# Lab Session 01: Fundamentals of Java Programming, Data types, Variables, Operators, Control Statements

# Date of the Session: 26/09/2020 Time of the Session: 11:20AM to 1:00PM

**Program Title:** Fundamentals of Java Programming, Data types, Variables, Operators, Control Statements, Tools used in the lab (Eclipse, JDK Toolkit)

**Pre Lab Task:** Write answers before entering into lab.

**Writing space for pre task :***( For Student’s use only)*

1. Identify and correct the errors in the program below, which prompts for the user’s age and then attempts to work out the year in which the user was born.

import java.util.Scanner;

public class SomeProg

{

public static void main (String[] args)

{

Scanner keyboard = new Scanner(System.in);

int YEAR;

int age, bornIn;

System.out.print(How old are you this year? );

age = keyboard.nextDouble();

bornIn = YEAR – age;

System.out.println("I think you were born in " + BornIn);

}

}

Ans:- import java.util.Scanner;

public class SomeProg

{

public static void main (String[] args)

{

Scanner keyboard = new Scanner(System.in);

int YEAR;

int age, bornIn;

System.out.print(“How old are you this year? “);

age = keyboard.nextInt();

bornIn = YEAR – age;

System.out.println("I think you were born in " + bornIn);

}

}

1. The program below was written in an attempt to swap the value of two variables. However, it does not give the desired result:

Import java.util.Scanner;

public class SwapAttempt

{

public static void main(String[] args)

{

Scanner keyboard = new Scanner(System.in);

// declare variables

int x, y;

// enter values

System.out.print("Enter value for x ");

x = keyboard.nextInt();

System.out.print("Enter value for y ");

y = keyboard.nextInt();

// code attempting to swap two variables

x = y;

y = x;

//display results

System.out.println("x = " + x);

System.out.println("y = " + y);

}

}

1. Can you see why the program doesn’t do what we hoped?
2. What would be the actual output of the program?
3. How could you modify the program above so that the values of the two variables are swapped successfully?

Ans:- (a) The program doesn’t do what we hoped since we didn’t store the x value in any temp variable and over writed it’s value with y. So, we lost the original value of x and both x,y values are printed the same.

(b) x = y value

y = x value

(c) // code attempting to swap two variables

x = x + y;

y = x – y;

x = x – y;

3. Consider the following program:

Import java.util.Scanner;

public class Colours

{

public static void main(String[] args)

{

int x;

Scanner keyboard = new Scanner(System.in);

System.out.print("Enter a number: ");

x = keyboard.nextInt();

if (x > 10)

{

System.out.println("Green");

System.out.println("Blue");

}

System.out.println("Red");

}

}

What would be the output from this program if

1. The user entered 10 when prompted?
2. The user entered 20 when prompted?

(c) The braces used in the if statement are removed, and the user enters 10 when prompted?

(d) The braces used in the if statement are removed, and the user enters 20 when prompted?

Ans:- (a) Red

(b) Green

Blue

Red

(c) Blue

Red

(d) Green

Blue

Red

**In Lab Task**

1. **Problem Description:**

Write the Steps involved in Verification and Installation of a JDK and JRE Software, demonstrate the setting up environment variables path and class path to JDK and JRE in a system. Develop a simple “Hello World” program and test the compilation and execution of the program.

**Test Cases:**

**Test Case 1:**

**Input:**

Enter the commands “java and javac” in command prompt environment to check whether system is able to recognize the commands and run the appropriate executable components.

**Output:**

Successful for running java and javac for compiling and executing java files.

**Test Case 2:**

Save, compile and run a sample java programs to check whether programs able to compile and run java files.

**Writing space for In lab:***(For Student’s use only)*

**Steps for JDK Installation**

You run a self-installing executable file to unpack and install the JDK on Windows computers.

Install JDK on Windows computers by performing the actions described in the following topics:

        Downloading the JDK Installer

        Running the JDK Installer

        Installing the JDK Silently

        Setting the PATH Environment Variable

## **Steps for JRE Installation**

When installing JRE on Windows computers, you must select the JRE installer that is appropriate for your Windows system.

The 64-bit Windows operating systems come with a 64-bit Internet Explorer (IE) browser as the standard (default) for viewing web pages.

Install JRE on Windows computers by performing the actions described in the following topics:

        JRE Proxy Settings and Authentication

        Downloading the JRE Installer

        Running the JRE Installer

### **Setting the PATH Environment Variable**

It is useful to set the PATH variable permanently for JDK 10 so that it is persistent after rebooting.

If you do not set the PATH variable, then you must specify the full path to the executable file every time that you run it. For example:

C:\> "C:\Program Files\Java\jdk-10\bin\javac" MyClass.java

To set the PATH variable permanently, add the full path of the jdk-10\bin directory to the PATH variable. Typically, the full path is:

C:\Program Files\Java\jdk-10\bin

To set the PATH variable on Microsoft Windows:

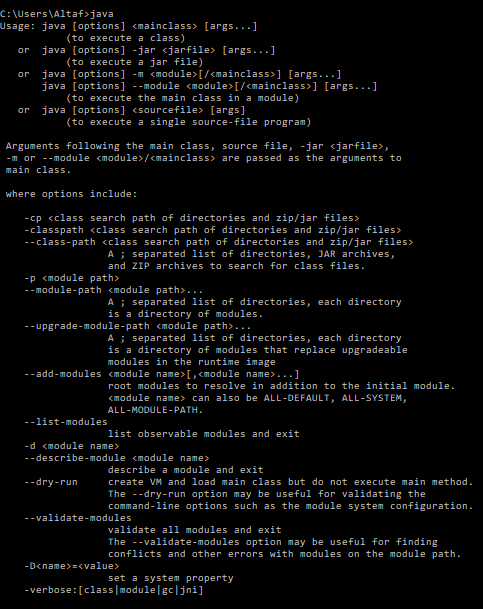
1. Select **Control Panel** and then **System**.
2. Click **Advanced** and then **Environment Variables**.
3. Add the location of the bin folder of the JDK installation to the PATH variable in **System Variables**.

4.     The following is a typical value for the PATH variable:

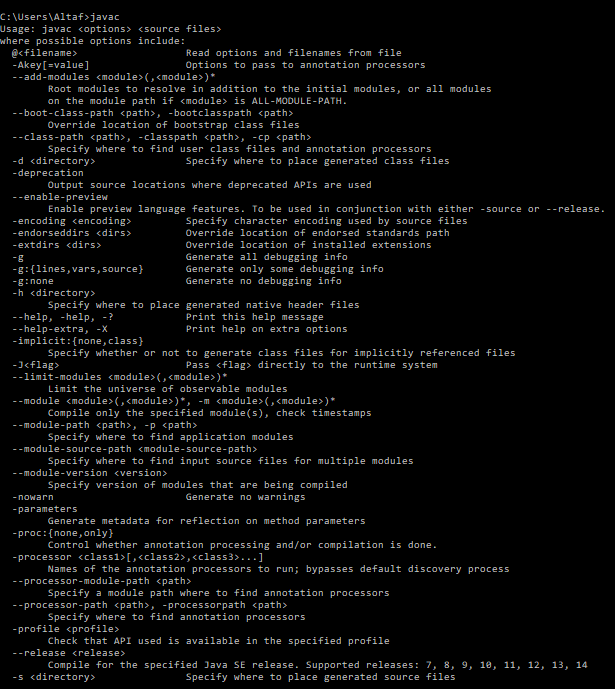
C:\WINDOWS\system32;C:\WINDOWS;"C:\Program Files\Java\jdk-10\bin"

To Verify if JDK and JRE is installed or not run javac and java commands in the command prompt, if commands are not recognized check the installation steps again.

**Output for java command:**



**Output for javac command:**



//a Java Hello World program\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 26/09/2020

\*Program name : HelloWorld.java

\*Topics : print statement

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

class HelloWorld{

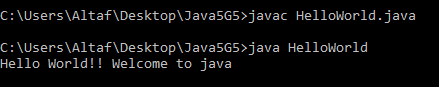
public static void main(String[] args){

System.out.println("Hello World!! Welcome to java");

}

}

**Output:**



1. **Problem Description**

Given the meal price (base cost of a meal), tip percent (the percentage of the meal price being added as tip), and tax percent (the percentage of the meal price being added as tax) for a meal, find and print the meal's total cost.

Note: Be sure to use precise values for your calculations, or you may end up with an incorrectly rounded result!

**Input Format**

There are 3 lines of numeric input:

The first line has a double, mealCost (the cost of the meal before tax and tip).

The second line has an integer,tipPercent (the percentage of mealCost being added as tip).

The third line has an integer, taxPercent (the percentage of mealCost being added as tax).

**Output Format**

Print the total meal cost, where totalCost is the rounded integer result of the entire bill (mealCost with added tax and tip).

Example:

Given:

mealCost = 45 , tipPercent = 20 , taxPercent = 8

Calculations:

tip = 45 X 20/100 = 9

tax = 45 X 8/100 = 3.6

totalCost = mealCost + tip + tax = 45 + 9 + 3.6 = 57.6

round(totalCost) = 57

***Test Cases:***

**Test Case 1:**

***Input :***

Enter the details : 45 20 8

***Output :***

1. Total cost of the meal : 57

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//A Java program to demonstrate a given problem by implementing variables, data types, operators and control statements

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 27/09/2020

\*Program name : Practice7.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.Scanner;

class Meal{

//Properties of meal object(adding variables)

double baseCost;

double tipPercent;

double taxPercent;

double tip;

double tax;

double totalCost;

Scanner console = new Scanner(System.in);

//Addinig methods to a meal class

void read(){

System.out.print("Enter the Base Cost of meal: ");

baseCost=console.nextDouble();

System.out.print("Tip Percent: ");

tipPercent=console.nextDouble();

System.out.print("Tax Percent: ");

taxPercent=console.nextDouble();

}

void calBill(){

tip = baseCost\*(tipPercent/100);

tax = baseCost\*(taxPercent/100);

totalCost = baseCost+tip+tax;

}

void display(){

System.out.println("\nMeal base cost: "+baseCost);

System.out.println("Tip Percentage: "+tipPercent);

System.out.println("tax Percentage: "+taxPercent);

System.out.println("Tip Cost: "+tip);

System.out.println("Tax Cost: "+tax);

System.out.println("Total cost of meal: "+(int)totalCost);

}

}

public class Practice7{

public static void main(String...args){

Meal m1= new Meal();

m1.read();

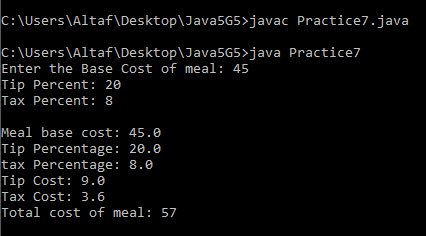
m1.calBill();

m1.display();

}

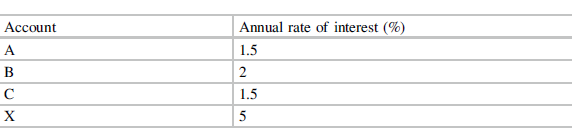
}

**Output:**



1. **Problem Description:**

Consider a bank that offers four different types of account (‘A’, ‘B’, ‘C’ and ‘X’). The following table illustrates the annual rate of interest offered for each type of account.



Design and implement a program that allows the user to enter an amount of money and a type of bank account, before displaying the amount of money that can be earned in one year as interest on that money for the given type of bank account.

You should use the switch statement when implementing this program. Hint: be careful to consider the case of the letters representing the bank accounts. You might want to restrict this to, say, just upper case. Or you could enhance your program by allowing the user to enter either lower case or upper case letters.

