# Lab Report 01

Course Name: Object Oriented Programming

Experiment: Lab01 Introduction to Java

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to use Eclipse IDE

1.2 to learn the differences between syntax errors, runtime errors, and logic errors.

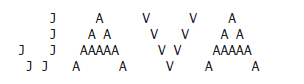
1.3 to learn how to write simple Java programs.

**2. Experimental Contents**

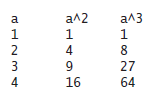
2.1 (Display three messages) Write a program that displays Welcome to Java, Welcome to Computer Science, and Programming is fun.

2.2 (Display five messages) Write a program that displays Welcome to Java five times.

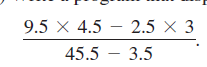
2.3 (Display a pattern) Write a program that displays the following pattern:



2.4 (Print a table) Write a program that displays the following table:



2.5 (Compute expressions) Write a program that displays the result of



2.6 (Approximate ) p can be computed using the following formula:



Write a program that displays the result of



and. Use 1.0 instead of 1 in your program.

2.7 (Area and perimeter of a circle) Write a program that displays the area and perimeter of a circle that has a radius of 5.5 using the following formula:

perimeter = 2 \* radius \* 

area = radius \* radius \* 

2.8 (Area and perimeter of a rectangle) Write a program that displays the area and perimeter of a rectangle with the width of 4.5 and height of 7.9 using the following formula:

area = width \* height

2.9 (Average speed in miles) Assume a runner runs 14 kilometers in 45 minutes and 30 seconds. Write a program that displays the average speed in miles per hour. (Note that 1 mile is 1.6 kilometers.)

2.10 (Population projection) The U.S. Census Bureau projects population based on the following assumptions:

■ One birth every 7 seconds

■ One death every 13 seconds

■ One new immigrant every 45 seconds

Write a program to display the population for each of the next five years. Assume the current population is 312,032,486 and one year has 365 days. Hint: In Java, if two integers perform division, the result is an integer. The fractional part is truncated. For example, 5 / 4 is 1 (not 1.25) and 10 / 4 is 2 (not 2.5). To get an accurate result with the fractional part, one of the values involved in the division must be a number with a decimal point. For example, 5.0 / 4 is 1.25 and 10 / 4.0 is 2.5.

**3. Please show your questions analysis, code and results.**

# Lab Report 02

Course Name: Object Oriented Programming

Experiment: Lab02 Elementary Programming

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to input data from console;

1.2 to understand variables, constants, methods, and classes;

1.3 to learn java data types: byte, short, int, long, float, double,and their literals;

1.4 to learn type cast;

1.5 to learn how to use java API.

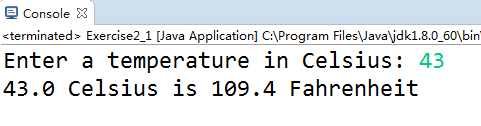
**2. Experimental Contents**

2.1 (Convert Celsius to Fahrenheit) Write a program that reads a Celsius degree in a double value from the console, then converts it to Fahrenheit and displays the result. The formula for the conversion is as follows:

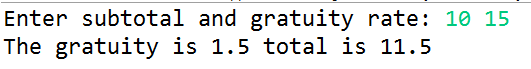
fahrenheit = (9 / 5) \* celsius + 32

Hint: In Java, 9 / 5 is 1, but 9.0 / 5 is 1.8.

Here is a sample run:



2.2 (Financial application: calculate tips) Write a program that reads the subtotal and the gratuity rate, then computes the gratuity and total. For example, if the user enters 10 for subtotal and 15% for gratuity rate, the program displays $1.5 as gratuity and $11.5 as total. Here is a sample run:



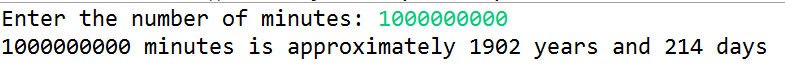
2.3 (Sum the digits in an integer) Write a program that reads an integer between 0 and 1000 and adds all the digits in the integer. For example, if an integer is 932, the sum of all its digits is 14.

Hint: Use the % operator to extract digits, and use the / operator to remove the extracted digit. For instance, 932 % 10 = 2 and 932 / 10 = 93.

Here is a sample run:



2.4 (Find the number of years) Write a program that prompts the user to enter the minutes (e.g., 1 billion), and displays the number of years and days for the minutes. For simplicity, assume a year has 365 days. Here is a sample run:



2.5 (Physics: acceleration) Average acceleration is defined as the change of velocity divided by the time taken to make the change, as shown in the following formula:



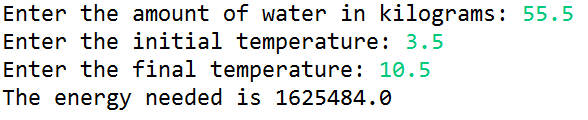
Write a program that prompts the user to enter the starting velocity v0 in meters/second, the ending velocity v1 in meters/second, and the time span t in seconds, and displays the average acceleration. Here is a sample run:



2.6 (Science: calculating energy) Write a program that calculates the energy needed to heat water from an initial temperature to a final temperature. Your program should prompt the user to enter the amount of water in kilograms and the initial and final temperatures of the water. The formula to compute the energy is

Q = M \* (finalTemperature – initialTemperature) \* 4184

where M is the weight of water in kilograms, temperatures are in degrees Celsius, and energy Q is measured in joules. Here is a sample run:



2.7 (Financial application: compound value) Suppose you save $100 each month into a savings account with the annual interest rate 5%. Thus, the monthly interest rate is 0.05/12 = 0.00417. After the first month, the value in the account becomes

100 \* (1 + 0.00417) = 100.417

After the second month, the value in the account becomes

(100 + 100.417) \* (1 + 0.00417) = 201.252

After the third month, the value in the account becomes

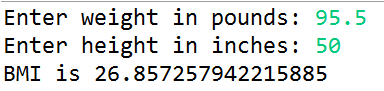
(100 + 201.252) \* (1 + 0.00417) = 302.507

and so on.

Write a program that prompts the user to enter a monthly saving amount and displays the account value after the sixth month.



2.8 (Health application: computing BMI) Body Mass Index (BMI) is a measure of health on weight. It can be calculated by taking your weight in kilograms and dividing by the square of your height in meters. Write a program that prompts the user to enter a weight in pounds and height in inches and displays the BMI. Note that one pound is 0.45359237 kilograms and one inch is 0.0254 meters. Here is a sample run:

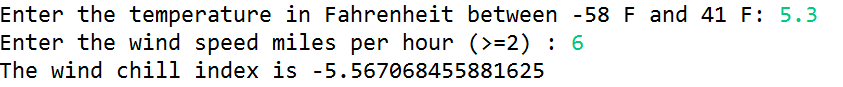


2.9 (Science: wind-chill temperature) How cold is it outside? The temperature alone is not enough to provide the answer. Other factors including wind speed, relative humidity, and sunshine play important roles in determining coldness outside. In 2001, the National Weather Service (NWS) implemented the new wind-chill temperature to measure the coldness using temperature and wind speed. The formula is

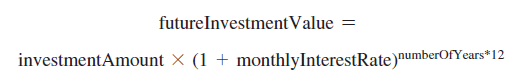


where ta is the outside temperature measured in degrees Fahrenheit and v is the speed measured in miles per hour. twc is the wind-chill temperature. The formula cannot be used for wind speeds below 2 mph or temperatures below -58 ºF or above 41ºF.

Write a program that prompts the user to enter a temperature between -58 ºF and 41ºF and a wind speed greater than or equal to 2 and displays the wind-chill temperature. Use Math.pow(a, b) to compute v0.16. Here is a sample run:

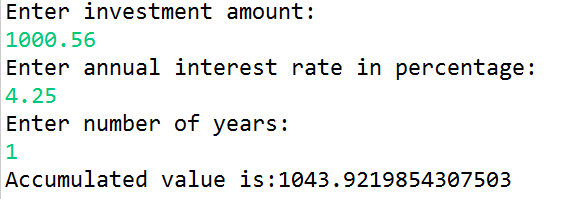


2.10 (Financial application: calculate future investment value) Write a program that reads in investment amount, annual interest rate, and number of years, and displays the future investment value using the following formula:



For example, if you enter amount 1000, annual interest rate 3.25%, and number of years 1, the future investment value is 1032.98.

Here is a sample run:



**3. Please show your questions analysis, code and results.**

# Lab Report 03

Course Name: Object Oriented Programming

Experiment: Lab03 Selections&Loops&Characters&Strings

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

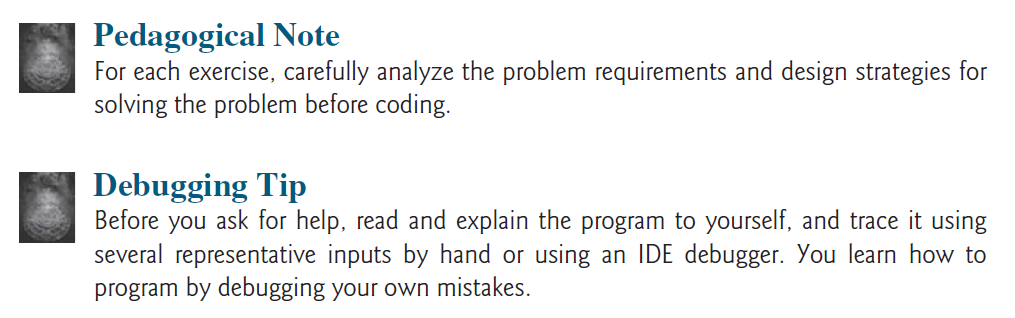
**1. Experimental Purposes and Requirements**

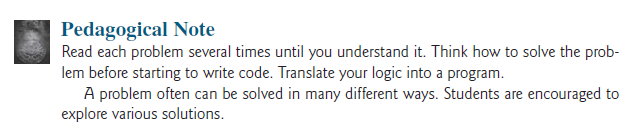
1.1 to learn how to selections;

1.2 to learn how to loops;

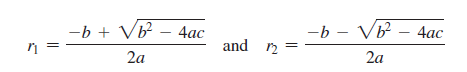
1.3 to learn java API library:Math, String, and Character

**2. Experimental Contents**





3.1 (Algebra: solve quadratic equations) The two roots of a quadratic equation ax2 + bx + c = 0 can be obtained using the following formula:



b2 - 4ac is called the discriminant of the quadratic equation. If it is positive, the equation has two real roots. If it is zero, the equation has one root. If it is negative, the equation has no real roots.

Write a program that prompts the user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is positive, display two roots. If the discriminant is 0, display one root. Otherwise, display “The equation has no real roots”.

Hint 1: you can use Math.pow(x, 0.5) to compute.

Hint 2: you can use the flowing selection syntax.

**if** (discriminant < 0) {

}

**else** **if** (discriminant == 0) {

}

**else** {

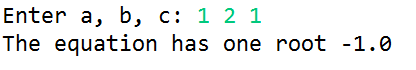
}

Here are some sample runs.

Test example 1:



Test example 2:



Test example 3:



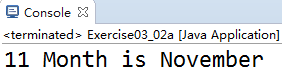
3.2 (Random month) Write a program that randomly generates an integer between 1 and 12 and displays the English month name January, February, …, December for the number 1, 2, …, 12, accordingly.

Hint 1: you can use Math.random() to generate a random.

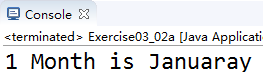
Hint 2: you can use the switch selection syntax.

Here are some sample runs.

