**SUTD 50.001 Introduction to Information Systems and Programming**

**Problem Set 1A**

|  |
| --- |
| **Note:**   * **For all questions, please access the vocareum link found at eDimension for the starter code and to submit.** * **The Vocareum link is for submission only. Please work on the problems in Android studio, and this includes writing code for the test cases.** * **To prevent hard-coding, test cases used in Vocareum *may* be different from those provided here and will not be given to you.** * **Tips on using Android Studio are found at the end of this problem set** |

**Cohort Questions (Week 1)**

**Session 1**

**Title: Fibonacci Numbers Generator**

1. [5 points] Write a JAVA program that returns the first *n* numbers in the fibonacci sequence, in this format:

Eg. For *n* = 5 the output is

0,1,1,2,3

When submitting, return these numbers in this format as a string, instead of printing.

**Session 2**

**Title: Iterating with Iterator**

1. [5 points] Suppose that *integers* is a variable of type List<Integer>. Write a program that uses an iterator to compute the sum of all integer values in the List.

(Test case inputs: (1, 2, 3, 4, 5) Expected output: 15)

**Title: Iterating with For-Each**

1. [5 points] Write a second program that does the same thing as in the previous question but using a for-each loop. (Test case inputs: (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) Expected output: 55)

**Session 3**

**Title: The Account Class**

1. [10 points] Design a class name Account that contains:

Private int data field named id for the account (default 0)

Private double data field named balance for the account (default 0)

Private double data field named annualInterestRate that stores the current interest rate (in percentage, 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

Write a test program that creates an Account object, with withdraw and deposit method to withdraw / deposit the amount, and print the balance, monthly interest and the date when the account was created. Note that the account balance is allowed to be negative.

Test case

public class TestAccount{

public static void main (String[] args) {

Account account = new Account(1122, 20000);

Account.setAnnualInterestRate(4.5);

account.withdraw(2500);

account.deposit(3000);

System.out.println("Balance is " + account.getBalance());

System.out.println("Monthly interest is " +

account.getMonthlyInterest());

}

}

Expected output

Balance is 20500.0

Monthly interest is 76.875

**Homework Questions (Week 1)**

**Title: Prime Number Checker**

1. [5 points] Write a static method, that reads in a number (you can assume that the input number is always >=3). Return 1 if it is prime, return 0 if it is not prime.

Hints: use %. a%b= remainder of a/b. e.g. 13%5=3, 4%2=0

(Test case inputs: 4, 7, 14, 23, 99 Expected outputs: 0, 1, 0, 1, 0)

1. [20 points] **Geometry: The MyRectangle2D class**

Define the MyRectangle2D class that contains:

• Two double data fields named x and y that specify the center of the rectangle with get and set methods: getX, setX, getY, setY. (Assume that the rectangle sides are parallel to x- or y- axes.)

• The double data fields width and height with get and set methods: getWidth, setWidth, getHeight, setHeight.

• A no-arg constructor that creates a default rectangle with (0, 0) for (x, y) and 1 for both width and height.

• A constructor that creates a rectangle with the specified x, y, width, and height: MyRectangle2D(double x, double y, double width, double height)

• A method getArea() that returns the area of the rectangle.

• A method getPerimeter() that returns the perimeter of the rectangle.

• A method contains(double x, double y) that returns true if the specified point (x, y) is inside this rectangle. See Figure 1(a).

• A method contains(MyRectangle2D r) that returns true if the specified rectangle is inside this rectangle. See Figure 1(b).

• A method overlaps(MyRectangle2D r) that returns true if the specified rectangle overlaps with this rectangle. See Figure 1(c).

  

(a) (b) (c)

Figure 1: (a) A point is inside the rectangle. (b) A rectangle is inside another rectangle. (c) A rectangle overlaps another rectangle.

Implement data fields, all constructors, methods getArea(), getPerimeter(), and contains(double x, double y), contains(MyRectangle2D r), and overlaps(MyRectangle2D r)

Please develop test cases to test your code properly before submission

**Homework Questions (Week 2)**

**Title: 2 x 2 Linear Equations**

1. [10 points] Design a class named LinearEquation for a 2x2 system equations of variables x,y:

ax+by=e

cx+dy=f

The class contains:

Private data fields a, b, c, d, e, and f, the coefficients.

A constructor with the arguments for a, b, c, d, e, and f.

Six get methods for a, b, c, d, e, and f.

A method named isSolvable() that returns true if ad - bc is not 0 (why?)

Methods getX() and getY() that return the solution for the equation

Write a test program that prompts user to enter a, b, c, d, e and f and displays the result.

If ad – bc is 0, report that “The equation has no solution.”