**Writing space for post lab:***(For Student’s use only)*

**Program:**

//A Java program that allows the user to enter an amount of money and a type of bank account and display the amount of money that can be earned in one year

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : In Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 28/09/2020

\*Program name : InterestPractice.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.Scanner;

class Bank{

double amount;

char bank;

double interest;

int flag;

double result;

Scanner console = new Scanner(System.in);

void input(){

System.out.print("Enter the amount of money: ");

amount = console.nextDouble();

System.out.print("Enter the type of Bank account: ");

bank = console.next().charAt(0);

}

void bankCheck(char bank){

switch(bank){

case 'A':

case 'a':

interest = 1.5;

break;

case 'B':

case 'b':

interest = 2;

break;

case 'C':

case 'c':

interest = 1.5;

break;

case 'X':

case 'x':

interest = 5;

break;

default:

System.out.println("Invalid Account type");

flag=1;

}

}

void calculator(){

result = amount \*interest/100;

}

void display(){

System.out.println("\nAmount: "+amount);

System.out.println("Bank account type: "+bank);

System.out.println("Amount of money earned as interest in one year: "+result);

}

}

public class InterestPractice{

public static void main(String args[]){

Bank b1 = new Bank();

b1.input();

b1.bankCheck(b1.bank);

if(b1.flag==1){

} else{

b1.calculator();

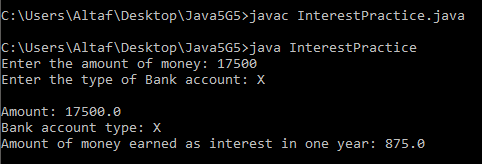
b1.display();

}

}

}

**Output:**



1. **Problem Description:**

Consider a vending machine that offers the following options:

[1] Get gum

[2] Get chocolate

[3] Get popcorn

[4] Get juice

[5] Display total sold so far

[6] Quit

Design and implement a program that continuously allows users to select from these options. When options 1–4 are selected an appropriate message is to be displayed acknowledging their choice. For example, when option 3 is selected the following message could be displayed:

Here is your popcorn

When option 5 is selected, the number of each type of item sold is displayed. For example:

3 items of gum sold

2 items of chocolate sold

6 items of popcorn sold

9 items of juice sold

When option 6 is chosen the program terminates. If an option other than 1–6 is entered an appropriate error message should be displayed, such as:

Error, options 1-6 only!

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//A Java program that continuously allows users to select from the displayed options and perform the actions accordingly

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : In Lab

\*Problem No. : 04

\*Programmer : Afrose Hussain

\*Date : 29/09/2020

\*Program name : Practice11.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.Scanner;

class VendingMachine{

byte gum;

byte chocolate;

byte popcorn;

byte juice;

int option;

Scanner console = new Scanner(System.in);

void process(){

while(option!=6){

System.out.print("\nEnter your choice: ");

option = console.nextInt();

switch(option){

case 1:

gum++;

System.out.println("Here is your gum");

break;

case 2:

chocolate++;

System.out.println("Here is your chocolate");

break;

case 3:

popcorn++;

System.out.println("Here is your popcorn");

break;

case 4:

juice++;

System.out.println("Here is your juice");

break;

case 5:

display();

break;

case 6:

break;

default:

System.out.println("Error, options 1-6 only!");

}

}

}

void display(){

System.out.println("\n"+gum+" items of gum sold");

System.out.println(chocolate+" items of chocolate sold");

System.out.println(popcorn+" items of popcorn sold");

System.out.println(juice+" items of juice sold");

}

}

public class Practice11{

public static void main(String...args){

VendingMachine v1 = new VendingMachine();

System.out.println("-----Vending Machine Options-----");

System.out.println("[1] Get gum");

System.out.println("[2] Get chocolate");

System.out.println("[3] Get popcorn");

System.out.println("[4] Get juice");

System.out.println("[5] Display total sold so far");

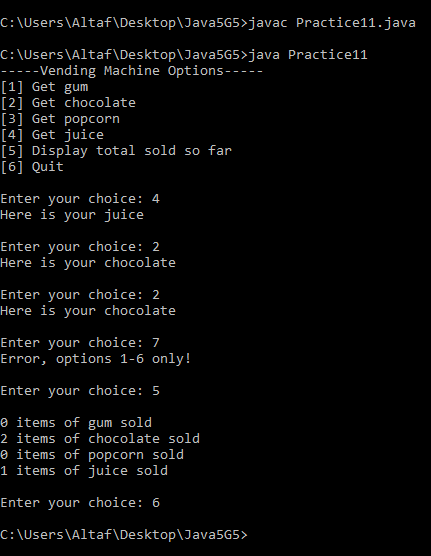
System.out.println("[6] Quit");

v1.process();

}

}

**Output:**



**Post Lab Task:**

1. **Problem Description:**

Develop a text-based game that asks the user to guess a number! The system then outputs whether the guess was right or wrong! Let’s make the program give some help to the user! i.e. output whether the guess was higher or lower than the secret number!

// Hint :Use (Math.random() \* ((max - min) + 1)) + min from lang package

|  |  |
| --- | --- |
| Number  Entered | Message to Display |
| 3 | Too Low |
| 5 | You Win! |
| 7 | Too High |
| 8 | Too High |



**Test Cases:**

**Test Case 1:**

**Input**

Enter the number you guess : 3

**Output :**

The number is too Low

**Test Case 2:**

**Input :**

Enter the number you guess : 5

**Output :**

You Win!

**Writing space for post lab:***(For Student’s use only)*

**Program:**

//A Java program to demonstrate a given problem of guessing a number by implementing variables, data types, operators and control statements

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 30/09/2020

\*Program name : Practice8.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements, Escape sequences

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class GuessNumber{

//Adding a method to determine the guess number with secret number

public String determineGuess(int userNumber,int secretNumber,int count){

if(userNumber<0 || userNumber>100){

return "\nYour guess is invalid!";

} else if(userNumber == secretNumber){

return "\nCorrect!! You won \nTotal Guesses: "+count;

} else if(userNumber<secretNumber){

return "\nYour guess is Too Low, try again.\nTry number: "+count;

} else if(userNumber>secretNumber){

return "\nYour guess is Too High, try again.\nTry number: "+count;

} else{

return "\nYour guess is incorrect.\nTry number: "+count;

}

}

}

public class Practice8{

public static void main(String...args){

Scanner console = new Scanner(System.in);

int secretNumber = (int)(Math.random()\*100)+1; //random number from 1 to 100

int userNumber=0;

GuessNumber g1 = new GuessNumber();

int count =1;

while(userNumber!=secretNumber){

System.out.print("\nEnter your guess between 1 and 100: ");

userNumber = console.nextInt();

String result = g1.determineGuess(userNumber, secretNumber, count);

System.out.println(result);

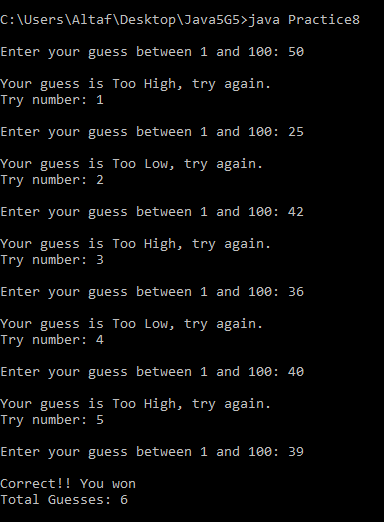
count++;

}

}

}

**Output:**



1. **Problem Description:**

A plumber opens a savings account with $100 000 at the beginning of January. He then makes a deposit of $1000 at the end of each month for the next 12 months (starting at the end of January). Interest is calculated and added to his account at the end of each month (before the $1000 deposit is made). The monthly interest rate depends on the amount A in his account at the time when interest is calculated, in the following way:

A ≤110000 : 1 per cent

110000 < A ≤125000 : 1.5 per cent

A >125000 : 2 per cent

Write a program which displays, for each of the 12 months, under suitable headings, the situation at the end of the month as follows: the number of the month, the interest rate, the amount of interest and the new balance. (Answer: values in the last row of output should be 12, 0.02, 2534.58, and 130263.78).

**Writing space for post lab:***(For Student’s use only)*

**Program:**

//A Java program which displays, for each of the 12 months, under suitable headings, the situation at the end of the month as : the number of the month, the interest rate, the amount of interest and the new balance

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : Post Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 01/10/2020

\*Program name : PlumberSavings.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements, Escape sequences

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

class BankActivity{

double amount = 100000;

int month;

double interestRate;

double interest;

void calculation(){

for(month=1;month<=12;month++){

if(amount<=110000){

interestRate=0.01;

} else if(amount>110000 && amount<=125000){

interestRate=0.015;

} else if(amount>125000){

interestRate=0.02;

} else{

System.out.println("Invalid");

}

interest = amount\*interestRate;

amount = amount+interest;

if(month!=12){ System.out.println(month+"\t"+interestRate+"\t\t"+String.format("%.2f",interest)+"\t\t\t"+String.format("%.2f",amount));

} else{

System.out.println(month+"\t"+interestRate+"\t\t"+String.format("%.2f",interest)+"\t\t\t"+String.format("%.2f",amount+1000));

}

amount = amount + 1000;

}

}

}

public class PlumberSavings{

public static void main(String args[]){

BankActivity b1 = new BankActivity();

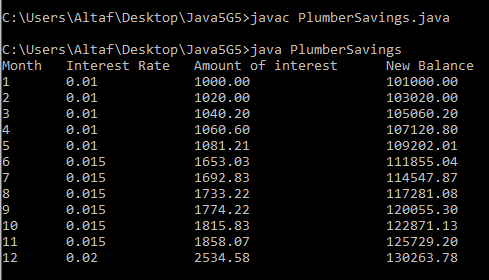
System.out.println("Month\tInterest Rate\tAmount of interest\tNew Balance");

b1.calculation();

}

}

**Output:**



1. **Problem Description:**

If you invest $1000 for one year at an interest rate of 12 per cent, the return is $1120 at the end of the year. But if interest is compounded at the rate of 1 per cent monthly (i.e. 1/12 of the annual rate), you get slightly more interest because it is compounded. Write a program which uses a forloop to compute the balance after a year of compounding interest in this way. The answer should be $1126.83. Evaluate the formula for this result separately as a check:

1000 ×1*.*0112.

**Writing space for post lab:***(For Student’s use only)*

**Program:**

//A Java program which uses a for loop to compute the balance after a year of compounding interest

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 01

\*Type of lab : Post Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 02/10/2020

\*Program name : CompInterest.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements, Escape sequences

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

class Invest{

double amount = 1000;

double interest;

void calculation(){

int month=1;

for(month=1;month<=12;month++){

interest = amount\*0.01;

amount = amount+interest;

}

System.out.println("\nThe balance after a year of compounding interest is $"+String.format("%.2f",amount));

}

}

public class CompInterest{

public static void main(String...args){

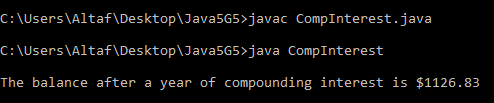
Invest i1 = new Invest();

i1.calculation();

}

}

**Output:**



Students Signature

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  |

*(For Evaluator’s use only)*

# 

**Date of the Session:** 17**/**10**/**2020 **Time of the Session: \_\_\_\_\_to\_\_\_\_\_\_**

# Lab Session 02: Implementation the concepts of Classes and Objects

**Program Title:** Implementation of classes and objects

**Pre Lab Task:**

Answer the following question before entering into lab. The following pre lab tasks have to be performed at home.

1. Consider the following class:

public class IdentifyMyParts

{

public static int x = 7;

public int y = 3;

}

* 1. What are the class variables?
  2. What are the instance variables?

**Ans:-** (a) class / static variables: x

(b) instance variables: y

1. What is the output from the following code:

IdentifyMyParts a = new IdentifyMyParts();

IdentifyMyParts b = new IdentifyMyParts();

a.y = 5;

b.y = 6;

a.x = 1;

b.x = 2;

System.out.println("a.y = " + a.y);

System.out.println("b.y = " + b.y);

System.out.println("a.x = " + a.x);

System.out.println("b.x = " + b.x);

System.out.println("IdentifyMyParts.x = " + IdentifyMyParts.x);

**Ans:-** a.y = 5

b.y = 6

a.x = 2

b.x = 2

IdentifyMyParts.x = 2

1. What's wrong with the following program and explain procedure to solve this error?

public class SomethingIsWrong {

public static void main(String[] args) {

Rectangle myRect;

myRect.width = 40;

myRect.height = 50;

System.out.println("myRect's area is " + myRect.area());

}

}

**Ans:-** In this program, there is no Rectangle class. And also, myRect is just a reference variable and any object of Rectangle class is not created and the instance variables width, height and instance method area are not created. The above code should be modified as below:

import java.lang.\*;

class Rectangle{

int width;

int height;

int area(){

return width\*height;

}

}

public class SomethingIsWrong {

public static void main(String[] args) {

Rectangle myRect = new Rectangle();

myRect.width = 40;

myRect.height = 50;

System.out.println("myRect's area is " + myRect.area());

}

}

**In labTask:***(For Student’s use only)*

1. **Problem Description**

Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables‐a part number (type String),a part description(type String),a quantity of the item being purchased (type int) and a price per item (double). Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. In addition, provide a method named getInvoice Amount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0.