Test example 1:



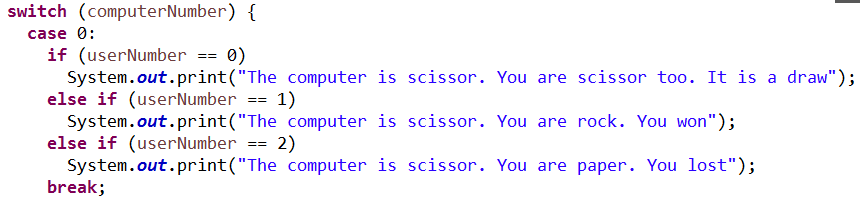
Test example 2:



3.3 (Game: scissor, rock, paper) Write a program that plays the popular scissor-rock-paper game. (A scissor can cut a paper, a rock can knock a scissor, and a paper can wrap a rock.) The program randomly generates a number 0, 1, or 2 representing scissor, rock, and paper. The program prompts the user to enter a number 0, 1, or 2 and displays a message indicating whether the user or the computer wins, loses, or draws.

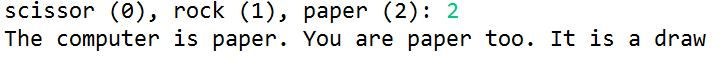
Hint 1: you can use Math.random() to generate a random.

Hint 2: you can use nested switch and if selection syntax like the following code.



Here are sample runs:

Test example 1:



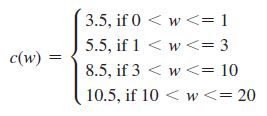
Test example 2:



Test example 3:



3.4 (Cost of shipping) A shipping company uses the following function to calculate the cost (in dollars) of shipping based on the weight of the package (in pounds).



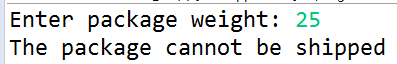
Write a program that prompts the user to enter the weight of the package and display the shipping cost. If the weight is greater than 20, display a message “the package cannot be shipped.”

Here are sample runs:

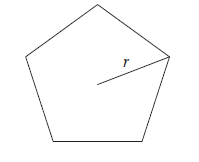
Test example 1:



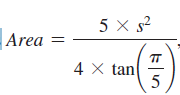
Test example 2:



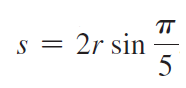
3.5 (Geometry: area of a pentagon) Write a program that prompts the user to enter the length from the center of a pentagon to a vertex and computes the area of the pentagon, as shown in the following figure.



The formula for computing the area of a pentagon is



where s is the length of a side. The side can be computed using the formula:



where r is the length from the center of a pentagon to a vertex. Round up two digits after the decimal point.

Hint:you can use the methods of Math class.

Here is a sample run:



3.6 (Find the character of an ASCII code) Write a program that receives an ASCII code (an integer between 0 and 127) and displays its character. Here is a sample run:

Hint: the conversion between integer number and character.



3.7 (Student major and status) Write a program that prompts the user to enter two characters and displays the major and status represented in the characters. The first character indicates the major and the second is number character 1, 2, 3, 4, which indicates whether a student is a freshman, sophomore, junior, or senior. Suppose the following characters are used to denote the majors:

M: Mathematics

C: Computer Science

I: Information Technology

Hint1: String s = input.nextLine();

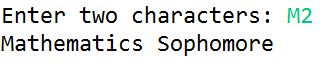
Hint2: s.charAt(0)/ s.charAt(1)

Here is a sample run:

Test example 1:

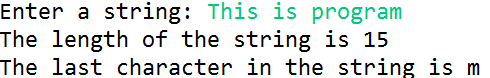


Test example 2:



3.8 (Process a string) Write a program that prompts the user to enter a string and displays its length and its last character.

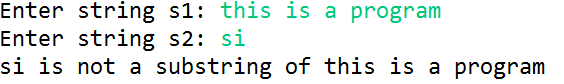
Hint: you can use method s.length() to get the length of a string.



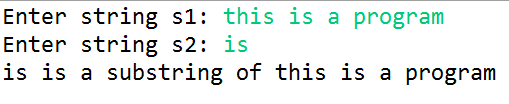
3.9 (Check substring) Write a program that prompts the user to enter two strings and reports whether the second string is a substring of the first string.

Hint: you can use method s1.indexof(s2).

Test example 1:

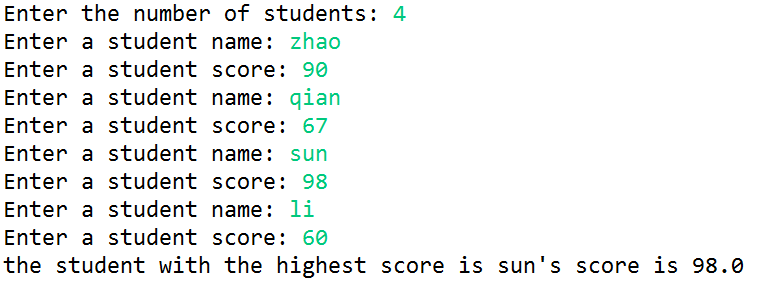


Test example 2:



3.10 (Find the highest score) Write a program that prompts the user to enter the number of students and each student’s name and score, and finally displays the name of the student with the highest score.

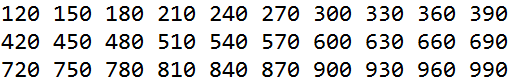
Test example:



3.11 (Find numbers divisible by 5 and 6) Write a program that displays all the numbers from 100 to 1,000, ten per line, that are divisible by 5 and 6. Numbers are separated by exactly one space.

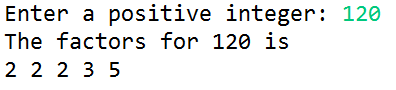
Hint: you can use for loop.

Test example:

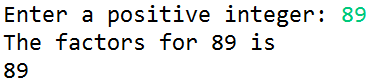


3.12 (Find the factors of an integer) Write a program that reads an integer and displays all its smallest factors in increasing order. For example, if the input integer is 120, the output should be as follows: 2, 2, 2, 3, 5.

Test example 1:



Test example 2:



**3. Please show your questions analysis, code and results.**

# Lab Report 04

Course Name: Object Oriented Programming

Experiment: Lab04 Methods

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to define and use methods;

1.2 to develop reusable code that is modular, easy to read, easy to debug, and easy to maintain.

**2. Experimental Contents**

4.1 (Sum the digits in an integer) Write a method that computes the sum of the digits in an integer. Use the following method header:

public static int sumDigits(long n)

For example, sumDigits(234) returns 9 (2 + 3 + 4). Write a test program that prompts the user to enter an integer and displays the sum of all its digits.

Hint 1: Use the % operator to extract digits, and the / operator to remove the extracted digit. For instance, to extract 4 from 234, use 234 % 10 (= 4). To remove 4 from 234, use 234 / 10 (= 23).

Hint 2: Use a loop to repeatedly extract and remove the digit until all the digits are extracted.

Test example 1:



Test example 2:



4.2 (Palindrome integer) Write the methods with the following headers

// Return the reversal of an integer, i.e., reverse(456) returns 654

public static int reverse(int number)

// Return true if number is a palindrome

public static boolean isPalindrome(int number)

Use the reverse method to implement isPalindrome. A number is a palindrome if its reversal is the same as itself. Write a test program that prompts the user to enter an integer and reports whether the integer is a palindrome.

Hint: the invocation relation between methods main, isPalindrome and reverse is that main invoke isPalindrome and isPalindrome invoke reverse.

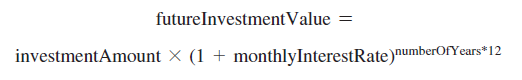
Test example 1:



Test example 1:



4.3 (Financial application: compute the future investment value) Write a method that computes future investment value at a given interest rate for a specified number of years. The future investment is determined using following the formula:

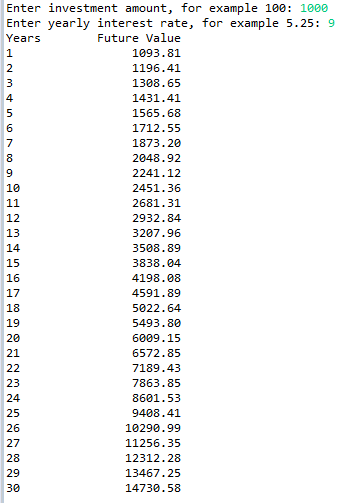


Use the following method header:

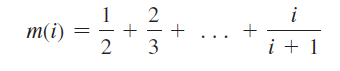
public static double futureInvestmentValue(double investmentAmount, double monthlyInterestRate, int years)

For example, futureInvestmentValue(10000, 0.05/12, 5) returns 12833.59.

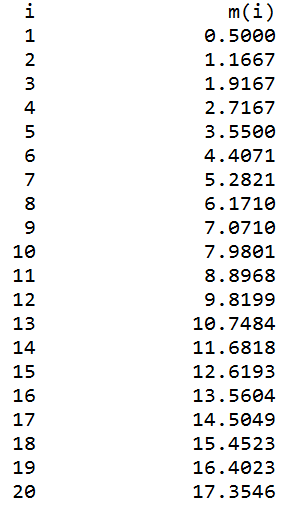
Write a test program that prompts the user to enter the investment amount (e.g., 1000) and the interest rate (e.g., 9%) and prints a table that displays future value for the years from 1 to 30, as shown below: (Hint: you can use System.out.printf() to display the results.)



4.4 (Sum series) Write a method to compute the following series:

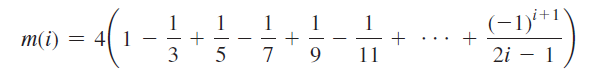


Write a test program that displays the following table:

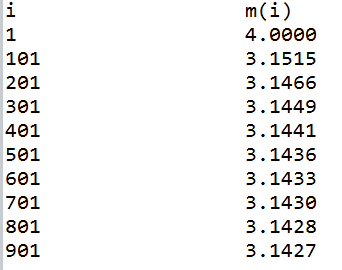


Attention: About the following Exercises 4.5 and 4.6, you can choose one of them to complete, or you can do both.

4.5 (Estimate )  can be computed using the following series:



Write a method that returns m(i) for a given i and write a test program that displays the following table:



4.6 (Check password) Some websites impose certain rules for passwords. Write a method that checks whether a string is a valid password. Suppose the password rules are as follows:

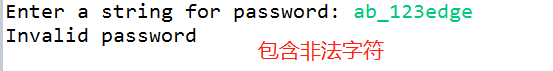
■ A password must have at least eight characters.

■ A password consists of only letters and digits.

■ A password must contain at least two digits.

Write a program that prompts the user to enter a password and displays Valid Password if the rules are followed or Invalid Password otherwise.(Hint: you can use Character class methods.)





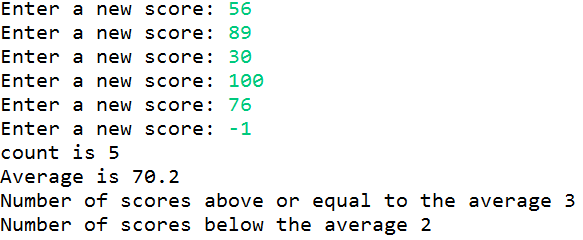




4.7 (Analyze scores) Write a program that reads an unspecified number of scores and determines how many scores are above or equal to the average and how many scores are below the average. Enter a negative number to signify the end of the input. Assume that the maximum number of scores is 100.

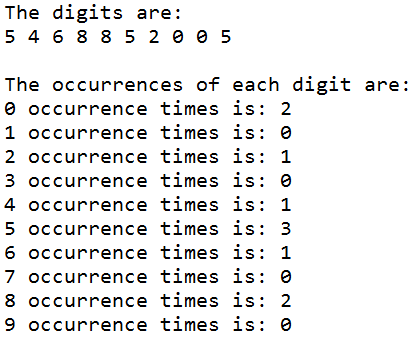
Hint 1: you can use do-while or while loop, please pay attention to the loop continuation condition

Hint 2: the scores should be stored into an array.