Test case:

public class TestLinearEquation {

public static void main(String[] args) {

double a = 1.0; double b = 2.0; double c = 3.0;

double d = 5.0; double e = 6.0; double f = 7.0;

LinearEquation equation = new LinearEquation(a, b, c, d, e, f);

if (equation.isSolvable()) {

System.out.println("x is " +

equation.getX() + " and y is " + equation.getY());

}

else {

System.out.println("The equation has no solution");

}

LinearEquation equation2 = new LinearEquation(1.25, 2.0, 2.0, 4.2, 3.0, 6.0);

if (equation2.isSolvable()) {

System.out.println("x is " + equation2.getX() + " y is " + equation2.getY());

}

LinearEquation equation3 = new LinearEquation(1.0, 2.0, 2.0, 4.0, 3.0, 6.0);

System.out.println(equation3.isSolvable());

}

}

Output:

x is -16.0 and y is 11.0

x is 0.48000000000000115 y is 1.2

false

**Title: The Triangle class.**

1. [10 points] Design a class named Triangle that extends GeometricObject. The class contains:

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

A no-arg constructor to create a default triangle

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

A method named getArea() that returns the area

A method named getPerimeter() that returns the perimeter

A method named toString() that returns description of the triangle

Triangle: side1 = 1.0 side2 = 2.0 side3 = 3.0

Write a test program to test the code. The program should create the Triangle object with these sides and color and filled properties set.

Test case:

public class TestTriangle {

public static void main(String[] args) {

Triangle myTri = new Triangle();

myTri.setColor("red");

myTri.setFilled(true);

System.out.println(myTri);

System.out.println(myTri.isFilled());

Triangle myTri2 = new Triangle(2.0, 4.0, 5.5);

System.out.println(myTri2);

System.out.println("area : " + myTri2.getArea() + " perimeter: " + myTri2.getPerimeter());

}

}

Output

Triangle: side1 = 1.0 side2 = 1.0 side3 = 1.0

true

Triangle: side1 = 2.0 side2 = 4.0 side3 = 5.5

area : 3.0714155938264036 perimeter: 11.5

**Title: Subclasses of Account**

1. [10 points] In Week-1, the Account class was defined to model a bank account. An account has the properties: account id, balance, annual interest rate, and date created, and methods to deposit and withdraw funds. Create a subclass for checking account CheckingAccount. A checking account has an overdraft limit of 5000. Provide constructors for CheckingAccount similar to Account. Override withdraw() to print out “over limit” if the amount withdrawing exceeds the overdraft limit.

Test Case:

public class TestCheckingAccount {

public static void main(String[] args) {

CheckingAccount myCheckAcc = new CheckingAccount(1024, 8000.0);

myCheckAcc.deposit(2000);

myCheckAcc.withdraw(15000);

System.out.println(myCheckAcc.getBalance());

myCheckAcc.withdraw(200);

System.out.println(myCheckAcc.getBalance());

myCheckAcc.deposit(7000);

myCheckAcc.withdraw(200);

System.out.println(myCheckAcc.getBalance());

}

}

Output

-5000.0

over limit

-5000.0

1800.0

**Title: String Operation**

1. [30 points]

(Part-I) Design and implement a static method to determine if an input string has all unique characters. Assume the character set is ASCII, which encodes 128 characters into 7-bit binary integers.

<http://en.wikipedia.org/wiki/ASCII>

Design the most efficient method you can think of, with respect to processing speed.

(Part-II) Design and implement a static method to determine if two input strings are permutation of each other. Assume the character set is ASCII, which encodes 128 characters into 7-bit binary integers.

Design the most efficient method you can think of, with respect to processing speed.

Package your methods into class Pset1.

public class Pset1 {

public static boolean isAllCharacterUnique(String sIn) {

//todo: add your implementation

}

public static boolean isPermutation(String sIn1, String sIn2) {

//todo: add your implementation

}

}

Test case:

public class TestPset1 {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

System.out.println(Pset1.isAllCharacterUnique("abcdefghijklmnopqrstuvABC"));

System.out.println(Pset1.isAllCharacterUnique("abcdefgghijklmnopqrstuvABC"));

System.out.println(Pset1.isPermutation("@ab", "a@b"));

System.out.println(Pset1.isPermutation("abcd", "bcdA"));

}

}

Output:

true

false

true

false

**Android Studio Tips**

1. To run a pure Java project in android studio

* Launch Android Studio
* Select **Start A New Android Studio Project**
* Name your project at the **Create Android Project**, click Next,
* Select all defaults at **Target Android Devices** (i.e. just click Next)
* At **Add An Activity to Mobile**, select **Add No Activity**, click Finish
* Wait for a while …
* Select **File** 🡪 **New 🡪 New Module** and choose **Java Library**, enter the relevant information and click Finish.

For more information:

https://stackoverflow.com/questions/16626810/can-android-studio-be-used-to-run-standard-java-projects

1. Automatically generate constructors, getters and setters:

* Place the cursor anywhere within your class definition.
* Select **Code** 🡪 **Generate** and choose the relevant option.

1. To pass arguments to your public static void main() function:

* Select **Run** 🡪 **Edit Configurations**
* Enter the arguments in the **Program Arguments** box