Write a test application named InvoiceTest that demonstrates class Invoice’s capabilities

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//A java program that demonstrates the use of constructors, classes and objects with an invoice example

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 02

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 17/10/2020

\*Program name : InvoiceTest.java

\*Topics : Implementation of classes and objects

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class Invoice{

String partNumber;

String partDescription;

int quantity;

double price;

Scanner console = new Scanner(System.in);

Invoice(){

partNumber = null;

partDescription = null;

quantity = 0;

price = 0.0;

}

void get(){

System.out.print("Enter Part Number: ");

partNumber = console.next();

System.out.print("Enter Part Description: ");

partDescription = console.next();

System.out.print("Enter quantity: ");

quantity = console.nextInt();

System.out.print("Enter price: ");

price = console.nextDouble();

}

void set(){

if(quantity<0){

quantity=0;

}

if(price<0){

price=0.0;

}

}

double getInvoice(){

double totalCost = quantity\*price;

return totalCost;

}

void display(){

System.out.println("\nPart Number: "+partNumber);

System.out.println("Part Description: "+partDescription);

System.out.println("Price: "+price);

System.out.println("Quantity: "+quantity);

System.out.println("Total invoice amount: "+getInvoice());

}

}

public class InvoiceTest{

public static void main(String...args){

Invoice i1 = new Invoice();

i1.get();

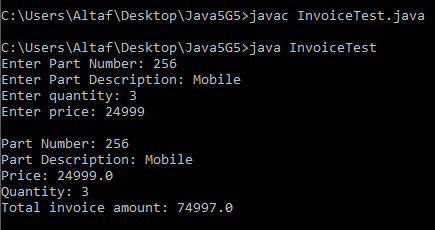
i1.set();

i1.display();

}

}

**Output:**



1. **Problem Description**

Create a class called Book to represent a book. A Book should include four pieces of information as instance variables‐a book name, an ISBN number, an author name and a publisher. Your class should have a constructor that initializes the four instance variables. Provide a mutator method and accessor method (query method) for each instance variable. In addition, provide a method named getBookInfo that returns the description of the book as a String (the description should include all the information about the book). You should use this keyword in member methods and constructor**.**

Write a test application named BookTest to create an array of object for 30 elements for class Book to demonstrate the class Book's capabilities.

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//A Java program to create an array of object for 30 elements for class Book to demonstrate the class Book's capabilities.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 02

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 18/10/2020

\*Program name : BookTest.java

\*Topics : Fundamentals of java programming

Variables, Data types, operators & Control Statements, arrays

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class Book{

String bookName;

int ISBN;

String author;

String publisher;

Scanner sc = new Scanner(System.in);

//no parameterised constructor

public Book(){

bookName="unknown";

ISBN=0;

author="unknown";

publisher="unknown";

}

public String getBookName(){

System.out.print("Enter Book Name: ");

String book = sc.nextLine();

return book;

}

public int getISBN(){

System.out.print("Enter ISBN Number: ");

int isbn = sc.nextInt();

return isbn;

}

public String getAuthorName(){

System.out.print("Enter Author Name: ");

String authorName = sc.nextLine();

return authorName;

}

public String getPublisherName(){

System.out.print("Enter Publisher Name: ");

String publisherName = sc.nextLine();

return publisherName;

}

public void setBookName(String bookName){

this.bookName = bookName;

}

public void setISBN(int ISBN){

this.ISBN = ISBN;

}

public void setAuthorName(String author){

this.author = author;

}

public void setPublisherName(String publisher){

this.publisher = publisher;

}

public String getBookInfo(){

return (bookName+", which has the ISBN number "+ISBN+" was written by "+author+" and published by "+publisher);

}

}

public class BookTest{

public static void main(String args[]){

Book[] b = new Book[30];

b[0] = new Book();

b[0].setBookName(b[0].getBookName());

b[0].setAuthorName(b[0].getAuthorName());

b[0].setPublisherName(b[0].getPublisherName());

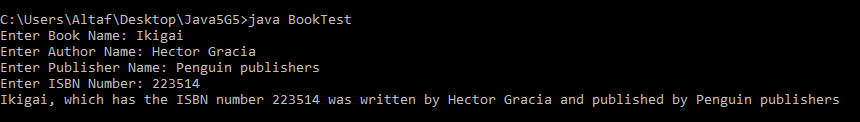
b[0].setISBN(b[0].getISBN());

System.out.println(b[0].getBookInfo());

}

}

**Output:**



1. **Problem Description:**

Define a class Customer in java having following specifications:

**Instance attributes :**

* + - customernumber - numeric value
    - customername - string value
    - price, qty, discount, totalprice, netprice - numeric value

**Instance methods :**

init() to assign initial values of customernumber as 101, customername as “Pratap”,qty as 0 and price, discount &netprice as 0.

caldiscount ( ) – To calculate discount, totalprice and netprice

totalprice = price \* qty

discount is 25% of totalprice, if totalprice>=50000

discount 15% of totalprice, if totalprice>=25000 and totalprice<50000

discount 10% of totalprice, if totalprice<250000

netprice= totalprice - discount

input() – to read data members customername, customernumbar, price, qty and

call caldiscount() to calculate discount, totalprice and netprice.

show( ) – to display Customer details.

**Test Cases:**

**Test Case 1:**

**Input**

Enter the Details:

112 Pratap 4 12000 13 11000

**Output**

The total bill has to be paid is

112 Pratap 41760

**Writing space for In lab work: (For Student’s use only)**

**Program:**

//A java program to calculate the net price of a quantity of items after availing discount

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 02

\*Type of lab : In Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 19/10/2020

\*Program name : Practice9.java

\*Topics : Implementation of classes and objects

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class Customer{

String customerName;

int customerNumber;

double price;

int quantity;

double discount;

double totalPrice;

double netPrice;

Scanner console = new Scanner(System.in);

void init(){

customerNumber = 101;

customerName = "Afrose";

quantity = 0;

price = 0;

totalPrice = 0;

netPrice = 0;

}

void input(){

System.out.print("Enter the customer name: ");

customerName = console.next();

System.out.print("Enter the customer number: ");

customerNumber = console.nextInt();

System.out.print("Enter price: ");

price = console.nextDouble();

System.out.print("Enter quantity: ");

quantity = console.nextInt();

}

void calDiscount(){

totalPrice = price \* quantity;

if(totalPrice>=50000){

discount=25;

} else if(totalPrice>=25000 && totalPrice<50000){

discount = 15;

} else if(totalPrice<25000){

discount=10;

} else{

System.out.println("Total Price is out of range");

}

netPrice = totalPrice - (discount\*totalPrice)/100;

}

void show(){

System.out.println("\nCustomer Name: "+customerName);

System.out.println("Customer Number: "+customerNumber);

System.out.println("Price of each quantity: "+price);

System.out.println("Quantity: "+quantity);

System.out.println("Total Price: "+totalPrice);

System.out.println("Discount: "+discount+"%");

System.out.println("Net Price: "+netPrice);

}

}

public class Practice9{

public static void main(String...args){

Customer c1 = new Customer();

c1.init();

c1.input();

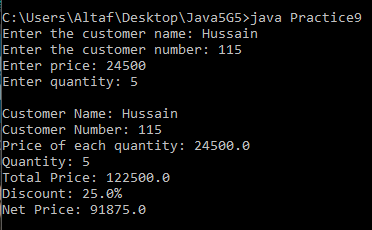
c1.calDiscount();

c1.show();

}

}

**Output:**



**Post Lab Task:**

1. **Problem Description**

A program in a class CountPoor that counts the number of families that are considered to be poor. Write and use a class Family that has the attributes:

* income - a double value that is the income for the family
* size - the number of people in the family and the following methods:
  + Family(income , size ) - a constructor that sets the attributes
  + isPoor(housingCost ,foodCost) - a method that returns true if housingCost + foodCost \* size is greater than half the family income ( foodCost is the average food cost for an individual , while housingCost is for the family )
  + toString() - a method that returns a string containing the information about the family

The program should read an integer k from the keyboard and then create an array of size k whose base type is Family. It should then create k objects whose base type isFamily. It should then create k objects of type Family and put them in the array, reading the income and size for each family from the keyboard. After reading an average housing cost and average food cost from the keyboard, it should display the families are the poor.

**Test Cases:**

**Test Case 1:**

**Input**

Enter the Details:

150000 120 4

**Output**

The Family is Middle class family

**Test Case 2:**

**Input**

Enter the Details:

15000 40 6

**Output**

The Family is a Poor family

**Writing space for Post lab work:** *(For Student’s use only)*

**Program**:

//A java program that counts the number of poor families from a given array of Families

/\*\*\*\*\*\*

\*Lab session number : 02

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 20/10/2020

\*Program name : Practice1.java

\*Topics : Implementation of classes,arrays & objects

\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class Family{

double income;

int size;

double housingCost;

double foodCost;

public Family(double income, int size){

this.income = income;

this.size = size;

}

public void getCosts(double housingCost, double foodCost){

this.housingCost = housingCost;

this.foodCost = foodCost;

}

public boolean isPoor(){

return (housingCost + (foodCost \* size) > (income / 2));

}

public String toString(){

return ("income : " + income + "\nsize: " + size + "\nHousing Cost:"

+ housingCost + "\nFood Cost: " + foodCost);

}

}

class CountPoor{

public static int count(Family[] families){

int poorFamilies = 0;

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

if(families[i].isPoor()){

poorFamilies++;

}

}

return poorFamilies;

}

}

public class Practice1{

public static void main(String...args){

int numOfFamilies;

double income, housingCost, foodCost;

int size;

System.out.print("Enter number of families: ");

Scanner console = new Scanner(System.in);

numOfFamilies = console.nextInt();

//creating array of objects

Family[] families = new Family[numOfFamilies];

for(int i = 0; i < numOfFamilies; i++){

System.out.print("Enter income: ");

income = console.nextDouble();

System.out.print("Enter num of members in family: ");

size = console.nextInt();

families[i] = new Family(income, size);

System.out.print("Enter Housing Cost: ");

housingCost = console.nextDouble();

System.out.print("Enter avg food cost per person: ");

foodCost = console.nextDouble();

families[i].getCosts(housingCost, foodCost);

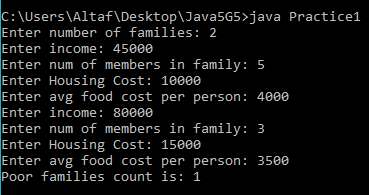
}

System.out.println("Poor families count is: " + CountPoor.count(families));

}

}

**Output**:



1. **Problem Description**

Create a class called Employee that includes three pieces of information as instance variables:

* a first name (typeString),
* a last name (typeString)
* a monthly salary (double).
* Your class should have a constructor that initializes the three instance variables. Provide a set and a get method for each instance variable. If the monthly salary is not positive, set it to 0.0.
* Write a test application named EmployeeTest that demonstrates class Employee’s capabilities.
* Create two Employee objects and display each object’s yearly salary. Then give each Employee a 10% raise and display each Employee’s yearly salary again**.**

**Writing space for Post lab work:** *(For Student’s use only)*

**Program:**

//A java program that demonstrates the use of constructors, classes and objects with an employee example

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 02

\*Type of lab : Post Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 21/10/2020

\*Program name : EmployeeTest.java

\*Topics : Implementation of classes and objects

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import required packages

import java.lang.\*;

import java.util.\*;

class Employee{

String firstName;

String lastName;

double monthlySalary;

Scanner console = new Scanner(System.in);

Employee(){

firstName = null;

lastName = null;

monthlySalary = 0.0;

}

void get(){

System.out.print("\nEnter First Name: ");

firstName = console.next();

System.out.print("Enter Last Name: ");

lastName = console.next();

System.out.print("Enter salary: ");

monthlySalary = console.nextDouble();

}

void set(){

if(monthlySalary<0){

monthlySalary=0;

}

}

double raisedSalary(){

double raisedSalary=monthlySalary\*12;

double increment=raisedSalary\*0.1;

raisedSalary=raisedSalary+increment;

return raisedSalary;

}

void display(){

System.out.println("\nEmployee name: "+firstName+" "+lastName);

System.out.println("Yearly salary: "+monthlySalary\*12);

System.out.println("Yearly salary after 10% raise: "+raisedSalary());

}

}

public class EmployeeTest{

public static void main(String args[]){

Employee e1 = new Employee();

Employee e2 = new Employee();

e1.get();

e1.set();

e2.get();

e2.set();

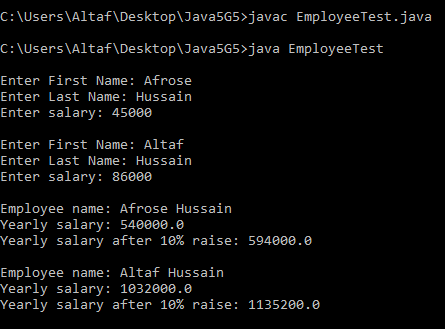
e1.display();

e2.display();

}

}

**Output:**



Students Signature

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  |

*(For Evaluator’s use only)*

**Date of the Session:** 24**/**10**/**2020 **Time of the Session:** 11:20AM **to** 1:00PM

# Lab Session 03: Implementation of Arrays

**Program Title:** Implementation of Arrays

**Pre Lab Task:** Answer the following questions before entering into lab.

1. Consider the following explicit creation of an array:

**int[] someArray = {2,5,1,9,11};**

1. What would be the value of someArray.length?

A. The value would be 5.

1. What is the value of someArray[2]?

A. It is 1.

1. What would happen if you tried to access someArray[6]?

A. Array Index out of bound exception.

1. Create the equivalent array by using the new operator and then assigning the value of each element individually.

A. int[] someArray = new int[5];

someArray[0] = 2;

someArray[1] = 5;

someArray[2] = 1;

someArray[3] = 9;

someArray[4] = 11;

1. Write a standard for loop that will double the value of every item in someArray.

A. for(int i = 0; i < someArray.length; i++)

someArray[i] = 2 \* someArray[i];

1. Can we change the size of an array at run time?

A. No, we can’t change the size of array during run time, it is fixed.

1. Can you explain different steps of declaring multidimensional arrays in Java?

A. Two dimensional array:

int[][] twoD\_arr = new int[10][20];

Three dimensional array:

int[][][] threeD\_arr = new int[10][20][30];

1. There are 2 int type array data type. One is containing 50 elements, and another one is containing 30 elements. Can we assign the array of 50 elements to an array of 30 elements?