4.8 (Count single digits) Write a program that generates 10 random integers between 0 and 9 and displays the count for each number.

Hint 1: you can refer to LISTING 7.4 CountLettersInArray.java



4.9 (Average an array) Write two overloaded methods that return the average of an array with the following headers:

public static int average(int[] array)

public static double average(double[] array)

Write a test program that prompts the user to enter ten double values, invokes this method, and displays the average value.

Hint: you can refer to LISTING 7.6 LinearSearch.java, not to refer to search algorithm, but to refer to Passing Arrays to Methods.

Test example:



4.10 Write two overloaded sort methods that can sort an array elements from large to small with the following headers:

public static void selectionSort(int[] list)

public static void selectionSort(double[] list)

Write a test program that reads in ten double numbers, invokes the method, and displays the sorted numbers.

Hint: you can refer to LISTING 7.8 SelectionSort.java.



**3. Please show your questions analysis, code and results.**

# Lab Report 05

Course Name: Object Oriented Programming

Experiment: Lab05 Objects and Classes

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to describe objects and classes, and use classes to model objects;

1.2 to learn how to access objects;

1.3 to distinguish between object reference variables and primitive data type variables.

**2. Experimental Contents**

5.1 (The Rectangle class) Following the example of the Circle class in Section 9.2(P323 Figure 9.3), design a class named Rectangle to represent a rectangle. The class contains:

■ Two double data fields named width and height that specify the width and height of the rectangle. The default values are 1 for both width and height.

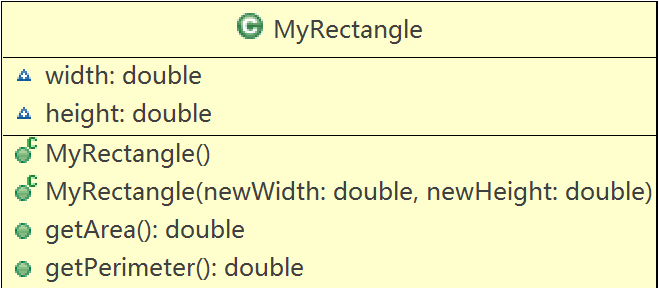
■ A no-arg constructor that creates a default rectangle.

■ A constructor that creates a rectangle with the specified width and height.

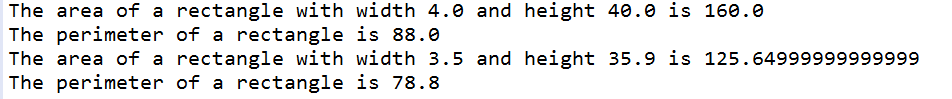
■ A method named getArea() that returns the area of this rectangle.

■ A method named getPerimeter() that returns the perimeter.

Draw the UML(You are allowed not to draw UML, but if you want, you can!) diagram for the class and then implement the class. Write a test program that creates two Rectangle objects—one with width 4 and height 40 and the other with width 3.5 and height 35.9. Display the width, height, area, and perimeter of each rectangle in this order.



Test Example:



5.2 (The Stock class) Following the example of the Circle class in Section 9.2(P323 Figure 9.3), design a class named Stock that contains:

■ A string data field named symbol for the stock’s symbol.

■ A string data field named name for the stock’s name.

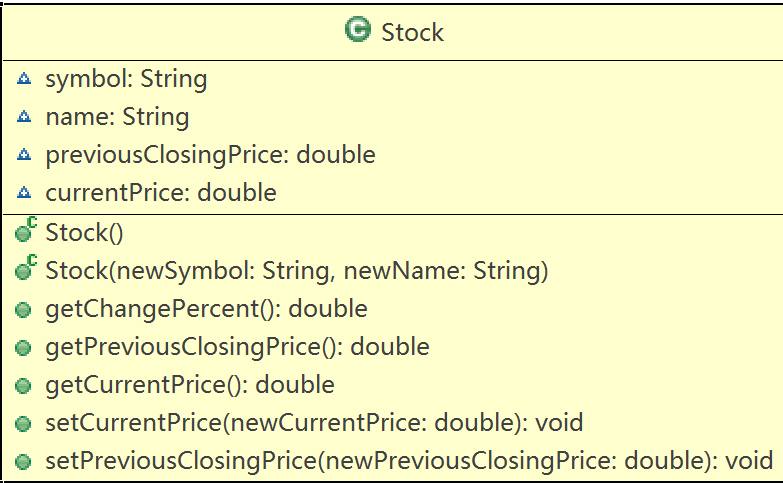
■ A double data field named previousClosingPrice that stores the stock price for the previous day.

■ A double data field named currentPrice that stores the stock price for the current time.

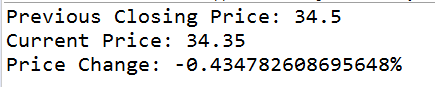
■ A constructor that creates a stock with the specified symbol and name.

■ A method named getChangePercent() that returns the percentage changed from previousClosingPrice to currentPrice.

Draw the UML diagram for the class and then implement the class. Write a test program that creates a Stock object with the stock symbol ORCL, the name Oracle Corporation, and the previous closing price of 34.5. Set a new current price to 34.35 and display the price-change percentage.



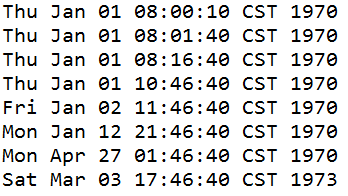
Test Example:



5.3 (Use the Date class) Write a program that creates a Date object, sets its elapsed time to 10000, 100000, 1000000, 10000000, 100000000, 1000000000, 10000000000, and 100000000000, and displays the date and time using the toString() method, respectively.

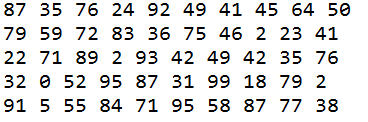
Hint: you can use method setTime of Date class to set the elapsed time, if you do not know how to use the API, please read the online or offline reference document.

( <https://docs.oracle.com/javase/9/docs/api/overview-summary.html>)



5.4 (Use the Random class) Write a program that creates a Random object with seed 1000 and displays the first 50 random integers between 0 and 100 using the nextInt(100) method.

Test Example:



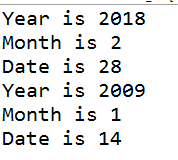
5.5 (Use the GregorianCalendar class) Java API has the GregorianCalendar class in the java.util package, which you can use to obtain the year, month, and day of a date. The no-arg constructor constructs an instance for the current date, and the methods get(GregorianCalendar.YEAR), get(GregorianCalendar.MONTH), and get(GregorianCalendar.DAY\_OF\_MONTH) return the year, month, and day. Write a program to perform two tasks:

■ Display the current year, month, and day.

■ The GregorianCalendar class has the setTimeInMillis(long), which can be used to set a specified elapsed time since January 1, 1970. Set the value to 1234567898765L and display the year, month, and day.

Note: The first month of the year in the Gregorian and Julian calendars is JANUARY which is 0; the last depends on the number of months in a year.

Test Example:



**3. Review questions**

5.6 What is wrong with each of the following programs?

1 public class ShowErrors {

2 public static void main(String[] args) {

3 ShowErrors t = new ShowErrors(5);

4 }

5 }

(a)

1 public class ShowErrors {

2 public static void main(String[] args) {

3 ShowErrors t = new ShowErrors();

4 t.x();

5 }

6 }

(b)

1 public class ShowErrors {

2 public void method1() {

3 Circle c;

4 System.out.println("What is radius "

5 + c.getRadius());

6 c = new Circle();

7 }

8 }

(c)

1 public class ShowErrors {

2 public static void main(String[] args) {

3 C c = new C(5.0);

4 System.out.println(c.value);

5 }

6 }

7

8 class C {

9 int value = 2;

10 }

(d)

5.7 What is wrong in the following code?

1 class Test {

2 public static void main(String[] args) {

3 A a = new A();

4 a.print();

5 }

6 }

7

8 class A {

9 String s;

10

11 A(String newS) {

12 s = newS;

13 }

14

15 public void print() {

16 System.out.print(s);

17 }

18 }

5.8 What is the output of the following code?

public class A {

boolean x;

public static void main(String[] args) {

A a = new A();

System.out.println(a.x);

}

}

**4. Please show your questions analysis, code and results.**

# Lab Report 06

Course Name: Object Oriented Programming

Experiment: Lab06 Objects and Classes

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to encapsulate data fields to make classes easy to maintain;

1.2 to learn some important concepts: object-type argument, immutable objects and classes, this and encapsulation.

**2. Experimental Contents**

6.1 (Stopwatch) Design a class named StopWatch. The class contains:

■ Private data fields startTime and endTime with getter methods.

■ A no-arg constructor that initializes startTime with the current time.

■ A method named start() that resets the startTime to the current time.

■ A method named stop() that sets the endTime to the current time.

■ A method named getElapsedTime() that returns the elapsed time for the stopwatch in milliseconds.

Draw the UML diagram for the class and then implement the class. Write a test program that measures the execution time of sorting 100,000 numbers using selection sort.

Note: you can compare two sort methods to observe which is better. One is your own selection sort method, the other is java.util.Arrays.sort.

Test example 1:



Test example 2:



**package** test;

**import** java.util.Arrays;

**import** java.util.\*;

**public** **class** StopWatch {

**public** **static** **void** main(String[] args){

**int**[] num = **new** **int**[100000];

StopWatch x = **new** StopWatch();

**for**(**int** i = 0; i < 100000; i++) {

num[i] = (**int**)(Math.*random*()\*100000);

}

x.start();

Arrays.*sort*(num);

x.end();

System.***out***.print("The sort time is " + x.getElapsedTime());

}

**private** **long** startTime;

**private** **long** endTime;

**void** start(){

startTime = System.*currentTimeMillis*();

}

**void** end(){

endTime = System.*currentTimeMillis*();

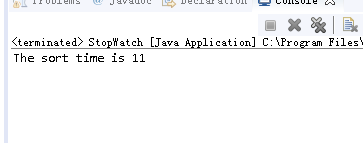
}

**long** getElapsedTime(){

**return** endTime - startTime;

}

}



6.2 (The Account class) Design a class named Account that contains:

■ A private int data field named id for the account (default 0).

■ A private double data field named balance for the account (default 0).

■ A private double data field named annualInterestRate that stores the current interest rate (default 0). Assume all accounts have the same interest rate.

■ A private Date data field named dateCreated that stores the date when the account was created.

■ A no-arg constructor that creates a default account.

■ A constructor that creates an account with the specified id and initial balance.

■ The accessor and mutator methods for id, balance, and annualInterestRate.

■ The accessor method for dateCreated.

■ A method named getMonthlyInterestRate() that returns the monthly interest rate.

■ A method named getMonthlyInterest() that returns the monthly interest.

■ A method named withdraw that withdraws a specified amount from the account.

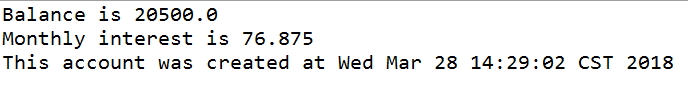
■ A method named deposit that deposits a specified amount to the account.

Draw the UML diagram for the class and then implement the class.

Write a test program that creates an Account object with an account ID of 1122, a balance of $20,000, and an annual interest rate of 4.5%. Use the withdraw method to withdraw $2,500, use the deposit method to deposit $3,000, and print the balance, the monthly interest, and the date when this account was created.

(Hint: The method getMonthlyInterest() is to return monthly interest, not the interest rate. Monthly interest is balance \* monthlyInterestRate. monthlyInterestRate is annualInterestRate / 12. Note that annualInterestRate is a percentage,e.g., like 4.5%. You need to divide it by 100.)

