMonth to validate the month if the month is out of range set to 1 and invokes another method checkDay to validate the value of the day based on the current month and year.

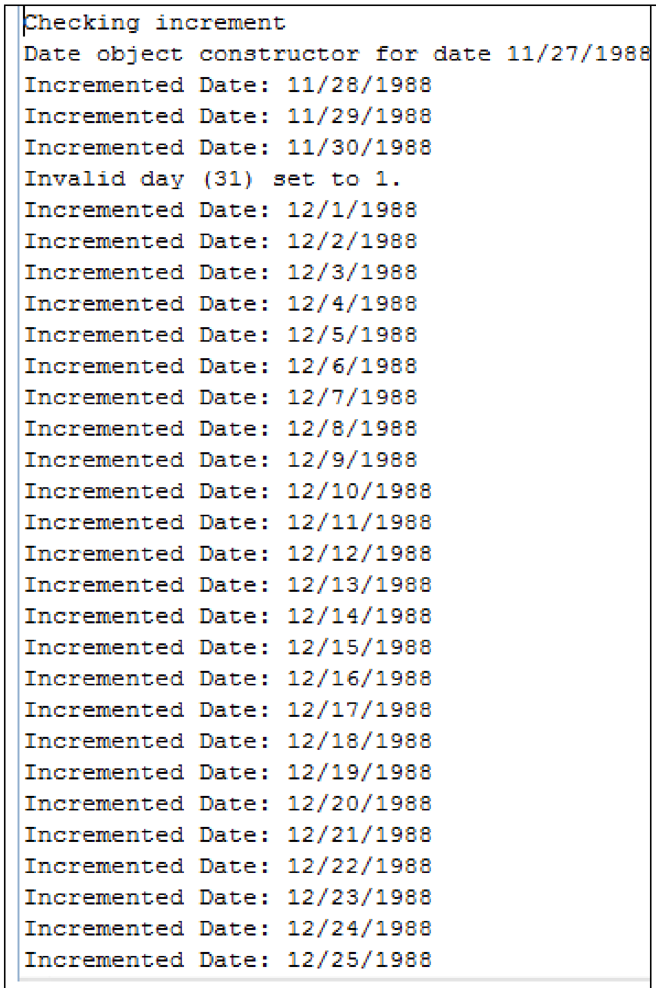
b. Provide a method nextDay to increment the day by one.

Write a test application named DateApp which test the method nextDay in a loop that prints the date during each iteration to illustrate that the method works correctly. Test the following cases.

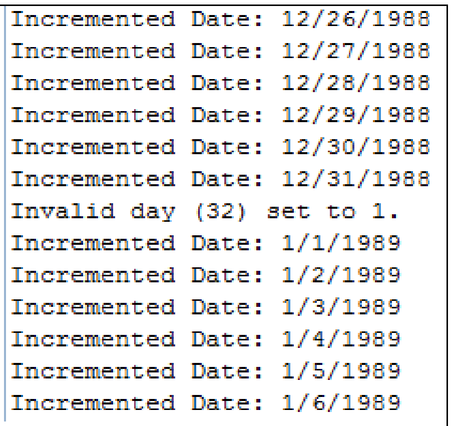
a. Incrementing into the next month.

b. Incrementing into the next year.

Sample Output:



```
Checking increment
Date object constructor for date 11/27/1988
Incremented Date: 11/28/1988
Incremented Date: 11/29/1988
Incremented Date: 11/30/1988
Invalid day (31) set to 1.
Incremented Date: 12/1/1988
Incremented Date: 12/2/1988
Incremented Date: 12/3/1988
Incremented Date: 12/4/1988
Incremented Date: 12/5/1988
Incremented Date: 12/6/1988
Incremented Date: 12/7/1988
Incremented Date: 12/8/1988
Incremented Date: 12/9/1988
Incremented Date: 12/10/1988
Incremented Date: 12/11/1988
Incremented Date: 12/12/1988
Incremented Date: 12/13/1988
Incremented Date: 12/14/1988
Incremented Date: 12/15/1988
Incremented Date: 12/16/1988
Incremented Date: 12/17/1988
Incremented Date: 12/18/1988
Incremented Date: 12/19/1988
Incremented Date: 12/20/1988
Incremented Date: 12/21/1988
Incremented Date: 12/22/1988
Incremented Date: 12/23/1988
Incremented Date: 12/24/1988
Incremented Date: 12/25/1988
```



```
Incremented Date: 12/26/1988
Incremented Date: 12/27/1988
Incremented Date: 12/28/1988
Incremented Date: 12/29/1988
Incremented Date: 12/30/1988
Incremented Date: 12/31/1988
Invalid day (32) set to 1.
Incremented Date: 1/1/1989
Incremented Date: 1/2/1989
Incremented Date: 1/3/1989
Incremented Date: 1/4/1989
Incremented Date: 1/5/1989
Incremented Date: 1/6/1989
```