A. Yes, we can assign provided they should the same type. The compiler will check the only type of the array, not the size. But only the first 30 elements will be assigned.

1. What is the meaning of anonymous array? Explain with an example?

A. An array in Java without any name is anonymous array. It is an array just for creating and using instantly.

class Test {

public static void main(String[] args)

{

// anonymous array

sum(new int[]{ 1, 2, 3 }); //in this way we declare and use an anonymous array.

}

public static void sum(int[] a)

{

int total = 0;

// using for-each loop

for (int i : a)

total = total + i;

System.out.println("The sum is:" + total);

}

}

**In lab Task:***(For Student’s use only)*

1. **Problem Description:**

A history professor has so many students in her class that she has trouble determining how well the class does on exams. She has discovered that you are a computer whiz and has asked you to write an application to perform some simple statistical analyses on exam scores. Your application must work for any class size up to 100.

Write and test a computer application that does the following:

a. Reads the test grades.

b. Calculates the class mean, standard deviation (defined in Programming Problem 2), and percentage of the test scores falling in the ranges < 10, 10–19, 20–29, 30–39, . . . , 80–89, and ≥ 90.

c. Prints a summary showing the mean, the standard deviation, and a histogram of the percentage distribution of test scores.

**Writing space for In lab work:** *(For Student’s use only)*

**Program**:

//a java program to implement arrays

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 03

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 24/10/2020

\*Program name : HistoryStudents.java

\*Topics : Creating arrays, accessing elements in arrays

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class History{

double mean, stand;

int a,b,c,d,e,f,g,h,i,j;

void mean(int[] students, int n){

for(int k=0;k<n;k++){

mean=mean+students[k];

}

mean=mean/n;

}

void sd(int[] students, int n){

for(int k=0;k<n;k++){

stand=stand+((students[k]-mean)\*(students[k]-mean));

}

stand=stand/n;

stand=Math.sqrt(stand);

}

void histogram(int[] students, int n){

for(int k=0;k<n;k++){

if(students[k]<10){

a++;

} else if(students[k]<20){

b++;

} else if(students[k]<30){

c++;

} else if(students[k]<40){

d++;

} else if(students[k]<50){

e++;

} else if(students[k]<60){

f++;

} else if(students[k]<70){

g++;

} else if(students[k]<80){

h++;

} else if(students[k]<90){

i++;

} else if(students[k]<=100){

j++;

} else{

System.out.println("Invalid output");

}

}

}

void display(){

System.out.println("Mean: "+String.format("%.2f",mean));

System.out.println("Standard Deviation: "+String.format("%.2f",stand));

System.out.println("0-9:\t"+a);

System.out.println("10-19:\t"+b);

System.out.println("20-29:\t"+c);

System.out.println("30-39:\t"+d);

System.out.println("40-49:\t"+e);

System.out.println("50-59:\t"+f);

System.out.println("60-69:\t"+g);

System.out.println("70-79:\t"+h);

System.out.println("80-89:\t"+i);

System.out.println("90-100:\t"+j);

}

}

public class HistoryStudents{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter the number of students: ");

int num = sc.nextInt();

int[] students = new int[100];

for(int i=0;i<num;i++){

System.out.print("Enter "+(i+1)+"th student's marks: ");

students[i] = sc.nextInt();

}

History h = new History();

h.mean(students,num);

h.sd(students,num);

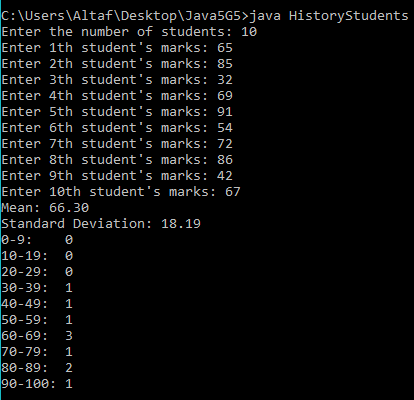
h.histogram(students,num);

h.display();

}

}

**Output**:



1. **Problem Description:**

A Java program to find the frequency of each element in the array. In this program, we have an array of elements to count the occurrence of its each element. One of the approaches to resolve this problem is to maintain one array to store the counts of each element of the array. Loop through the array and count the occurrence of each element as frequency and store it in another array fr.

1 2 8 3 2 2 2 5 1

In the given array, 1 has appeared two times, so its frequency is 2, and 2 has appeared four times so have frequency 4 and so on.

**Test Cases:**

**Test Case 1:**

**Input**

1 1 3 3 2 2 4 2 4 4 4 5 6 2

**Output**

1 frequency 2

2 frequencies 4

3 frequency 1

4 frequencies 4

5 frequency 1

6 frequency 1

**Writing space for In lab:***(For Student’s use only)*

**Program:**

// A java program to count the frequency of elements in the given array

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 03

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 25/10/2020

\*Program name : FrequencyCalculator.java

\*Topics : creating arrays, accessing elements in arrays

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

class Frequency{

public static int[][] freq(int[] A){

int[][] freq = new int[A.length][2];

int k = 0;

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

if(A[i] == Integer.MIN\_VALUE){

continue;

}

freq[k][0] = A[i];

freq[k][1] = 1;

for(int j = i + 1; j < A.length; j++){

if(A[i] == A[j]){

A[j] = Integer.MIN\_VALUE;

freq[k][0] = A[i];

freq[k][1] += 1;

}

}

k++;

}

return Arrays.copyOfRange(freq, 0, k);

}

public static void display(int[][] A){

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

System.out.println(A[i][0] + " Frequency is " + A[i][1]);

}

}

}

public class FrequencyCalculator{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter elements of array: ");

String data = sc.nextLine();

String[] d = data.split(" ");

int[] A = Arrays.stream(d).mapToInt(Integer::parseInt).toArray();

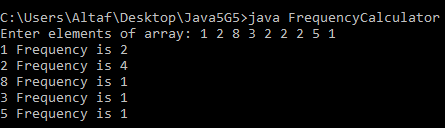
int[][] freq = Frequency.freq(A);

Frequency.display(freq);

}

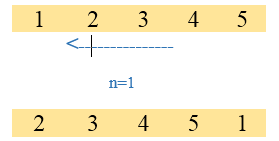
}

**Output**:



1. **Problem Description:**

A Java program to left rotate the elements of an array. In this program, we need to rotate the elements of an array towards the left by the specified number of times. In the left rotation, each element of the array will be shifted to its left by one position and the first element of the array will be added to end of the list. This process will be followed for a specified number of times.



Consider above array, if n is 1 then, all elements of the array will be moved to its left by one position such that second element of the array will take the first position, the third element will be moved to the second position and so on. The first element of the array will be added to the last of the array.

**Test Cases:**

**Test Case 1:**

**Input**

1

1 2 3 4 5

**Output:**

2 3 4 5 1

**Test Case 2:**

**Input**

3

1 2 3 4 5

**Output:**

4 5 1 2 3

**Writing space for In lab:***(For Student’s use only*

**Program:**

// A java program to left shift an array

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 03

\*Type of lab : In Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 26/10/2020

\*Program name : ArrayRotation.java

\*Topics : Creating arrays, accessing elements in arrays

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

class LeftShift{

public static void leftShift(int[] A, int shiftBy){

int[] temp = new int[shiftBy];

//elements to be added at last are stored in a temporary array

for(int i = 0; i < shiftBy; i++){

temp[i] = A[i];

}

//shifting elements after given number to front

int len = A.length;

int j = 0;

for(int i = shiftBy; i < len; i++){

A[j] = A[i];

j++;

}

//adding elements at end

int k = 0;

for(int i = j; i < len; i++){

A[i] = temp[k];

k++;

}

}

public static void display(int[] A){

System.out.print("Rotated Array: ");

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

System.out.print(A[i] + " ");

}

}

}

public class ArrayRotation{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter elements of array: ");

String data = sc.nextLine();

System.out.print("Enter number of elements to rotate: ");

int shiftBy = sc.nextInt();

String[] d = data.split(" ");

int[] A = Arrays.stream(d).mapToInt(Integer::parseInt).toArray();

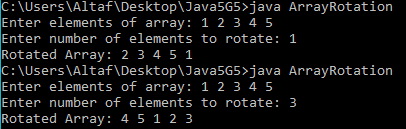
LeftShift.leftShift(A, shiftBy);

LeftShift.display(A);

}

}

**Output:**



**Post Lab Task:**

**Answer the following questions before leaving from lab:**

1. **Problem Description**

Many of the patterns we have seen for traversing arrays can also be written recursively. It is not common, but it is a useful exercise.

1. Write a method called maxInRange that takes an array of integers and two indexes, lowIndex and highIndex, and \_nds the maximum value in the array, but only considering the elements between lowIndex and highIndex, including both. This method should be recursive. If the length of the range is 1, that is, if lowIndex == highIndex, we know immediately that the sole element in the range must be the maximum. So that's the base case.If there is more than one element in the range, we can break the array into two pieces, find the maximum in each of the pieces, and then find the maximum of the maxima.
2. Methods like maxInRange can be awkward to use. To find the largest element in an array, we have to provide the range for the entire array. double max = maxInRange(a, 0, a.length - 1); Write a method called max that takes an array and uses maxInRange to \_nd and return the largest element.

**Program**:

// A java program to find maximum element in array using recursion

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 03

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 27/10/2020

\*Program name : RecursiveMax.java

\*Topics : creating arrays, accessing elements in arrays

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

class ArrayFunc{

int maxInRange(int A[],int low,int high){

if(low==high){

return A[low];

}

int mid = (low+high)/2;

int a = maxInRange(A,low,mid);

int b = maxInRange(A,mid+1,high);

if(a>b){

return a;

} else if(a<b){

return b;

} else{

return a;

}

}

int max(int A[]){

return maxInRange(A, 0, A.length - 1);

}

}

public class RecursiveMax{

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

System.out.print("Enter elements of array: ");

String data = sc.nextLine();

String[] d = data.split(" ");

int[] A = Arrays.stream(d).mapToInt(Integer::parseInt).toArray();

System.out.println("1 - Max In range\n2 - Max in array");

int opt = sc.nextInt();

ArrayFunc af = new ArrayFunc();

if(opt == 1){

System.out.print("Enter lower index value: ");

int lowIndex = sc.nextInt();

System.out.print("Enter higher index value: ");

int highIndex = sc.nextInt();

System.out.println("The Maximum element in the given range is:"+af.maxInRange(A,lowIndex,highIndex));

}

if(opt == 2){

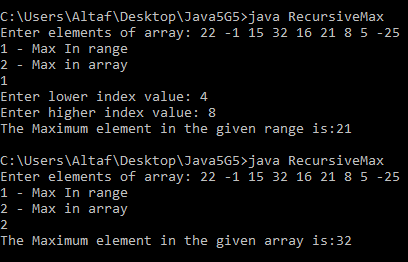
System.out.println("The Maximum element in the given array is:" + af.max(A));

}

}

}

**Output**:



2. Trace out the output of this program?

class evaluate

{

public static void main(String args[])

{

int arr[] = new int[] {0 , 1, 2, 3, 4, 5, 6, 7, 8, 9};

int n = 6;

n = arr[arr[n] / 2];

System.out.println(arr[n] / 2);

}

}

A. Output of this program is 1,

class evaluate

{

public static void main(String args[])

{

int arr[] = new int[] {0 , 1, 2, 3, 4, 5, 6, 7, 8, 9};

int n = 6;

n = arr[arr[n] / 2]; //arr[n] = 6, arr[n]/2 = 3, arr[3] = 3 => n = 3

System.out.println(arr[n] / 2); // arr[3] = 3, 3/2 = 1, Output is 1.