Test example :



**package** test;

**import** java.util.\*;

**public** **class** Account {

**public** **static** **void** main(String[] args) {

Account account = **new** Account(1122, 20000);

account.setAnnualInterestRate(4.5);

account.withdraw(2500.0);

account.deposit(3000.0);

System.***out***.println("Balance: $" + account.getBalance());

System.***out***.println("Monthly Interest: " + account.getMonthlyInterest());

System.***out***.println("Date Created: " + account.getDateCreated());

}

**private** **int** id;

**private** **double** balance;

**private** **double** annualInterestRate;

**private** Date dateCreated;

Account(){

dateCreated = **new** Date();

id = 0;

balance = 0;

annualInterestRate = 0;

}

Account(**int** Id, **double** Balance){

dateCreated = **new** Date();

id = Id;

balance = Balance;

annualInterestRate = 0;

}

**void** setId(**int** newid) {

id = newid;

}

**void** setBalance(**int** newbalance) {

balance = newbalance;

}

**void** setAnnualInterestRate(**double** newannualInterestRate) {

annualInterestRate = newannualInterestRate;

}

**int** getId() {

**return** id;

}

**double** getBalance() {

**return** balance;

}

**double** getAnnualInterestRate() {

**return** annualInterestRate;

}

**double** getMonthlyInterestRate() {

**return** annualInterestRate / 12;

}

**double** getMonthlyInterest() {

**return** balance \* (annualInterestRate / 12);

}

String getDateCreated() {

**return** dateCreated.toString();

}

**void** withdraw(**double** amount) {

balance -= amount;

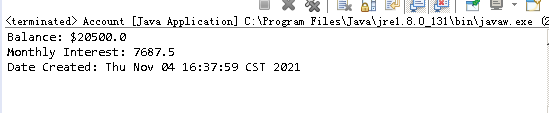
}

**void** deposit(**double** amount) {

balance += amount;

}

}



6.3 (The Fan class) Design a class named Fan to represent a fan. The class contains:

■ Three constants named SLOW, MEDIUM, and FAST with the values 1, 2, and 3 to denote the fan speed.

■ A private int data field named speed that specifies the speed of the fan (the default is SLOW).

■ A private boolean data field named on that specifies whether the fan is on (the default is false).

■ A private double data field named radius that specifies the radius of the fan (the default is 5).

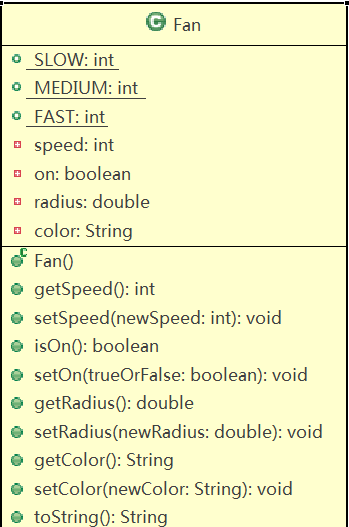
■ A string data field named color that specifies the color of the fan (the default is blue).

■ The accessor and mutator methods for all four data fields.

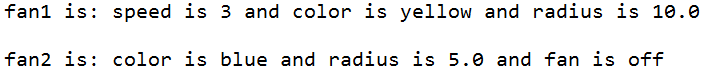
■ A no-arg constructor that creates a default fan.

■ A method named toString() that returns a string description for the fan. If the fan is on, the method returns the fan speed, color, and radius in one combined string. If the fan is not on, the method returns the fan color and radius along with the string “fan is off” in one combined string.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two Fan objects. Assign maximum speed, radius 10, color yellow, and turn it on to the first object. Assign medium speed, radius 5, color blue, and turn it off to the second object. Display the objects by invoking their toString method.



Test example :



6.4 (Geometry: n-sided regular polygon) In an n-sided regular polygon, all sides have the same length and all angles have the same degree (i.e., the polygon is both equilateral and equiangular). Design a class named RegularPolygon that contains:

■ A private int data field named n that defines the number of sides in the polygon with default value 3.

■ A private double data field named side that stores the length of the side with default value 1.

■ A private double data field named x that defines the x-coordinate of the polygon’s center with default value 0.

■ A private double data field named y that defines the y-coordinate of the polygon’s center with default value 0.

■ A no-arg constructor that creates a regular polygon with default values.

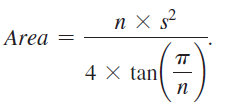
■ A constructor that creates a regular polygon with the specified number of sides and length of side, centered at (0, 0).

■ A constructor that creates a regular polygon with the specified number of sides, length of side, and x- and y-coordinates.

■ The accessor and mutator methods for all data fields.

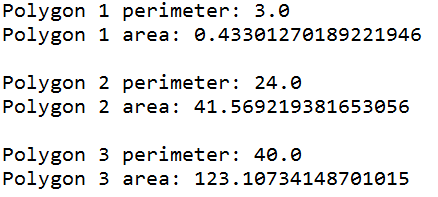
■ The method getPerimeter() that returns the perimeter of the polygon.

■ The method getArea() that returns the area of the polygon. The formula for computing the area of a regular polygon is



Draw the UML diagram for the class and then implement the class. Write a test program that creates three RegularPolygon objects, created using the no-arg constructor, using RegularPolygon(6, 4), and using RegularPolygon(10, 4, 5.6, 7.8). For each object, display its perimeter and area.

Test example :



6.5 (Algebra: quadratic equations) Design a class named QuadraticEquation for a quadratic equation ax2 + bx + x = 0. The class contains:

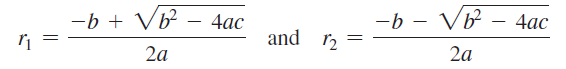
■ Private data fields a, b, and c that represent three coefficients.

■ A constructor for the arguments for a, b, and c.

■ Three getter methods for a, b, and c.

■ A method named getDiscriminant() that returns the discriminant, which is b2 - 4ac.

■ The methods named getRoot1() and getRoot2() for returning two roots of the equation



These methods are useful only if the discriminant is nonnegative. Let these methods return 0 if the discriminant is negative.

Draw the UML diagram for the class and then implement the class. Write a test program that prompts the user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is positive, display the two roots. If the discriminant is 0, display the one root. Otherwise, display “The equation has no roots.”

Test example 1:



Test example 2:



Test example 3:



**3. Please show your questions analysis, code and results.**

# Lab Report 07

Course Name: Object Oriented Programming

Experiment: Lab07 Object-oriented thinking

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to explore the differences between the procedural paradigm and object-oriented paradigm;

1.2 to design programs using the object-oriented paradigm.

**2. Experimental Contents**

7.1 (The Time class) Design a class named Time. The class contains:

■ The data fields hour, minute, and second that represent a time.

■ A no-arg constructor that creates a Time object for the current time. (The values of the data fields will represent the current time.)

■ A constructor that constructs a Time object with a specified elapsed time since midnight, January 1, 1970, in milliseconds. (The values of the data fields will represent this time.)

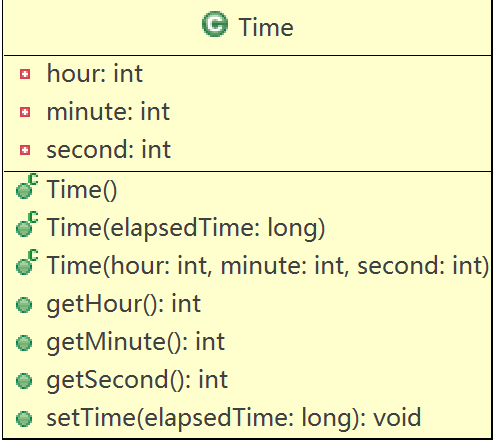
■ A constructor that constructs a Time object with the specified hour, minute, and second.

■ Three getter methods for the data fields hour, minute, and second, respectively.

■ A method named setTime(long elapseTime) that sets a new time for the object using the elapsed time. For example, if the elapsed time is 555550000 milliseconds, the hour is 10（对24求余数，天数舍弃）, the minute is 19, and the second is 10.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two Time objects (using new Time() and new Time(555550000)) and displays their hour, minute, and second in the format hour:minute:second.

(Hint: The first two constructors will extract the hour, minute, and second from the elapsed time. For the no-arg constructor, the current time can be obtained using System.currentTimeMillis()，注意：这个时间和我们这里有8小时时差)



Test example 1:



public class Time {

public static void main(String args[]){

Time t1 = new Time();

System.out.println("Hour: " + t1.hour + " Minute: " + t1.minute + " Second: " + t1.second);

Time t2 = new Time(555550000);

System.out.println("Hour: " + t2.hour + " Minute: " + t2.minute + " Second: " + t2.second);

}

private long hour;

private long minute;

private long second;

Time(){

long totalSeconds = System.currentTimeMillis() / 1000;

long totalMinutes = totalSeconds / 60;

long totalHours = totalMinutes / 60;

this.hour = totalHours % 24;

this.minute = totalMinutes % 60;

this.second = totalSeconds % 60;

}

Time(long elapsedTime){

long totalSeconds = elapsedTime / 1000;

long totalMinutes = totalSeconds / 60;

long totalHours = totalMinutes / 60;

this.hour = totalHours % 24;

this.minute = totalMinutes % 60;

this.second = totalSeconds % 60;

}

Time(int newHour, int newMinute, int newSecond){

hour = newHour;

minute = newMinute;

second = newSecond;

}

long getHour(){

return hour;

}

long getMinute(){

return minute;

}

long getSecond(){

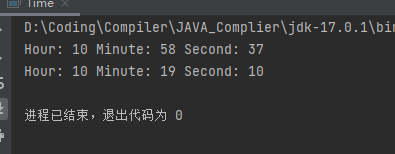
return second;

}

void setTime(long elapsedTime){

}

}



7.2 (The BMI class) Add the following new constructor in the BMI class:

/\*\* Construct a BMI with the specified name, age, weight, feet, and inches\*/

public BMI(String name, int age, double weight, double feet,double inches)

Write a test program that creates two BMI objects as following and display their BMI and status.





Test example:



“public class BMI {

public static void main(String args[]){

BMI b1 = new BMI("John Doe", 18, 145, 5, 10);

System.out.println("The BMI for " + b1.name + " is " + String.format("%.2f", b1.getBMI()) + " " + b1.judge());

BMI b2 = new BMI("John Doe", 215, 5, 10);

System.out.println("The BMI for " + b2.name + " is " + String.format("%.2f", b2.getBMI()) + " " + b2.judge());

}

String name;

int age;

double weight;

double feet;

double inches;

BMI(String newName, int newAge, double newWeight, double newFeet, double newInches){

this.name = newName;

this.weight = newWeight;

this.age = newAge;

this.feet = newFeet;

this.inches = newInches;

}

BMI(String newName, double newWeight, double newFeet, double newInches){

this.name = newName;

this.weight = newWeight;

this.feet = newFeet;

this.inches = newInches;

}

double getBMI(){

return Math.round((weight \* 0.45359237 / Math.pow((feet \* 12 + inches) \* 0.0254, 2)) \* 100) / 100.0;

}

String judge(){

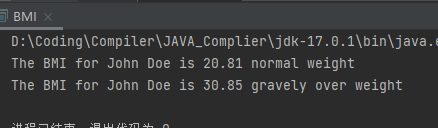
if(this.getBMI() < 20) return "gravely below weight";

else if(this.getBMI() > 25) return "gravely over weight";

else return "normal weight";

}

}



7.3 (The MyInteger class) Design a class named MyInteger. The class contains:

■ An int data field named value that stores the int value represented by this object.

■ A constructor that creates a MyInteger object for the specified int value.

■ A getter method that returns the int value.

■ The methods isEven(), isOdd(), and isPrime() that return true if the value in this object is even, odd, or prime, respectively.

■ The static methods isEven(int), isOdd(int), and isPrime(int) that return true if the specified value is even, odd, or prime, respectively.

■ The static methods isEven(MyInteger), isOdd(MyInteger), and isPrime(MyInteger) that return true if the specified value is even, odd, or prime, respectively.

■ The methods equals(int) and equals(MyInteger) that return true if the value in this object is equal to the specified value.

■ A static method parseInt(char[]) that converts an array of numeric characters to an int value.

■ A static method parseInt(String) that converts a string into an int value.

Draw the UML diagram for the class and then implement the class. Write a client program that tests all methods in the class.

The client program is showed here:

**public** **class** Exercise07\_03 {

**public** **static** **void** main(String[] args) {

MyInteger n1 = **new** MyInteger(5);

System.***out***.println("n1 is even? " + n1.isEven());

System.***out***.println("n1 is prime? " + n1.isPrime());

System.***out***.println("15 is prime? " + MyInteger.*isPrime*(15));

**char**[] chars = {'3', '5', '3', '9'};

System.***out***.println(MyInteger.*parseInt*(chars));

String s = "3539";

System.***out***.println(MyInteger.*parseInt*(s));

MyInteger n2 = **new** MyInteger(24);

System.***out***.println("n2 is odd? " + n2.isOdd());

System.***out***.println("45 is odd? " + MyInteger.*isOdd*(45));

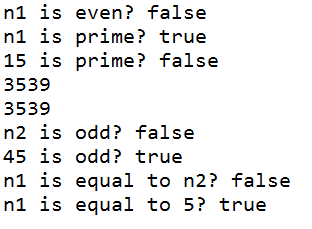
System.***out***.println("n1 is equal to n2? " + n1.equals(n2));

System.***out***.println("n1 is equal to 5? " + n1.equals(5));

}

}

Test example:



public class MyInteger {

public static void main(String[] args) {

MyInteger n1 = new MyInteger(5);

System.out.println("n1 is even? " + n1.isEven());

System.out.println("n1 is prime? " + n1.isPrime());

System.out.println("15 is prime? " + MyInteger.isPrime(15));

char[] chars = {'3', '5', '3', '9'};

System.out.println(MyInteger.parseInt(chars));

String s = "3539";

System.out.println(MyInteger.parseInt(s));

MyInteger n2 = new MyInteger(24);

System.out.println("n2 is odd? " + n2.isOdd());

System.out.println("45 is odd? " + MyInteger.isOdd(45));

System.out.println("n1 is equal to n2? " + n1.equals(n2));

System.out.println("n1 is equal to 5? " + n1.equals(5));

}

private int value;

MyInteger(int newValue){

value = newValue;

}

public int getValue() {

return value;

}

public boolean isEven() {

return (value % 2) == 0;

}

public boolean isOdd() {

return (value % 2) == 1;

}

public boolean isPrime() {

if (value == 1 || value == 2) {

return true;

}

else {

for (int i = 2; i < value; i++) {

if (value % i == 0) return false;

}

}

return true;

}

public static boolean isEven(int myInt) {

return (myInt % 2) == 0;

}

public static boolean isOdd(int myInt) {

return (myInt % 2) == 1;

}

public static boolean isPrime(int myInt) {

if (myInt == 1 || myInt == 2) {

return true;

}

else {

for (int i = 2; i < myInt; i++) {

if (myInt % i == 0) return false;

}

}

return true;

}

public static boolean isEven(MyInteger myInt) {

return myInt.isEven();

}

public static boolean isOdd(MyInteger myInt) {

return myInt.isOdd();

}

public static boolean isPrime(MyInteger myInt) {

return myInt.isPrime();

}

public boolean equals(int testInt) {

if (testInt == value)

return true;

return false;

}

public boolean equals(MyInteger myInt) {

if (myInt.value == this.value)

return true;

return false;

}

public static int parseInt(char[] values) {

int sum = 0;

for (char i : values) {

sum \*= 10;

sum += Character.getNumericValue(i);

}

return sum;

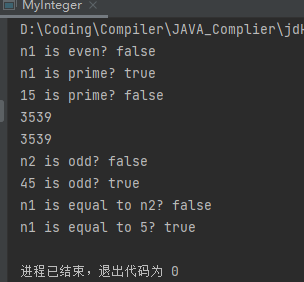
}

public static int parseInt(String value) {

return Integer.parseInt(value);

}

}



7.4 (The MyPoint class) Design a class named MyPoint to represent a point with x- and y-coordinates. The class contains:

■ The data fields x and y that represent the coordinates with getter methods.

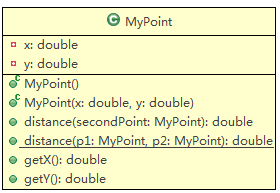
■ A no-arg constructor that creates a point (0, 0).

■ A constructor that constructs a point with specified coordinates.

■ A method named distance that returns the distance from this point to a specified point of the MyPoint type.

■ A method named distance that returns the distance from this point to another point with specified x- and y-coordinates.

Draw the UML diagram for the class and then implement the class. Write a test program that creates the two points (0, 0) and (10, 30.5) and displays the distance between them.



Hint: you can use the following two statements to get the distance between p1 and p2.



Test example:



public class MyPoint {

public static void main(String argus[]){

MyPoint p1 = new MyPoint(0, 0);

MyPoint p2 = new MyPoint(10, 30.5);

System.out.println(p1.distance(p2));

System.out.println(MyPoint.distance(p1, p2));

}

private double x;

private double y;

MyPoint(){

}

MyPoint(double newX, double newY){

x = newX;

y = newY;

}

public double distance(MyPoint secondPoint){

return Math.sqrt(Math.pow(this.x - secondPoint.x, 2) + Math.pow(this.y - secondPoint.y, 2));

}

public static double distance(MyPoint firstPoint, MyPoint secondPoint){

return Math.sqrt(Math.pow(firstPoint.x - secondPoint.x, 2) + Math.pow(firstPoint.y - secondPoint.y, 2));

}

public double getX(){

return x;

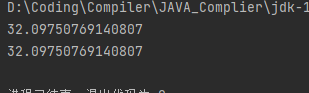
}

public double getY(){

return y;

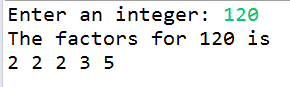
}

}

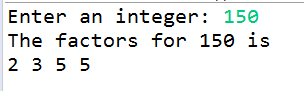


7.5 (Displaying the prime factors) Write a program that prompts the user to enter a positive integer and displays all its smallest factors in decreasing order. For example, if the integer is 120, the smallest factors are displayed as 5, 3, 2, 2, 2. Use the StackOfIntegers class to store the factors (e.g., 2, 2, 2, 3, 5) and retrieve and display them in reverse order.

Test example 1:



Test example 2:



import java.util.Scanner;

public class find {

public static void main(String argus[]){

Scanner input = new Scanner(System.in);

System.out.print("Enter an integer: ");

int number = input.nextInt();

System.out.println("The factors for "+ number + " is ");

int x = 2;

while(number != 1){

if(number % x == 0){

System.out.print(x + " ");

number /= x;

x = 1;

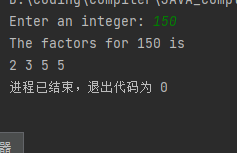
}

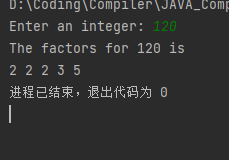
x++;

}

}

}





**3. Please show your questions analysis, code and results.**

# Lab Report 08

Course Name: Object Oriented Programming

Experiment: Lab08 Object-oriented thinking

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to explore the differences between the procedural paradigm and object-oriented paradigm;

1.2 to design programs using the object-oriented paradigm.

**2. Experimental Contents**

8.1 (Geometry: the Circle2D class) Define the Circle2D class that contains:

■ Two double data fields named x and y that specify the center of the circle with getter methods.

■ A data field radius with a getter method.

■ A no-arg constructor that creates a default circle with (0, 0) for (x, y) and 1 for radius.

■ A constructor that creates a circle with the specified x, y, and radius.

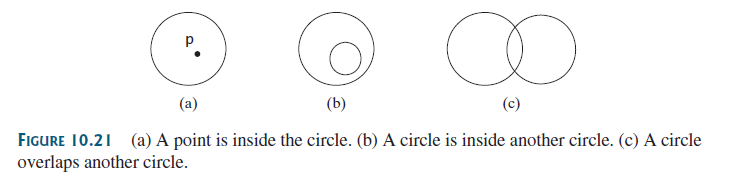
■ A method getArea() that returns the area of the circle.

■ A method getPerimeter() that returns the perimeter of the circle.

■ A method contains(double x, double y) that returns true if the specified point (x, y) is inside this circle (see Figure 10.21a). （hint: 点到圆心点的距离是否小于等于当前圆半径）

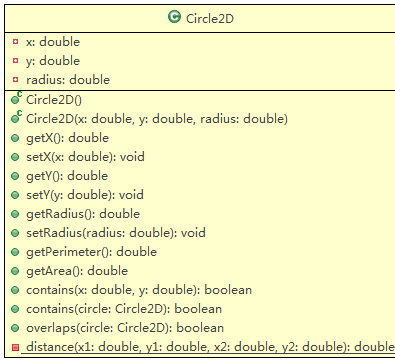
■ A method contains(Circle2D circle) that returns true if the specified circle is inside this circle (see Figure 10.21b).（hint: 两个圆心点之间的距离加上传入参数圆的半径是否小于等于当前圆半径）

■ A method overlaps(Circle2D circle) that returns true if the specified circle overlaps with this circle (see Figure 10.21c).（hint:判断方法：两个圆心点之间的距离小于等于两个圆的半径之和）

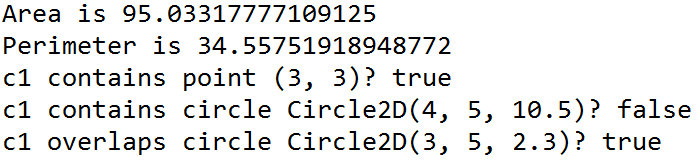


Draw the UML diagram for the class and then implement the class. Write a test program that creates a Circle2D object c1 (new Circle2D(2, 2, 5.5)), displays its area and perimeter, and displays the result of c1.contains(3,3), c1.contains(new Circle2D(4, 5, 10.5)), and c1.overlaps(new Circle2D(3, 5, 2.3)).

（hint:由于总是需要计算两点距离，建议定义一个静态方法计算两点距离，计算方法Math.sqrt((x1 - x2) \* (x1 - x2) + (y1 - y2) \* (y1 - y2))）



Test example:



8.2 (The MyDate class) Design a class named MyDate. The class contains:

■ The data fields year, month, and day that represent a date. month is 0-based, i.e., 0 is for January.

■ A no-arg constructor that creates a MyDate object for the current date.

■ A constructor that constructs a MyDate object with a specified elapsed time since midnight, January 1, 1970, in milliseconds.

■ A constructor that constructs a MyDate object with the specified year, month, and day.

■ Three getter methods for the data fields year, month, and day, respectively.

■ A method named setDate(long elapsedTime) that sets a new date for the object using the elapsed time.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two MyDate objects (using new MyDate() and new MyDate(34355555133101L)) and displays their year, month, and day.

(Hint 1: The first two constructors will extract the year, month, and day from the elapsed time. For example, if the elapsed time is 561555550000 milliseconds, the year is 1987, the month is 9, and the day is 18.

hint2：You may use the GregorianCalendar class to simplify coding.