}

}

Students Signature

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  |

*(For Evaluator’s use only)*

**Date of the Session:** 07 **/**11 **/**2020 **Time of the Session:** 11:20AM **to** 1:00PM

# Lab Session 04 Use String and String Tokenizer classes and develop a java application

**Title of the Program:** Implementation of Inheritance types.

**Pre Lab Task:**

**Answer the following question before entering into lab**

1. Observe the following code and Answer Questions from A to C

class String {

java.lang.String s;

public String(java.lang.String s) {

this.s = s;

}

public java.lang.String toString() {

return s;

}

}

1. Where the control goes after executing return statement?

Ans: Control goes to the line of the program after function call, generally it is assigning the string object to a reference variable.

1. Describe the role of this keyword in above code.

Ans: As the name of local variable and instance variable is same , to differentiate between them ‘this’ keyword is used.’this.s’ represents the instance variable ‘s’ and ‘s’ represents the local variable. The function of that statement is the value of the local variable assigned to the instance variable .

1. Explain string representation object returned by tostring() in the above code?

Ans: String class is in lang package of java library.

1. Here's a puzzler: normally, the statement x++ is exactly equivalent to x = x + 1. But if x is a char, it's not exactly the same! In that case, x++ is legal, but x = x + 1 causes an error. Try it out and see what the error message is, then see if you can figure out what is going on.

Ans: The error message is,

error: incompatible types: possible lossy conversion from int to char

x=x+1;

In case of x++, automatic type conversion occurs. So, the corresponding character of the value just after x based on the ASCII value will be returned. But in the case of x=x+1, type conversion does not occur. So , according to that statement, we are trying to add an ‘int’ value to a ‘char’ value .

1. Why the string immutable in java?

Ans: Strings are immutable because:

1)String uses a string constant pool.

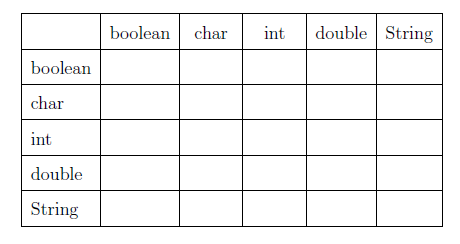
2)Strings are used between multiple clients.

3)Strings are frequently used in hashing.

4)Strings are thread safe synchronized.

5)Strings may be misused that leads to threats.

1. Make a bigger copy of the following table and fill it in. At the intersection of each pair of types, you should indicate whether it is legal to use the + operator with these types, what operation is performed (addition or concatenation), and what the type of the result is.



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Data Type | boolean | char | int | double | String |
| boolean | Error | Error | Error | Error | Concatenation |
| char | Error | Adds ASCII value of given characters(result is the integer) | Adds the ASCII value of given character and given integer(result is integer) | Adds ASCII value of character and double number(result is double type) | Concatenation |
| int | Error | Adds the ASCII value of given character and given integer(result is integer) | Addition of integers takes place | Adds integer value and double value and gives double type as result | Concatenation |
| double | Error | Adds ASCII value of character and double number(result is double type) | Adds integer value and double value and gives double type as result | Addition of double numbers takes place | Concatenation |
| String | Concatenation | Concatenation | Concatenation | Concatenation | Concatenation |

**In Lab Tasks:**

1. **Problem Description**

Write a Java program to make a new string with each character of just before and after of t-string whichever it appears in m-string. Assume that m-string and non-empty t-string has given

**Test Case1: Input:**  The given string are: weablcoabmeabandab

Output: The new string is: el

**Test Case2: Input:**  The given string are: welcome Hai

Output: The new string is: Null

**Writing space for In lab:***(For Student’s use only)*

**Program:**

/\*a Java program to make a new string with each character of just before and after of t-string whichever it appears in m-string\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 04

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 07/11/2020

\*Program name : Practice13.java

\*Topics : String, BufferString

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class PatternString{

String m\_string;

String t\_string;

String result="";

//Adding parameterised constructor

public PatternString(String m, String t){

m\_string = m;

t\_string = t;

}

//Adding method

public void mAndTstring(){

int m\_len = m\_string.length();

int t\_len = t\_string.length();

for(int i=0;i<m\_len-t\_len+1;i++){

String temp= m\_string.substring(i,i+t\_len);

if(i>=0 && temp.equals(t\_string)){

result+=m\_string.substring(i-1,i);

}

if(i<m\_len-t\_len && temp.equals(t\_string)){

result+=m\_string.substring(i+t\_len,i+t\_len+1);

}

}

}

public void display(){

System.out.println("The given m\_string is: "+m\_string);

System.out.println("The given t\_string is: "+t\_string);

System.out.println("The output string is: "+result);

}

}

//Adding Driver class

public class Practice13{

public static void main(String...args){

String m\_string = "weablcoabmeabandab";

String t\_string = "ab";

PatternString ps = new PatternString(m\_string,t\_string);

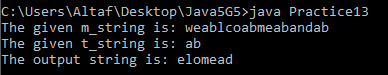
ps.mAndTstring();

ps.display();

}

}

**Output:**



1. **Problem Description**

Consider a program to enter and confirm a suitable code name for an agent. Declare two string objects, called codeNameandconfirm and then

1. Prompt to get the user to enter a suitable name into the codeNamestring;
2. Use a while loop to ensure that the string entered is greater than 6 characters in length, if it is not print “INVALID CODENAME” and ask the user to re-enter a code name;
3. Once a valid code name has been entered ask the user to re-enter the code name into the confirm string and then use an if else statement to ensure that the string entered matches the original code name; if it does, print a message “CODE NAME CONFIRMED” otherwise print a message saying “CODE NAME MIS-MATCH”;
4. Use the charAtmethod to ensure that the code name ends with an ‘X’ character;

(e) Finally use the startsWithmethod to ensure that, as well as being greater than 6 characters in length, the code name entered also starts with the words “Agent”.

**Writing space for In lab:***(For Student’s use only)*

**Program:**

/\*a Java program to enter and confirm a suitable code name for an agent\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 04

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 08/11/2020

\*Program name : CodeChecker.java

\*Topics : String, BufferString,String methods

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class Agent{

String codeName;

String confirmName;

Scanner sc=new Scanner(System.in);

public void getCode(){

while(true){

System.out.print("Enter the code: ");

codeName = sc.nextLine();

int i=0;

for(char c: codeName.toCharArray()) {

i++;

}

if(i<=6){

System.out.println("INVALID CODENAME");

} else{

break;

}

}

}

public void confirm(){

while(true){

System.out.print("Re-enter the code: ");

confirmName = sc.nextLine();

if(codeName.equals(confirmName)){

System.out.println("CODE NAME CONFIRMED");

break;

} else{

System.out.println("CODE NAME MIS-MATCH");

}

}

}

public void checkConditions(){

int len = codeName.length();

if(codeName.charAt(len-1)=='X' && codeName.startsWith("Agent") && len>6){

System.out.println("The given code is Suitable for an Agent");

} else{

System.out.println("The given code is not Suitable for an Agent");

}

}

}

public class CodeChecker{

public static void main(String...args){

Agent a1 = new Agent();

a1.getCode();

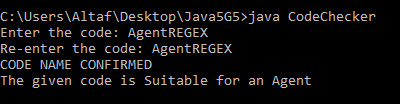
a1.confirm();

a1.checkConditions();

}

}

**Output:**



1. **Problem Description**

A website requires the users to input username and password to register. Write a program to check the validity of password input by users.

Following are the criteria for checking the password:

1. At least 1 letter between [a-z]
2. At least 1 number between [0-9]
3. At least 1 letter between [A-Z]
4. At least 1 character from [$#@]
5. Minimum length of transaction password: 6
6. Maximum length of transaction password: 12

Your program should accept a sequence of comma separated passwords and will check them according to the above criteria. Passwords that match the criteria are to be printed, each separated by a comma.

**Test Case1: Input**  ABd1234@1, a F1#, 2w3E\*, 2We3345

**Output**  ABd1234@1

**Test Case 2: Input** Welcome

**Output**  Invalid password

**Writing space for In lab:***(For Student’s use only*

**Program:**

/\*a Java program to check the validity of password input by users\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 04

\*Type of lab : In Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 09/11/2020

\*Program name : PasswordChecker.java

\*Topics : String, BufferString,String methods

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class Validator{

String password;

String[] arrOfPass;

Scanner sc = new Scanner(System.in);

public void getter(){

System.out.print("Enter password: ");

password = sc.nextLine();

}

public void checker(){

arrOfPass = password.split(", ");

int count=0;

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

if(arrOfPass[i].matches(".\*[a-z]{1,}.\*") && arrOfPass[i].matches(".\*[A-Z]{1,}.\*") && arrOfPass[i].matches(".\*[0-9]{1,}.\*") && arrOfPass[i].matches(".\*[$#@]{1,}.\*") && arrOfPass[i].length()>=6 && arrOfPass[i].length()<=12){

System.out.println(arrOfPass[i]);

count++;

}

if(count==0){

System.out.println("Invalid password");

}

}

}

}

public class PasswordChecker{

public static void main(String...args){

Validator v1 = new Validator();

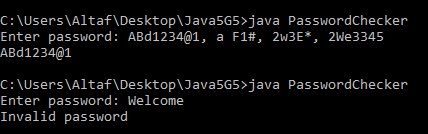
v1.getter();

v1.checker();

}

}

**Output:**



**Post lab task:**

1. **Problem Description**

Write a Java program to check whether a string is pq-balanced or not. A String is pq-balanced if for all the p's in the string at least one 'q' must exists right of the p's. But 'q' before the 'p' makes the pq-balanced false.

**Test Case 1: Input** The given strings is: gfpmpnppqab

**Output** The string is pq-balanced? True

**Test Case 2: Input**  The given strings is: gfpmpnpqpab

**Output** The string is pq-balanced? False

**Program:**

//a Java program to check whether a string is pq-balanced or not. A String is pq-balanced if for all the p's in the string at least one 'q' must exists right of the p's. But 'q' before the 'p' makes the pq-balanced false\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 04

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 10/11/2020

\*Program name : PQBalanced.java

\*Topics : String, BufferString,String methods

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class PQString{

String input;

Scanner sc = new Scanner(System.in);

public void getInput(){

System.out.print("The given string is: ");

input = sc.nextLine();

}

public void balanceChecker(){

int index\_p =-1;

int index\_q = -1;

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

if(input.charAt(i)=='q'){

index\_q = i;

break;

}

}

for(int i=input.length()-1;i>=0;i--){

if(input.charAt(i)=='p'){

index\_p = i;

break;

}

}

System.out.print("The string is pq-balanced? ");

System.out.println(index\_p<index\_q?"True":"False");

}

}

public class PQBalanced{

public static void main(String...args){

PQString p1 = new PQString();

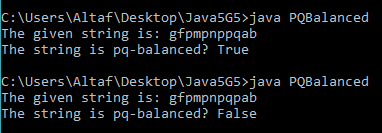
p1.getInput();

p1.balanceChecker();

}

}

**Output:**



1. **Problem Description**

Write a Java program to read a string and if a substring of length two appears at both its beginning and end, return a string without the substring at the beginning otherwise, return the original string unchanged.

**Test Case 1: Input** The given strings is: aamadamaa

**Output**  madamaa

**Test Case 2: Input** The given strings is: welcome

**Output**  welcome

**Program:**

/\*a Java program to read a string and if a substring of length two appears at both its beginning and end, return a string without the substring at the beginning otherwise, return the original string unchanged\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 04

\*Type of lab : Post Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 11/11/2020

\*Program name : SameSubstring.java

\*Topics : String, BufferString,String methods

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class SubString{

String input;

Scanner sc = new Scanner(System.in);

public void getInput(){

System.out.print("The given string is: ");

input = sc.nextLine();

}

public void subStringChecker(){

int l = 0;

int r = input.length();

if(input.charAt(0)==input.charAt(input.length()-2) && input.charAt(1)==input.charAt(input.length()-1)){

l=2;

}

System.out.println(input.substring(l,r));

}

}

public class SameSubstring {

public static void main(String args[]){

SubString s1 = new SubString();

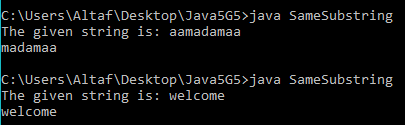
s1.getInput();

s1.subStringChecker();

}

}

**Output:**



Students Signature

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  |

*(For Evaluator’s use only)*

# Lab Session 05: Implementation of Inheritance and types of inheritance

**Date of the Session:** 23**/**12**/**2020 **Time of the Session:** 11:20AM **to** 1:00PM

**Title of the Program:** Implementation of Inheritance types.

**Pre Lab Task:**

**Answer the following question before entering into lab**

1. **Observe the following code and Answer Questions from A to C**

class A1

{

{

System.out.println(1);

}

}

class B extends A1

{

{

System.out.println(2);

}

}

class C extends B

{

{

System.out.println(3);

}

}

public class A

{

public static void main(String[] args)

{

C c = new C();

}

}

1. **Describe what type of inheritance it is?**

A. Multilevel inheritance is described in the above block of code.

1. **Which blocks are invoked when object of class c is created?**

A. All the IIBs in class C and all it's parent classes are invoked,(i.e) all IIBs in A1, B, C classes.

1. **Constructor are not defined any one of the above classes. But we are instantiating the object of class C. What will happen in this scenario?**

A. Super class constructor with no parameters will be invoked(ultimately object class constructor will be invoked)

1. **Private members of a class are inherited to sub class. True or false?**

**A.** No private members cannot be inherited by sub class, they can only be accessed in the class defined.

1. **What is wrong with the following code and how to correct it .**

public class A

{

public A()

{

System.out.println(1);

super();

System.out.println(2);

}

}

A. super() call should always be the first statement of a constructor. In any other situation, the program leads to a compile time error.

public class A{

public A(){

super();

System.out.println(1);

System.out.println(2);

}

}

1. **Which class is default super class for every class in java?**

A. Object class is the default super class for every class in java.

**In Lab Tasks:**

1. **Problem Description**

Create a class called “Automobile” which holds the following data members:

Make

Year/ Model (two pieces to be stored as one string)

Mileage

Price

Write an automobile base class with cars, Trucks and Vans as child classes, complete with parameter constructors which pass the parents member variables to the child class

**Test Cases 1:Input:**  { “Marithi”, “2019-Swift”, 23.5, 100000}

**Output:** Make=Maruthi

Year/Model=2019-Swift

Milage=23.5

Price=100000

**Test Case 2**:  **Input:** { “Hundai”,”i10-2020”, 20000, }

**Output:** Invalid Input parameter for Year/Model [since first four characters in the range of 2000-2030]

**Writing space for In lab:***(For Student’s use only)*

**Program:**

/\*A An automobile base class with cars, trucks as child classes, complete with parameters constructors which pass the parents member variables to the child class\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 05

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 23/12/2020

\*Program name : VehicleInheritance.java

\*Topics : Variables, Data types, Conrol statements, operators

Classes, Objects, constructors, inheritance & its types

this and super keyword Method Overriding

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class Automobile{

String make = "unknown";

String yr\_model = "unknown";

double mileage = 0.0;

double price = 0.0;

public Automobile(String m,String y\_m,double mil,double p){

make = m;

yr\_model = y\_m;

mileage = mil;

price = p;

}

public void display(){

System.out.println("Make = " + make);

System.out.println("Year/Model = " + yr\_model);

System.out.println("Mileage = " + mileage);

System.out.println("Price = "+ price);

}

}

class Car extends Automobile{

public Car(String m,String y\_m,double mil,double p){

super(m, y\_m, mil, p);

}

}

class Truck extends Automobile{

public Truck(String m,String y\_m,double mil,double p){

super(m, y\_m, mil, p);

}

}

class Van extends Automobile{

public Van(String m,String y\_m,double mil,double p){

super(m, y\_m, mil, p);

}

}

public class VehicleInheritance{

public static void main(String[] args) {

System.out.print("Enter Details: ");

Scanner sc = new Scanner(System.in);

String data = sc.nextLine();

String[] s = data.split(",");

if(s.length != 4){

System.out.println("invalid arguments");

System.exit(0);

}

int year = 0;

try{

year = Integer.parseInt(s[1].substring(0, 4));

}catch(NumberFormatException e){

System.out.println("Invalid Input parameter for Year/Model");

System.exit(1);

}

if(!(year >= 2000 && year <= 2030)){

System.out.println("Invalid Input parameter for Year/Model");

//System.out.println(year);

System.exit(0);

}

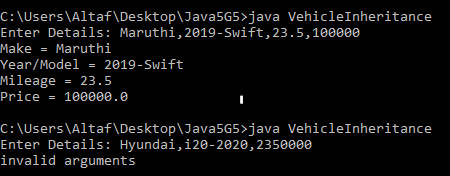
Car c = new Car(s[0], s[1], Double.parseDouble(s[2]), Double.parseDouble(s[3]));

c.display();

}

}

**Output:**



1. **Problem Description**

All the banks operating in India are controlled by RBI. RBI has set a well-defined guideline (e.g. minimum interest rate, minimum balance allowed, maximum withdrawal limit etc) which all banks must follow. For example, suppose RBI has set minimum interest rate applicable to a saving bank account to be 4% annually; however, banks are free to use 4% interest rate or to set any rates above it.

Write a Java program to implement bank functionality in the above scenario and demonstrate the dynamic polymorphism concept. Note: Create few classes namely Customer, Account, RBI (Base Class) and few derived classes (SBI, ICICI, PNB etc). Assume and implement required member variables and functions in each class.

**Test Cases 1**:  **Input**: Bank\_Name=”SBH”

deposit\_amount=10000

No\_of\_Months=4

Withdrawal\_amount=0

**Output:** 10400

**Test Cases2**:  **Input**: Bank\_Name=”ICICI”

deposit\_amount=10000

No\_of\_Months=0

Withdrawal\_amount=1000

**Output:** “maintain Minimum balance”

**Writing space for In lab:***(For Student’s use only)*

**Program:**

/\*a Java program to implement bank functionality in the above scenario and demonstrate the dynamic polymorphism concept\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 05

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 24/12/2020

\*Program name : BankInheritance.java

\*Topics : Variables, Data types, Conrol statements, operators

Classes, Objects, constructors, inheritance & its types

this and super keyword, dynamic polymorphism, Method Overriding

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class RBI{

public double minInt, minBal, maxWit;

public double bal;

public void setBal(double b){

bal = b;

}

void calInterest(int months){

double intAmt = (bal \* minInt \* ((double)months / 12.0)) / 100.0;

bal = bal + intAmt;

}

public void withdrawl(double amt){

if(amt > maxWit){

System.out.println("Maximum withdrawl is " + maxWit +", You cannot draw amount more than that");

return;

}

if(bal - amt < minBal){

System.out.println("Maintain minimum balance");

}else{

bal = bal - amt;

System.out.println("Remaining Balance is " + String.format("%.2f",bal));

}

}

}

class SBI extends RBI{

public SBI(double minInt, double minBal, double maxWit){

super.minInt=minInt;

super.minBal=minBal;

super.maxWit=maxWit;

}

}

class ICICI extends RBI{

public ICICI(double minInt, double minBal, double maxWit){

super.minInt=minInt;

super.minBal=minBal;

super.maxWit=maxWit;

}

}

class PNB extends RBI{

public PNB(double minInt, double minBal, double maxWit){

super.minInt=minInt;

super.minBal=minBal;

super.maxWit=maxWit;

}

}

class Customer{

String name;

int id;

public Customer(String n, int num, String accType, double deposit\_amt, int months, double withdrawl\_amt){

name = n;

id = num;

Account a = new Account(accType, deposit\_amt, months, withdrawl\_amt);

}

}

class Account{

public double deposit\_amt, withdrawl\_amt;

public int months;

public String accType;

public Account(String accType, double deposit\_amt, int months, double withdrawl\_amt){

this.accType = accType;

this.deposit\_amt = deposit\_amt;

this.months = months;

this.withdrawl\_amt = withdrawl\_amt;

if(accType.equals("SBI")){

SBI s = new SBI(5, 2000, 7500);

s.setBal(deposit\_amt);

s.calInterest(months);

s.withdrawl(withdrawl\_amt);

}

if(accType.equals("ICICI")){

ICICI i = new ICICI(6, 2500, 7500);

i.setBal(deposit\_amt);

i.calInterest(months);

i.withdrawl(withdrawl\_amt);

}

if(accType.equals("PNB")){

PNB p = new PNB(7, 3000, 7500);

p.setBal(deposit\_amt);

p.calInterest(months);

p.withdrawl(withdrawl\_amt);

}

}

}

public class BankInheritance{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

String bankName, name;

int id, months;

double deposit\_amt, withdrawl\_amt;

System.out.print("Enter Customer Name: ");

name = sc.nextLine();

System.out.print("Enter Bank Name: ");

bankName = sc.nextLine();

System.out.print("Enter Customer Id: ");

id = sc.nextInt();

System.out.print("Enter Deposit Amount: ");

deposit\_amt = sc.nextDouble();

System.out.print("Enter number of months: ");

months = sc.nextInt();

System.out.print("Enter withdrawl amount: ");

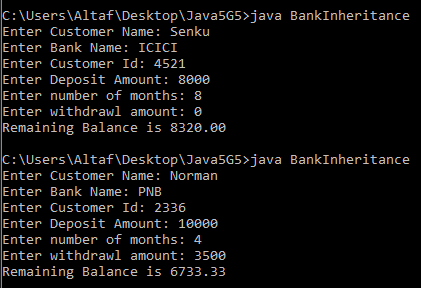
withdrawl\_amt = sc.nextDouble();

Customer c = new Customer(name, id, bankName, deposit\_amt, months, withdrawl\_amt);

}

}

**Output:**



**Post lab task:**

1. **Problem Description**

Create a class 'Degree' having a method 'getDegree' that prints "I got a degree". It has two subclasses namely 'Undergraduate' and 'Postgraduate' each having a method with the same name that prints "I am an Undergraduate" and "I am a Postgraduate" respectively. Call the method by creating an object of each of the three classes.

**Test Cases 1**:  **Input**: call print method using each class object

**Output**: ‘Get Degree’

‘Undergraduate’

‘Postgraduate’

**Test Case 2: Input** Call parent class print method using any one of the child class object

**Output:** ‘Get Degree’

‘Postgraduate’

**Test Case 3: Input** Call any child t class print method using parent class object

**Output:** Exception/ Error

**Writing space for Post Lab:***(For Student’s use only)*

**Program**:

/\*A java program to test various scenarios of inheritance\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 05

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose

\*Date : 26/12/2020

\*Program name : PracticeDegree.java

\*Topics : Variables, Data types, Conrol statements, operators

Classes, Objects, constructors, inheritance & its types

this and super keyword Method Overriding

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class Degree{

public void getDegree(){

System.out.println("I got a degree");

}

}

class Undergraduate extends Degree{

public void degreeName(){

System.out.println("I am a Undergraduate");

}

}

class Postgraduate extends Degree{

public void degreeName(){

System.out.println("I am a Postgraduate");

}

}

public class PracticeDegree{

public static void main(String...args){

Degree d1=new Degree();

Undergraduate ug=new Undergraduate();

Postgraduate pg=new Postgraduate();

ug.degreeName();

pg.degreeName();

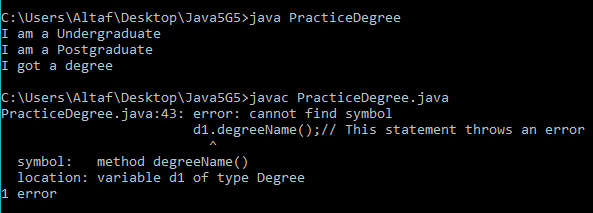
ug.getDegree();

d1.degreeName();// This statement throws an error

}

}

**Output**:



1. **Problem Description**

Create a class 'Student' with three data members which are name, age and address. The constructor of the class assigns default values name as "unknown", age as '0' and address as "not available". It has two members with the same name 'setInfo'. First method has two parameters for name and age and assigns the same whereas the second method takes has three parameters which are assigned to name, age and address respectively. Print the name, age and address of 3 students.