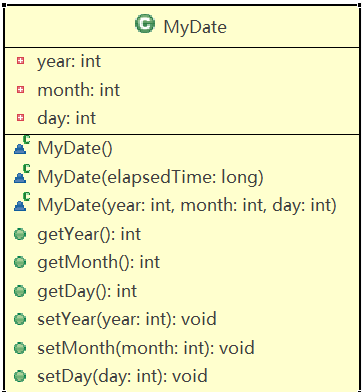
GregorianCalendar date = new GregorianCalendar();

date.get(Calendar.YEAR);

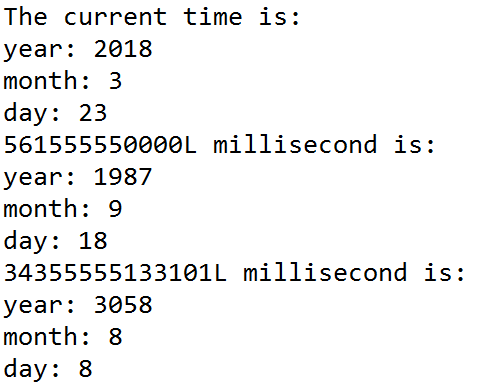
date.get(Calendar.MONTH);

date.get(Calendar.DAY\_OF\_MONTH);

)



Test example:

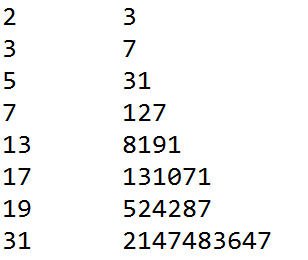


8.3 (Mersenne prime 梅森素数) A prime number is called a Mersenne prime if it can be written in the form 2p - 1 for some positive integer p. Write a program that finds all Mersenne primes with p … 100 and displays the output as shown below.

(Hint 1: You have to use BigInteger to store the number, because it is too big to be stored in long.

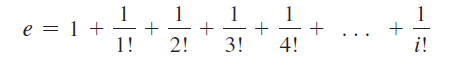
Hint 2：public static boolean isPrime(BigInteger num) 素数判断函数转换为BigInteger版

Hint 3: Your program may take several hours to run.)



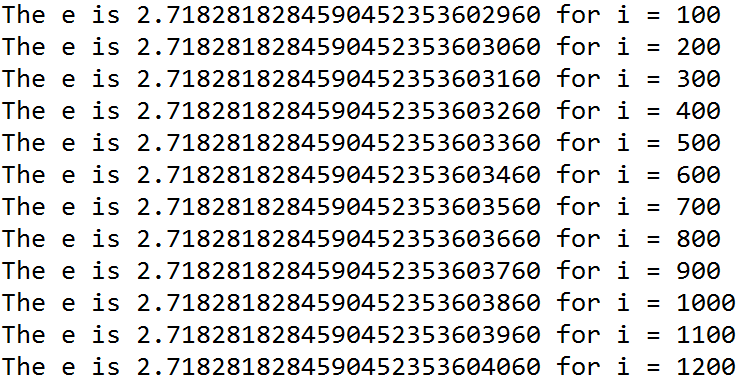


8.4 (Approximate e) Programming Exercise 5.26 approximates e using the following series:

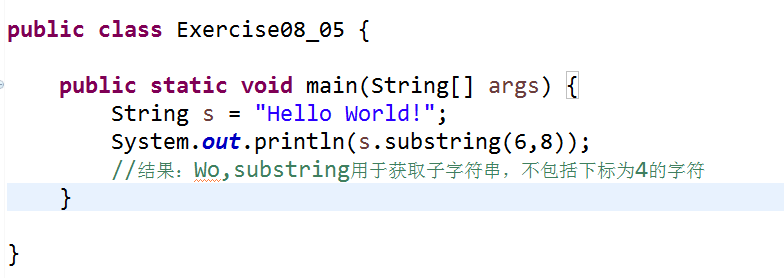


In order to get better precision, use BigDecimal with 25 digits of precision in the computation. Write a program that displays the e value for i = 100, 200, . . . ,and 1000.

Test example:



8.5 整理教材出现的String类的方法例子（Chapter04和Chapter10，并解释运行结果。示例如下：



**3. Please show your questions analysis, code and results.**

# Lab Report 09

Course Name: Object Oriented Programming

Experiment: Lab09 Inheritance and polymorphism

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn inheritance

1.2 to learn polymorphism.

**2. Experimental Contents**

9.1 (The Triangle class) Design a class named Triangle that extends GeometricObject. The class contains:

■ Three double data fields named side1, side2, and side3 with default values 1.0 to denote three sides of the triangle.

■ A no-arg constructor that creates a default triangle.

■ A constructor that creates a triangle with the specified side1, side2, and side3.

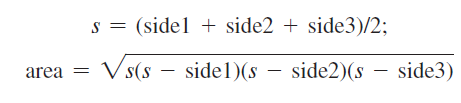
■ The accessor methods for all three data fields.

■ A method named getArea() that returns the area of this triangle.

■ A method named getPerimeter() that returns the perimeter of this triangle.

■ A method named toString() that returns a string description for the triangle.

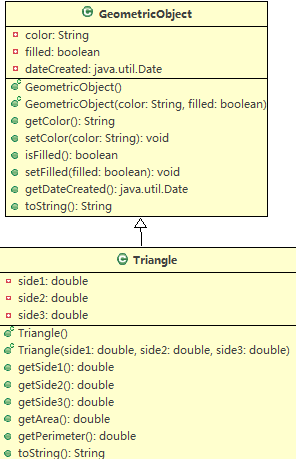
Using the following formula to compute the area of a triangle:



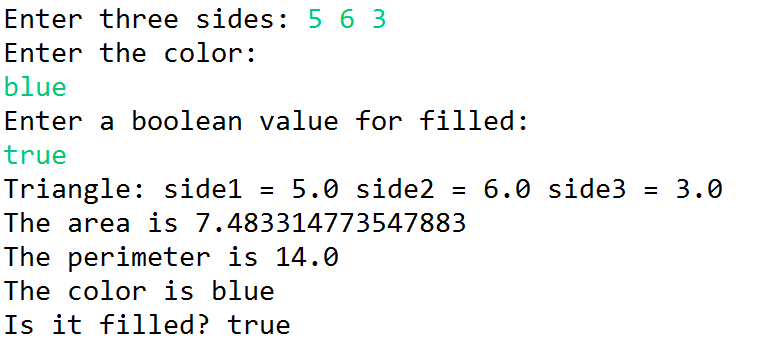
The toString() method is implemented as follows:

return "Triangle: side1 = " + side1 + " side2 = " + side2 +" side3 = " + side3;

Draw the UML diagrams for the classes Triangle and GeometricObject and implement the classes. Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a Triangle object with these sides and set the color and filled properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not.

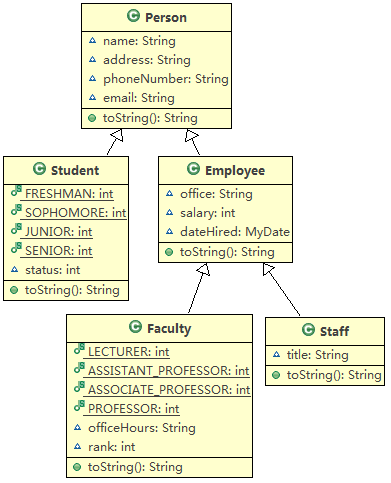


Test example:

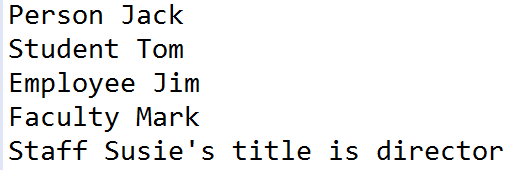


9.2 (The Person, Student, Employee, Faculty, and Staff classes) Design a class named Person and its two subclasses named Student and Employee.Make Faculty and Staff subclasses of Employee. A person has a name, address, phone number, and email address. A student has a class status (freshman, sophomore, junior, or senior). Define the status as a constant. An employee has an office, salary, and date hired. Use the MyDate class defined in Lab08 Exercise8.2 to create an object for date hired. A faculty member has office hours and a rank. A staff member has a title. Override the toString method in each class to display the class name and the person’s name.

Draw the UML diagram for the classes and implement them. Write a test program that creates a Person, Student, Employee, Faculty, and Staff, and invokes their toString() methods.



Test example:



**3. Please show your questions analysis, code and results.**

# Lab Report 10

Course Name: Object Oriented Programming

Experiment: Lab10 Inheritance and polymorphism

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

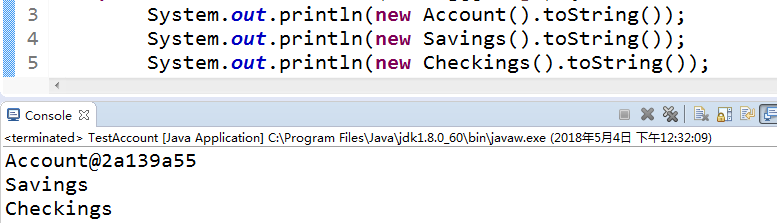
1.1 to learn inheritance

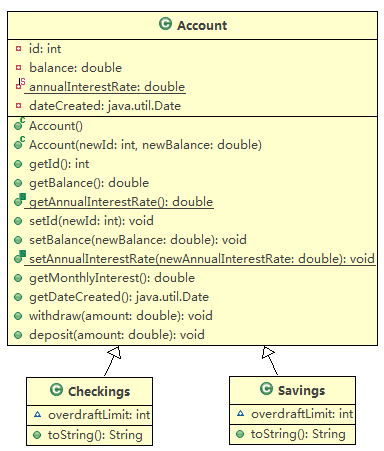
1.2 to learn polymorphism.

**2. Experimental Contents**

10.1 (Subclasses of Account) The Account class was defined to model a bank account. An account has the properties account number, balance, annual interest rate, and date created, and methods to deposit and withdraw funds. Create two subclasses for checking and saving accounts. A checking account has an overdraft limit, but a savings account cannot be overdrawn. Draw the UML diagram for the classes and then implement them. Write a test program that creates objects of Account, SavingsAccount, and CheckingAccount and invokes their toString() methods.

Test example:

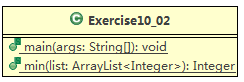




10.2 (Minimum element in ArrayList) Write the following method that returns the minimum value in an ArrayList of integers. The method returns null if the list is null or the list size is 0.

public static Integer min(ArrayList<Integer> list)

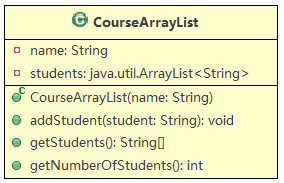
Write a test program that prompts the user to enter a sequence of numbers ending with 0, and invokes this method to return the smallest number in the input.



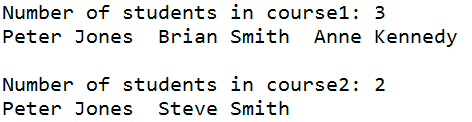
Test example:



10.3 (The Course class) Rewrite the Course class given in the lab10 handout. Use an ArrayList to replace an array to store students. Draw the new UML diagram for the class. You should not change the original contract of the Course class (i.e., the definition of the constructors and methods should not be changed, but the private members may be changed.)

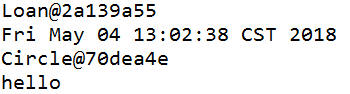


Test example:



10.4 (Use ArrayList) Write a program that creates an ArrayList and adds a Loan object, a Date object, a string, and a Circle object to the list, and use a loop to display all the elements in the list by invoking the object’s toString() method.（Loan/Circle are given in the lab10 handout）

Test example: (说明list中可以放不同类型的对象，区别于数组)



**3. Please show your questions analysis, code and results.**

# Lab Report 11

Course Name: Object Oriented Programming

Experiment: Lab11 Inheritance and polymorphism

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn inheritance

1.2 to learn polymorphism.

**2. Experimental Contents**

11.1 Design a Transaction class as follows:

■ A private data field to store the the date of this transaction.

■ A private data field to store the type of the transaction, such as 'W' for withdrawal, 'D'

■ A private data field to store the amount of the transaction.

■ A private data field to store the new balance after this transaction.

■ The getter methods for all above private data fields.

■ A constructor to construct a Transaction with the specified date, type, balance, and description.

Modify the given Account class as follows:

■ Add a new data field name of the String type to store the name of the customer.

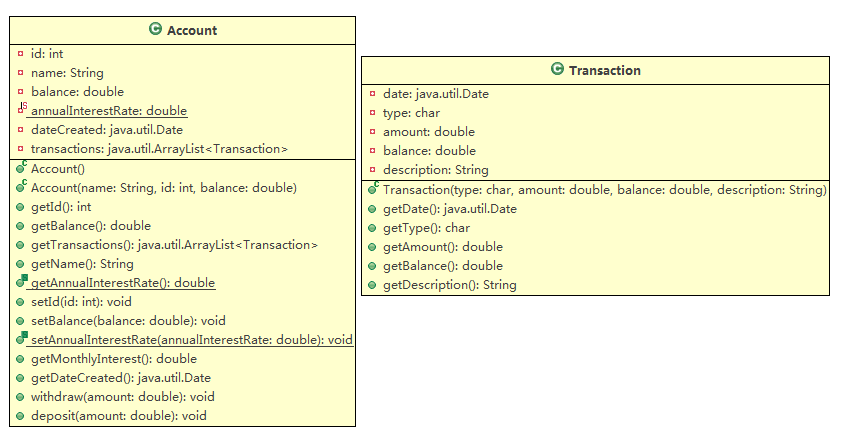
■ Add a new constructor that constructs an account with the specified name, id, and balance.

■ Add a new data field named transactions whose type is ArrayList that stores the transaction for the accounts. Each transaction is an instance of the Transaction class.

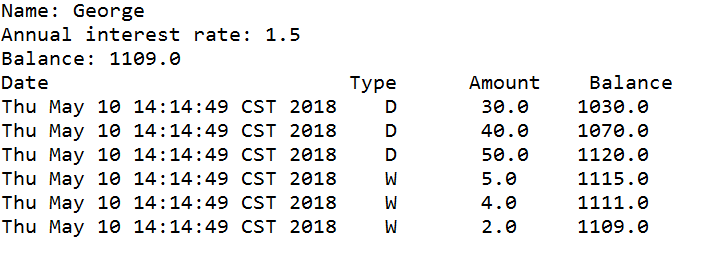
■ Modify the withdraw and deposit methods to add a transaction to the transactions array list.

The UML of Account class and Transaction class is shown as the following.

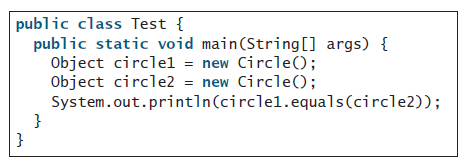
Write a test program that creates an Account with annual interest rate 1.5%, balance 1000, id 1122, and name George. Deposit $30, $40, and $50 to the account and withdraw $5, $4, and $2 from the account. Print an account summary that shows account holder name, interest rate, balance, and all transactions.

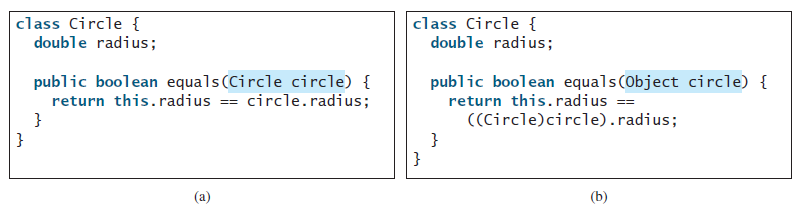


Test example:



11.2 When overriding the equals method, a common mistake is mistyping its signature in the subclass. For example, the equals method is incorrectly written as equals(Circle circle), as shown in (a) in following the code; instead, it should be equals(Object circle), as shown in (b). Show the output of running class Test with the Circle class in (a) and in (b), respectively.





11.3 What is wrong in the following code?

1 public class Test {

2 public static void main(String[] args) {

3 Object fruit = new Fruit();

4 Object apple = (Apple)fruit;

5 }

6 }

7

8 class Apple extends Fruit {

9 }

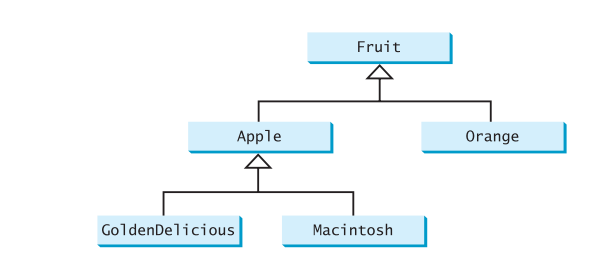
10

11 class Fruit {

12 }

11.4 Suppose that Fruit, Apple, Orange, GoldenDelicious, and Macintosh are

declared, as shown in following Figure.



Assume that the following code is given:

Fruit fruit = new GoldenDelicious();

Orange orange = new Orange();

Answer the following questions:

(1) Is fruit instanceof Fruit?

(2) Is fruit instanceof Orange?

(3) Is fruit instanceof Apple?

(4) Is fruit instanceof GoldenDelicious?

(5) Is fruit instanceof Macintosh?

(6) Is orange instanceof Orange?

(7) Is orange instanceof Fruit?

(8) Is orange instanceof Apple?

(9) Suppose the method makeAppleCider is defined in the Apple class. Can fruit invoke this method? Can orange invoke this method?

(10) Suppose the method makeOrangeJuice is defined in the Orange class. Can orange invoke this method? Can fruit invoke this method?

(11) Is the statement Orange p = new Apple() legal?

(12) Is the statement Macintosh p = new Apple() legal?

(13) Is the statement Apple p = new Macintosh() legal?

11.5 Indicate true or false for the following statements:

a. A protected datum or method can be accessed by any class in the same package.

b. A protected datum or method can be accessed by any class in different packages.

c. A protected datum or method can be accessed by its subclasses in any package.

d. A final class can have instances.

e. A final class can be extended.

f. A final method can be overridden.

**3. Please show your questions analysis, code and results.**

# Lab Report 12

Course Name: Object Oriented Programming

Experiment: Lab12 Exception Handling

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn exception and exception handling

1.2 to learn the advantage of exception

**2. Experimental Contents**

12.1 What RuntimeException will the following programs throw, if any?

**public** **class** Test {

**public** **static** **void** main(String[] args) {

System.***out***.println(1 / 0);

}

}

(a)

**public** **class** Test {

**public** **static** **void** main(String[] args) {

**int**[] list = **new** **int**[5];

System.***out***.println(list[5]);

}

}

(b)

**public** **class** Test {

**public** **static** **void** main(String[] args) {

String s = "abc";

System.***out***.println(s.charAt(3));

}

}

(c)

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Object o = **new** Object();

String d = (String) o;

}

}

(d)

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Object o = **null**;

System.***out***.println(o.toString());

}

}

(e)

**public** **class** Test {

**public** **static** **void** main(String[] args) {

System.***out***.println(1.0 / 0);

}

}

(f)

12.2 Suppose that statement2 causes an exception in the following try-catch block:

try {

statement1;

statement2;

statement3;

}

catch (Exception1 ex1) {

}

catch (Exception2 ex2) {

}

statement4;

Answer the following questions (Use the given Exercise12\_2 .java to analize):

■ Will statement3 be executed?

■ If the exception is not caught, will statement4 be executed?

■ If the exception is caught in the catch block, will statement4 be executed?

12.3 What is displayed when the following program is run?Why?

class Test {

public static void main(String[] args) {

try {

int[] list = new int[10];

System.out.println("list[10] is " + list[10]);

} catch (ArithmeticException ex) {

System.out.println("ArithmeticException");

} catch (RuntimeException ex) {

System.out.println("RuntimeException");

} catch (Exception ex) {

System.out.println("Exception");

}

}

}

12.4 What is displayed when the following program is run?Why?

class Test{

public static void main(String[] args) {

try {

method();

System.out.println("After the method call");

} catch (ArithmeticException ex) {

System.out.println("ArithmeticException");

} catch (RuntimeException ex) {

System.out.println("RuntimeException");

} catch (Exception e) {

System.out.println("Exception");

}

}

static void method() throws Exception {

System.out.println(1 / 0);

}

}

12.5 What is displayed when the following program is run?Why?

class Test {

public static void main(String[] args) {

try {

method();

System.out.println("After the method call");

} catch (RuntimeException ex) {

System.out.println("RuntimeException in main");

} catch (Exception ex) {

System.out.println("Exception in main");

}

}

static void method() throws Exception {

try {

String s = "abc";

System.out.println(s.charAt(3));

} catch (RuntimeException ex) {

System.out.println("RuntimeException in method()");

} catch (Exception ex) {

System.out.println("Exception in method()");

}

}

}

12.6 Suppose that statement2 causes an exception in the following statement:

try {

statement1;

statement2;

statement3;

}

catch (Exception1 ex1) {

}

finally {

statement4;

}

statement5;

Answer the following questions(Use the give Exercise12\_06.java to analyze):

■ If no exception occurs, will statement4 be executed, and will statement5 be

executed?

■ If the exception is of type Exception1, will statement4 be executed, and will

statement5 be executed?

■ If the exception is not of type Exception1, will statement4 be executed, and

will statement5 be executed?

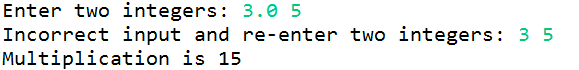
12.7 (InputMismatchException) Write a program that prompts the user to read two integers and displays their multiplication. Your program should prompt the user to read the number again if the input is incorrect.

Hint: you can refer to the book code Listing 12.5.

Test example 1:



Test example 2:



12.8 (ArrayIndexOutOfBoundsException) Write a program that meets the following

requirements:

■ Creates an array with 100 randomly chosen integers.

■ Prompts the user to enter the index of the array, then displays the corresponding element value. If the specified index is out of bounds, display the message Out of Bounds.

Test example 1:



Test example 2:



**3. Please show your questions analysis, code and results.**

# Lab Report 13

Course Name: Object Oriented Programming

Experiment: Lab13 Text file read and write

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to read data from text file

1.2 to learn how to write data to text file

**2. Experimental Contents**

**13.1** Show the contents of the file temp.txt after the following program is executed.

class Test {

public static void main(String[] args) throws Exception {

java.io.PrintWriter output = new java.io.PrintWriter("temp.txt");

output.printf("amount is %f %e\r\n", 32.32, 32.32);

output.printf("amount is %5.4f %5.4e\r\n", 32.32, 32.32);

output.printf("%6b\r\n", (1 > 2));

output.printf("%6s\r\n", "Java");

output.close();

}

}

**13.2** Rewrite the code in the preceding 13.1 question using a try-with-resources syntax.

**13.3** Suppose you enter 45 57.8 789, then press the Enter key. Show the contents of the variables after the following code is executed. Why? write program to analyze the result.

Scanner input = new Scanner(System.in);

int intValue = input.nextInt();

double doubleValue = input.nextDouble();

String line = input.nextLine();

**13.4**  Suppose you enter 45, press the Enter key, 57.8, press the Enter key, 789, and press the Enter key. Show the contents of the variables after the following code is executed. why?