Hint - Use array of objects

**Test Case 1: Input** {‘jhon’, 10, ‘Vijayawada’, ’San’ ,20, ’Guntur’, ’Ram’,30, ’Hyderabad’}

**Output:** Student1.name=’john’

Student1.age=10

Student1.address=’Vijayawada’

Student2.name=’San’

Student2.age=20

Student2.address=’Guntur’

Student3.name=’Ram’

Student3.age=30

Student3.address=’Hyderabad’

**Test Case 2: Input** {‘venkat’, 90, ‘Vijayawada’, ’Rahim’ ,20, ’Guntur’, ’Dileep’,30, ’Hyderabad’,’Venu’,40,’Vizag’}

**Output:** Array index Out of Bound Exception

**Writing space for Post lab:***(For Student’s use only)*

**Program**:

//a java program to create and print array of student objects

/\*\*\*\*\*\*

\*Lab session number : 05

\*Type of lab : Post Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 27/12/2020

\*Program name : ArrayStudents.java

\*Topics : Variables, Data types, Conrol statements, operators

Classes, Objects, constructors, inheritance & its type, this and super keyword,

Method Overriding

\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

class Student{

String name;

String address;

int age;

public Student(){

name = "unknown";

age = 0;

address = "Not Available";

}

public void setInfo(String name,int age, String address){

this.name = name;

this.age = age;

this.address = address;

}

public void setInfo(String name,int age){

this.name = name;

this.age = age;

}

public void display(){

System.out.println("Name: " + name);

System.out.println("Age: " + age);

System.out.println("Address: " + address);

}

}

public class ArrayStudents{

public static boolean onlyDigits(String s){

return (s.matches("[0-9]+"));

}

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter Student Details:");

String data = sc.nextLine();

Student[] s = new Student[3];

String[] details = data.split(",");

if (details.length > 9){

System.out.println("Array index Out of Bound Exception");

System.exit(1);

}

else if (details.length == 9){

int j = 0;

for(int i=0;i<3;i++){

s[i]=new Student();

s[i].setInfo(details[j], Integer.parseInt(details[j + 1]), details[j + 2]);

j=j+3;

s[i].display();

}

}else{

if(details.length < 6){

System.out.println("Array index Out of Bound Exception");

System.exit(1);

}

int[] check = {0, 0, 0};

if(onlyDigits(details[3])){

check[0] = 1;

}

if(onlyDigits(details[details.length - 1])){

check[2] = 1;

}

if((check[0] == 1 && onlyDigits(details[5])) || (check[0] == 0 && onlyDigits(details[6]))){

check[1] = 1;

}

int j = 0;

for(int i=0; i < 3; i++){

s[i]=new Student();

if (check[i] == 1){

s[i].setInfo(details[j], Integer.parseInt(details[j + 1]));

j=j+2;

s[i].display();

continue;

}

s[i].setInfo(details[j], Integer.parseInt(details[j + 1]), details[j + 2]);

j=j+3;

s[i].display();

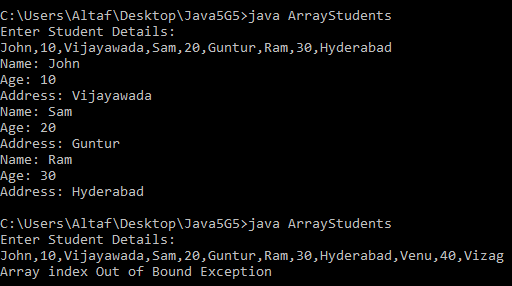
}

}

}

}

**Output**:



Students Signature

*(For Evaluator’s use only)*

|  |  |  |
| --- | --- | --- |
|  | |  |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  |

# Lab Session 06: Use interfaces and develop a java application

**Date of the Session:** 30**/**12**/**2020 **Time of the Session:** 11:20AM **to** 1:00PM

**Program Title:** Apply interfaces to a given application.

**Pre Lab Task:**

1. Can there be an abstract class with no abstract methods in it?

A. Yes, there can be abstract class with no abstract methods in it, If we want a class to not create objects we can do that.

1. What is the error in the below code and write the correct instruction?

interface A

{

private int i;

}

A. Members of interface cannot be private.

interface A{

int i;

}

1. Can we put a static method in interfaces?

A. Yes, we can have static methods interface, we can access it using interface name.

1. How do you access interface field ‘i’ in the below code?

class P

{

interface Q

{

int i = 111;

}

}

A. We can access the ‘i’ in class P as Q.i and outside the class P as P.Q.i

**In Lab task:**

1. **Problem Description**

Create an interface “Number” with the following abstract methods isZero( ), isPositive(), isNegative( ), isOdd( ), isEven( ), isPrime(), isAmstrong() the above methods return boolean primitive type. Implement this interface in “Verification” class.

**Test Case 1: Input** 151

**Output** positive and prime

**Test Case 1: Input -**153

**Output** negtive and Armstrong

**Writing space for In lab:***(For Student’s use only)*

**Program**:

//a java program to implement interfaces

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 06

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 30/12/2020

\*Program name : TestInterface.java

\*Topics : Implementation of Interfaces, interface, implements, defining an interface

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

//Creating iterface

interface Number{

public abstract boolean isZero();

public abstract boolean isPositive();

public abstract boolean isNegative();

public abstract boolean isOdd();

public abstract boolean isEven();

public abstract boolean isPrime();

public abstract boolean isAmstrong();

}

class Verification implements Number{

int number;

public Verification(int number){

this.number=number;

}

public boolean isZero(){

if(number==0){

return true;

} else{

return false;

}

}

public boolean isPositive(){

if(number>0){

return true;

} else{

return false;

}

}

public boolean isNegative(){

if(number<0){

return true;

} else{

return false;

}

}

public boolean isEven(){

if(number%2==0){

return true;

} else{

return false;

}

}

public boolean isOdd(){

if(number%2!=0){

return true;

} else{

return false;

}

}

public boolean isPrime(){

if(number>0){

for(int i=2;i<number/2;i++){

if(number%i==0){

return false;

}

}

return true;

} else{

for(int i=2;i<-number/2;i++){

if(-number%i==0){

return false;

}

}

return true;

}

}

public boolean isAmstrong(){

int d,sum=0,n;

n=number;

if(n>0){

while(n>0){

d=n%10;

sum=sum+(d\*d\*d);

n=n/10;

}

if(sum==number){

return true;

} else{

return false;

}

} else{

n=-n;

while(n>0){

d=n%10;

sum=sum+(d\*d\*d);

n=n/10;

}

if(sum==-number){

return true;

} else{

return false;

}

}

}

}

class TestInterface{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter a number: ");

int number = sc.nextInt();

Verification v = new Verification(number);

boolean zero = v.isZero();

boolean positive = v.isPositive();

boolean negative = v.isNegative();

boolean odd = v.isOdd();

boolean even = v.isEven();

boolean prime = v.isPrime();

boolean amstrong = v.isAmstrong();

if(zero){

System.out.println("It is zero");

} else if(positive){

if(even){

if(prime){

if(amstrong){

System.out.println("It is a positive, even, prime, amstrong number");

} else{

System.out.println("It is a positive, even, prime number");

}

}else if(amstrong){

System.out.println("It is a positive, even, amstrong number");

} else{

System.out.println("It is a positive, even number");

}

} else{

if(prime){

if(amstrong){

System.out.println("It is a positive, odd, prime, amstrong number");

} else{

System.out.println("It is a positive, odd, prime number");

}

}else if(amstrong){

System.out.println("It is a positive, odd, amstrong number");

} else{

System.out.println("It is a positive, odd number");

}

}

} else if(negative){

if(even){

if(prime){

if(amstrong){

System.out.println("It is a negative, even, prime, amstrong number");

} else{

System.out.println("It is a negative, even, prime number");

}

}else if(amstrong){

System.out.println("It is a negative, even, amstrong number");

} else{

System.out.println("It is a negative, even number");

}

} else{

if(prime){

if(amstrong){

System.out.println("It is a negative, odd, prime, amstrong number");

} else{

System.out.println("It is a negative, odd, prime number");

}

}else if(amstrong){

System.out.println("It is a negative, odd, amstrong number");

} else{

System.out.println("It is a negative, odd number");

}

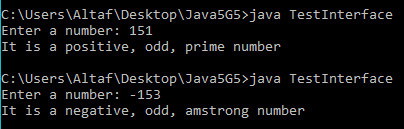
}

}

}

}

**Output**:



1. **Problem Description**

Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt in this class use the object of arithmetic class.

**Test Case 1: Input** 1424

**Output:** 2027776

**Test Case 2: Input** -12345

**Output:** 152399025

**Writing space for In lab:***(For Student’s use only)*

**Program**:

//a java program to implement interfaces

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 06

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 31/12/2020

\*Program name : ToTestInt.java

\*Topics : Implementation of Interfaces, interface, implements, defining an interface

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

interface Test{

double square(double a);

}

class Arithmetic implements Test{

public double square(double a){

return a \* a;

}

}

public class ToTestInt {

public static void main(String...args){

Scanner sc = new Scanner(System.in);

Arithmetic a = new Arithmetic();

System.out.print("Enter a Number: ");

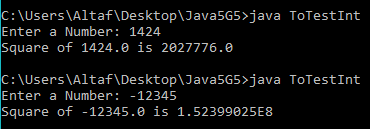
double n = sc.nextDouble();

System.out.println("Square of " + n + " is " + a.square(n));

}

}

**Output**:



1. **Problem Description**

Write an interface named “EmployeeDetails”which consist of following abstract methods 1. Enter Data 2. Display Data 3. Exit .Implement this interface in “Userselection” class to invoke the respective method according to the given menu input.

**Test Case 1: Input** {‘Varun’, 1245, 60000}

**Output:** Employee Name= ‘Varun’

Employee Id= 1245

Employee Salary= 60000

**Test Case 2: Input** {‘John. 1245’, 12345}

**Output:** Invalid number of arguments

**Writing space for In lab:***(For Student’s use only)*

**Program**:

//a java program to implement interfaces

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 06

\*Type of lab : In Lab

\*Problem No. : 03

\*Programmer : Afrose Hussain

\*Date : 02/01/2021

\*Program name : EmployeeInterface.java

\*Topics : Implementation of Interfaces, interface, implements, defining an interface

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

interface EmployeeDetails{

void enterDetails(String name, int id, double salary);

void displayData();

void exit();

}

class UserSelection implements EmployeeDetails{

String name;

int id;

double salary;

public void enterDetails(String name, int id, double salary){

this.name = name;

this.id = id;

this.salary = salary;

}

public void displayData(){

System.out.println("Employee Name = " + name);

System.out.println("Employee Id = " + id);

System.out.println("Employee Salary = " + salary);

}

public void exit(){

System.exit(1);

}

}

public class EmployeeInterface{

public static void main(String[] args){

System.out.print("Enter Details: ");

Scanner sc = new Scanner(System.in);

String data = sc.nextLine();

String[] s = data.split(",");

if (s.length != 3){

System.out.println("Invalid number of arguments");

}else{

UserSelection u = new UserSelection();

u.enterDetails(s[0], Integer.parseInt(s[1]), Double.parseDouble(s[2]));

u.displayData();

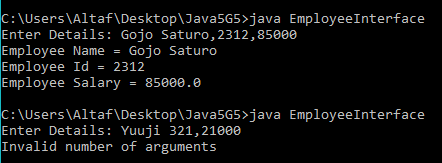
u.exit();

}

}

}

**Output**:



**Post Lab task:**

1. **Problem Description**

Write an interface called Shape3D that supports a method getVolume. Write a class called Cuboid that implements Shape3D and has three private double fields length, height, and breadth. getVolume() should return the volume of the Cuboid object. The constructor for Cuboid should allow the client to create a Cuboid object by specifying the three fields length, height and breadth.