Scanner input = new Scanner(System.in);

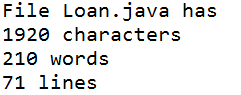
int intValue = input.nextInt();

double doubleValue = input.nextDouble();

String line = input.nextLine();

**13.5**  (Count characters, words, and lines in a file) Write a program that will count the number of characters, words, and lines in a file(the given Loan.java in the lab handout). Words are separated by whitespace characters.

Test example:



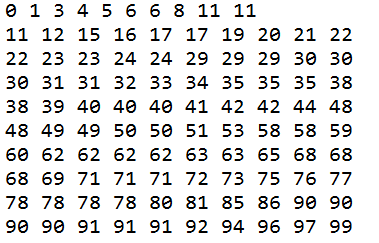
**13.6**  (Process scores in a text file) Suppose that a text file contains an unspecified number of scores separated by blanks. Write a program to read the file(the given testdata.txt file), reads the scores from the file, and displays their total and average.

Test example:



**13.7**  (Write/read data) Write a program to create a file named rw.txt if it does not exist. Write 100 integers(range from 0 to 99) created randomly into the file using text I/O. Integers are separated by spaces in the file. Read the data back from the file and display the data in increasing order (ten numbers per line).

Test example:



**13.8** (Count words) Write a program that counts the number of words in President Abraham Lincoln’s Gettysburg address from <http://cs.armstrong.edu/liang/data/Lincoln.txt>.

**Hint: you can refer to the book code Listing 12.17.**

Test example:



**3. Please show your questions analysis, code and results.**

# Lab Report 14

Course Name: Object Oriented Programming

Experiment: Lab14 Abstract class

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

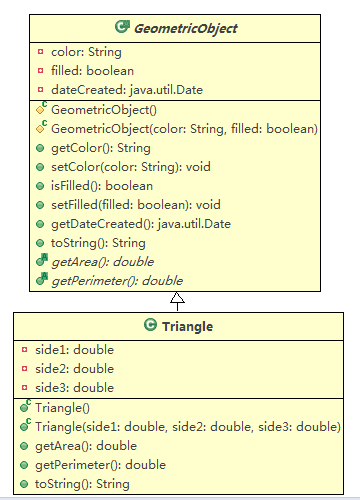
**1. Experimental Purposes and Requirements**

1.1 to learn how to define and use abstract class

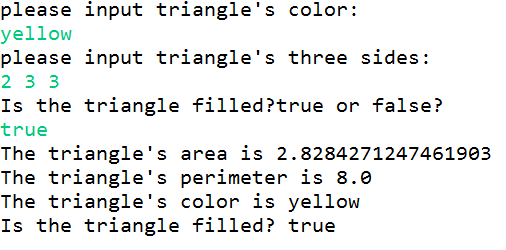
1.2 to learn why we need abstract class

**2. Experimental Contents**

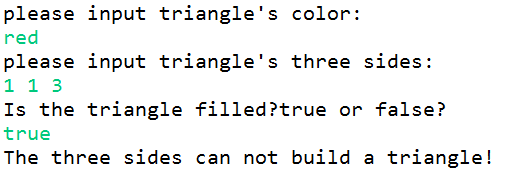
**14.1** (Triangle class) Design a new Triangle class that extends the abstract GeometricObject class. Draw the UML diagram for the classes Triangle and GeometricObject and then implement the Triangle class. Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a Triangle object with these sides and set the color and filled properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not.



Test example 1:



Test example 2:



**14.2** (Sort ArrayList) Write the following method that sorts an ArrayList of numbers.

public static void sort(ArrayList<Number> list)

Hint: you can refer to book code Listting 13.5

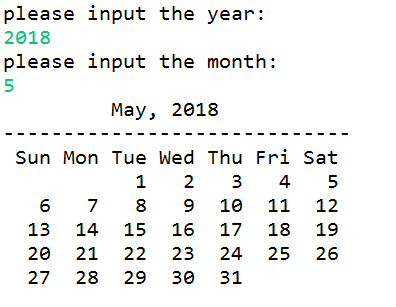
Test example 1:



**14.3** (Display calendars) Display a calendar for a specified month using the Calendar and GregorianCalendar classes.

Hint: you can refer to book code Listing 6.12

Test example :



**14.4** (ArrayList average) Write the following method that average an ArrayList of numbers:

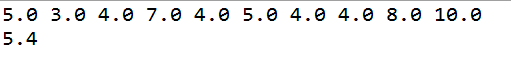
public static void average(ArrayList<Number> list)

Hint: the size of ArrayList is random and the numbers in ArrayList are also random.

Test example 1:



Test example 2:



Test example 3:



**3. Please show your questions analysis, code and results.**

# Lab Report 15

Course Name: Object Oriented Programming

Experiment: Lab15 Interface

Student Name: Class No.

Note:

* All your lab reports should be uploaded to BB before the deadline.

Caution

* Must be original works, to prohibit any copying or plagiarism.

**1. Experimental Purposes and Requirements**

1.1 to learn how to define and use interface

1.2 to learn why we need interface and multi-inheritance

**2. Experimental Contents**

**15.1** (Enable GeometricObject comparable) Modify the GeometricObject class to implement the Comparable interface, and define a static max method in the GeometricObject class for finding the larger of two GeometricObject objects. Draw the UML diagram and implement the new GeometricObject class. Write a test program that uses the max method to find the larger of two circles and the larger of two rectangles.

hint: 1) the max method return the object with bigger area of two geometric objects.

2) you can use the following code to test your class.

GeometricObject o1 = new Circle(5);

GeometricObject o2 = new Rectangle(3,4);

GeometricObject o3 = new Rectangle(5,6);

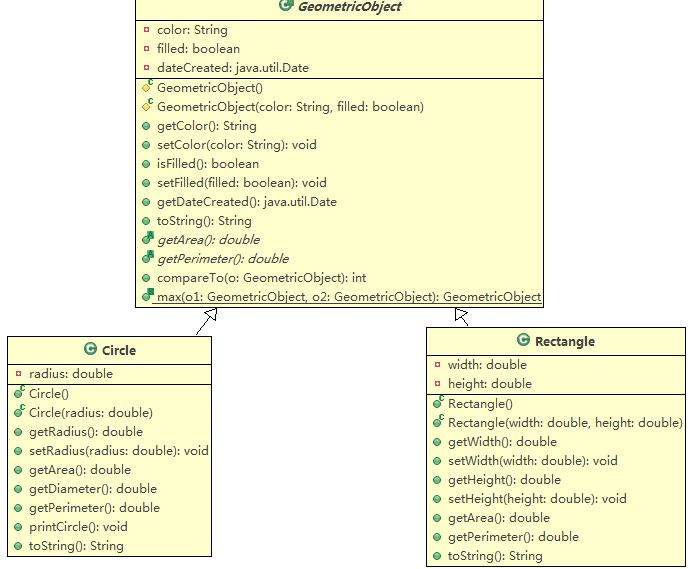
GeometricObject o4 = new Circle(2);

System.out.println("the max of o1 and o2 is: "+ GeometricObject.max(o1,o2));

System.out.println("the max of o3 and o4 is: "+ GeometricObject.max(o3, o4));

Test example:

1527213200(1)



**15.2** (The ComparableCircle class) Define a class named ComparableCircle that extends Circle and implements Comparable. Draw the UML diagram and implement the compareTo method to compare the circles on the basis of area. Write a test class to find the larger of two instances of ComparableCircle objects.

hint: you can use the following code to test your class.

ComparableCircle circle1 = new ComparableCircle(5);

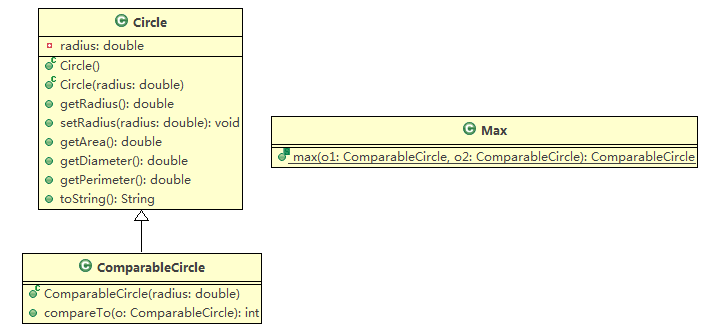
ComparableCircle circle2 = new ComparableCircle(15);

ComparableCircle circle3 = (ComparableCircle)Max.max(circle1, circle2);

System.out.println("The max circle's radius is " + circle3.getRadius());

Test example:





**15.3** (The Colorable interface) Design an interface named Colorable with a void method named howToColor(). Every class of a colorable object must implement the Colorable interface. Design a class named Square that extends GeometricObject and implements Colorable. Implement howToColor to display the message Color all four sides. Draw a UML diagram that involves Colorable, Square, and GeometricObject. Write a test program that creates an array of five GeometricObjects. For each object in the array, display its area and invoke its howToColor method if it is colorable.

hint: you can use the following code to test your class.

GeometricObject[] objects = {new Square(2), new Circle(5), new Square(5), new Rectangle(3, 4), new Square(4.5)};

for (int i = 0; i < objects.length; i++) {

if (objects[i] instanceof Colorable)

((Colorable)objects[i]).howToColor();

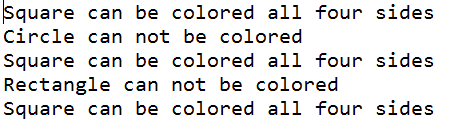
else {

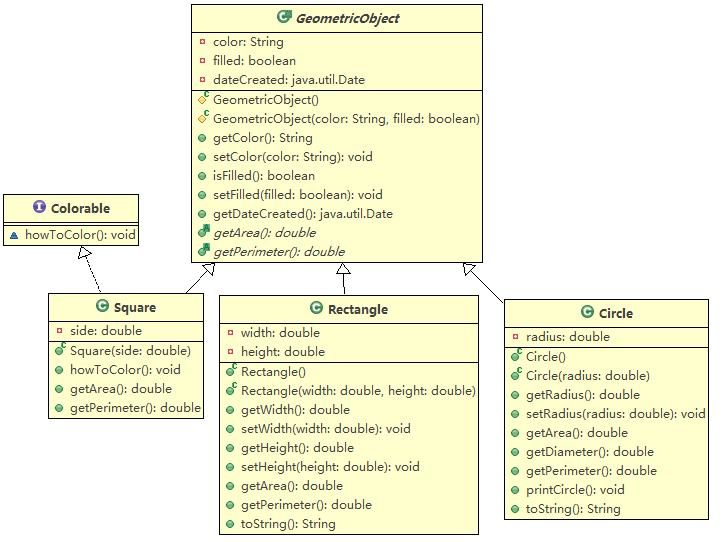
System.out.println(objects[i].toString()+" can not be colored");

}

}

Test example:





**15.4**  (Enable Circle comparable) Rewrite the given Circle class extend GeometricObject and implement the Comparable interface. Override the equals method in the Object class. Two Circle objects are equal if their radii are the same. Draw the UML diagram that involves Circle, GeometricObject, and Comparable.

hint: you can use the following code to test your class.

Circle obj1 = new Circle(1);

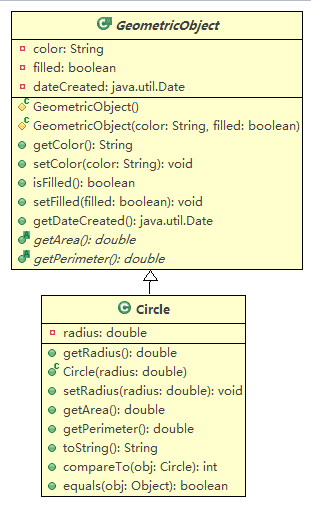
Circle obj2 = new Circle(2);

System.out.println(obj1.equals(obj2));

System.out.println(obj1.compareTo(obj2));

Test example:





**3. Please show your questions analysis, code and results.**