**Test Case 1: Input** 123 345 46

**Output:** 1952010

**Test Case 2: Input** -12345

**Output:** Invalid Number

**Writing space for Post lab:***(For Student’s use only)*

**Program**:

//a java program to implement interfaces

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 06

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 03/01/2021

\*Program name : Geometry3D.java

\*Topics : Implementation of Interfaces, interface, implements, defining an interface

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.Scanner;

interface Shape3D{

double getVolume();

}

class Cuboid implements Shape3D{

private double length, breadth, height;

public Cuboid(double l, double b, double h){

length = l;

breadth = b;

height = h;

}

public double getVolume(){

return length \* breadth \* height;

}

}

public class Geometry3D{

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter dimensions: ");

String data = sc.nextLine();

String[] s = data.split(" ");

if (s.length != 3){

System.out.println("Invalid number");

}else{

Cuboid c = new Cuboid(Integer.parseInt(s[0]), Integer.parseInt(s[1]), Integer.parseInt(s[2]));

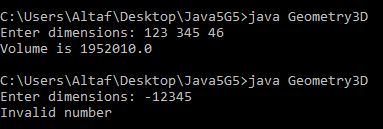
System.out.println("Volume is " + c.getVolume());

}

}

}

**Output**:



Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

# Lab Session 07: Create a package and access members from a package

**Date of the Session:** 06**/**01**/**2021 **Time of the Session:** 11:20AM **to** 1:00PM

**Program Title:** Implementation of packages.

**Pre Lab Task:**

1. Can we use a field or a method declared without access modifiers outside the package?

A. No, we can't use a field or a method with no-access (default) specifiers outside the package in which their class is defined

1. How to set class path for a package?

A. CLASSPATH can be set temporarily for that particular CMD shell session by issuing the following command:

> SET CLASSPATH=.;c:\javaproject\classes;d:\tomcat\lib\servlet-api.jar

• Instead of using the CLASSPATH environment variable, you can also use the command-line option -classpath or -cp of the javac and java commands, for example,

> java –classpath c:\javaproject\classes com.abc.project1.subproject2.MyClass3

1. Can we accesses protected data members of class outside package?

A. Yes, we can access the protected data members of the class outside the package by inheriting the class.

1. Is every class in java is a part of some package? Justify your answer.

A. Yes, every class in java belongs to a class. For all the program we write without creating a specific package, they get considered into an unnamed package whereas all the in-built classes we use like String, Scanner, etc belong to in-built packages like lang, util, etc.

1. Can we hide the class definitions that are used in packages?

A. Classes with default access specifier cannot be accessed outside the package, in this way we can hide the class definitions that are used in the package.

**In Lab task:**

1. Create a package with name placements. It consists of number\_of\_palcementsclass with one method called Read\_data to read year and thenumber of placements in that year. Now import placement package in your programand create an array of objects to store placement data in different year and compute total number of placements in all years.

**Test Case 1: Input** Number of years 3

2019 200

2018 134

2017 234

**Output** Total Number of placements568

**Test Case 2: Input** Number of years 2

2014 200

**Output** Invalid input

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//a Java program to create and access a package

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 07

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 06/01/2021

\*Program name : No\_Of\_Placements.java

\*Topics : Creating a package, accessing a package, adding classes to a package, benefits of package, types of package, setting classpath

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Creating a package

package Placements;

//Accessing a class inside a package

import java.util.Scanner;

public class No\_Of\_Placements{

public String yearPlacements;

public int year;

public int placements;

public void readData(){

try{

Scanner sc= new Scanner(System.in);

System.out.print("Enter year and placements: ");

yearPlacements = sc.nextLine();

String temp1 = yearPlacements.substring(0,4);

year = Integer.parseInt(temp1);

String temp2 = yearPlacements.substring(5);

placements = Integer.parseInt(temp2);

}

catch(Exception e){

System.out.println("Invalid Input");

System.exit(0);

}

}

}

//Access a Package

import Placements.No\_Of\_Placements;

import java.util.Scanner;

public class Practice15{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter number of years: ");

int years = sc.nextInt();

No\_Of\_Placements n[] = new No\_Of\_Placements[years];

for(int i=0;i<years;i++){

n[i] = new No\_Of\_Placements();

n[i].readData();

}

int total=0;

for(int j=0;j<years;j++){

total+=n[j].placements;

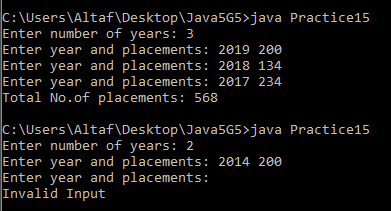
}

System.out.println("Total No.of placements: "+total);

}

}

**Output**:



1. Create a package with name shape. It consists of Triangle, Rectangle, square classes’attributes height and width. Import this package in your program and calculate areaof each shape.

**Test Case 1: Input 20 20**

**Output** area of triangle: 200

area of Rectangle: 400

area of square: 400

**Test Case 2: Input 20 40**

**Output** area of triangle: 400

area of Rectangle: 800

area of square: invalid dimensions

**Writing space for In lab:***(For Student’s use only)*

**Program**:

//a Java program to create and access a package

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 07

\*Type of lab : In Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 07/01/2021

\*Program name : Triangle.java

\*Topics : Creating a package, accessing a package, adding classes to a package, benefits of package, types of package, setting classpath

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Creating a package

package Shape;

//Accessing a class inside a package

import java.util.Scanner;

public class Triangle{

int height, width;

public void area(int height, int width){

this.height=height;

this.width=width;

System.out.println("Area of triangle: "+(height\*width/2));

}

}

//Creating Rectangle class in Shape package

package Shape;

public class Rectangle{

int height, width;

public void area(int height, int width){

this.height=height;

this.width=width;

System.out.println("Area of rectangle: "+(height\*width));

}

}

//Creating Square class in Shape package

package Shape;

public class Square{

int height, width;

public void area(int height, int width){

this.height=height;

this.width=width;

if(height==width){

System.out.println("Area of square: "+(height\*width));

} else{

System.out.println("Area of square: invalid dimensions");

}

}

}

//Geometry.java

//Accessing created package

import Shape.Triangle;

import Shape.Rectangle;

import Shape.Square;

import java.util.\*;

public class Geometry{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter the dimensions: ");

int x = sc.nextInt();

int y = sc.nextInt();

Triangle t1 = new Triangle();

t1.area(x,y);

Rectangle r1 = new Rectangle();

r1.area(x,y);

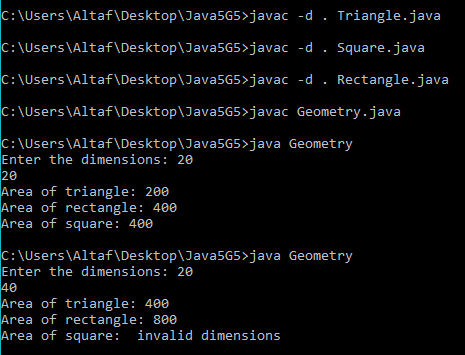
Square s1 = new Square();

s1.area(x,y);

}

}

**Output**:



**Post Lab task:**

1. Create a package with name Budget. It consists of state interface with two abstract methods revenue\_gained. Import this package in your program and check whether entire country revenue gained or lose by implementing state interface in 3 different classes with the names AP, KA and TN.

**Test Case 1: Input** 4.5%

-2.78%

-0.76%

**Output** “Revenue gained”

**Test Case 2: Input -**4.5%

-3.92%

-1.76%

**Output** “Revenue loss”

**Writing space for Post lab:***(For Student’s use only)*

**Program:**

//a Java program to create and access a package

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 07

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 08/01/2021

\*Program name : State.java

\*Topics : Creating a package, accessing a package, adding interface

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Creating a package

package Budget;

public interface State{

void setRevenue(double Revenue);

double getRevenue();

}

//Accessing created package

import Budget.State;

import java.util.\*;

class AP implements State{

double ap\_revenue;

public void setRevenue(double Revenue){

ap\_revenue=Revenue;

}

public double getRevenue(){

return ap\_revenue;

}

}

class KA implements State{

double ka\_revenue;

public void setRevenue(double Revenue){

ka\_revenue=Revenue;

}

public double getRevenue(){

return ka\_revenue;

}

}

class TN implements State{

double tn\_revenue;

public void setRevenue(double Revenue){

tn\_revenue=Revenue;

}

public double getRevenue(){

return tn\_revenue;

}

}

public class CountryRevenue{

public static void main(String...args){

Scanner sc= new Scanner(System.in);

System.out.print("Enter AP revenue: ");

double ap = sc.nextDouble();

System.out.print("Enter KA revenue: ");

double ka = sc.nextDouble();

System.out.print("Enter TN revenue: ");

double tn = sc.nextDouble();

AP a = new AP();

a.setRevenue(ap);

KA k = new KA();

k.setRevenue(ka);

TN t = new TN();

t.setRevenue(tn);

double total = a.ap\_revenue+k.ka\_revenue+t.tn\_revenue;

System.out.println("AP Revenue: "+a.getRevenue()+"%");

System.out.println("KA Revenue: "+k.getRevenue()+"%");

System.out.println("TN Revenue: "+t.getRevenue()+"%");

System.out.println("Total Revenue of the Country: "+String.format("%.2f",total)+"%");

if(total>0){

System.out.println("Revenue Gained");

} else{

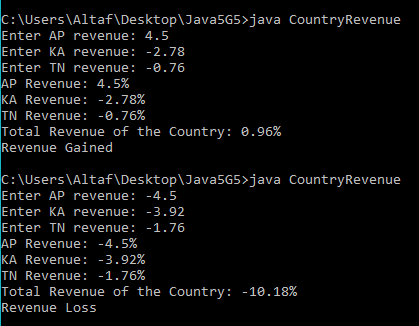
System.out.println("Revenue Loss");

}

}

}

**Output:**



1. Create a package called restaurant with two classes named menu and bill. Create class called customer which imports menu and Bill classes form package.

**Writing space for Post lab:***(For Student’s use only)*

**Output**:

//a Java program to create and access a package

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 07

\*Type of lab : Post Lab

\*Problem No. : 02

\*Programmer : Afrose Hussain

\*Date : 09/01/2021

\*Program name : Menu.java

\*Topics : Creating a package, accessing package, adding classes to a package, benefits of package, types of package, setting classpath

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Creating a package

package Restaurant;

public class Menu{

public void getMenu(){

System.out.println("\n\t\tMENU");

System.out.println("\n1. Chicken Biryani\tRs.340");

System.out.println("\n2. Veg Biryani\t\tRs.240");

System.out.println("\n3. Fried Rice\t\tRs.100");

System.out.println("\n4. Meals\t\tRs.140");

}

}

//Creating Bill class in package

package Restaurant;

import java.util.\*;

public class Bill{

Scanner sc = new Scanner(System.in);

int one,two,three,four;

double bill,tax;

public void askCustomer(){

System.out.print("\nNumber of First items you want: ");

one = sc.nextInt();

System.out.print("Number of Second items you want: ");

two = sc.nextInt();

System.out.print("Number of Third item you want: ");

three = sc.nextInt();

System.out.print("Number of Fourth item you want: ");

four = sc.nextInt();

}

public void finalBill(){

bill=(one\*340)+(two\*240)+(three\*100)+(four\*140);

if(bill<=750){

tax=5;

} else{

tax=8;

}

bill=bill+(bill\*tax/100);

}

public void display(String name){

System.out.println("Customer name: "+name);

System.out.println("Tax percent: "+tax+"%");

System.out.println("Total bill: "+bill);

}

}

//Accessing created package

import Restaurant.Menu;

import Restaurant.Bill;

import java.util.\*;

public class NewCustomer{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter customer name: ");

String name = sc.nextLine();

Menu m = new Menu();

Bill b = new Bill();

m.getMenu();

b.askCustomer();

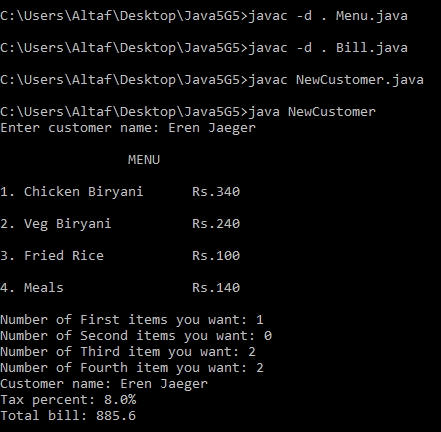
b.finalBill();

b.display(name);

}

}

**Output**:



1. Which of the following is an incorrect statement about packages?

a) Package defines a namespace in which classes are stored

b) A package can contain other package within it

c) Java uses file system directories to store packages

d) A package can be renamed without renaming the directory in which the classes are stored

A. d) A package can be renamed without renaming the directory in which the classes are stored.

Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

# Lab Session 08: Develop Java Application using Method overloading and Method overriding

**Date of the Session:** 20**/**01**/**2021 **Time of the Session:** 11:20AM **to** 1:00PM

**Program Title:** Apply interfaces to a given application.

**Pre Lab Task:**

1. If a method can perform more than 1 type of tasks, where the function name remains same, which feature of OOP is used here?

A. The feature of method overloading is used here.

1. Can we overload main() method?

A. Yes, there is possibility of overloading main() method. However, the JVM starts the execution of the program from public static void main(String args[]) only.

1. Which method is invoked before garbage collection?

A. The finalize() method is invoked before garbage collection. This method is responsible for garbage collection in java which is called implicitly by the JVM. It’s defined in the Object class of java.

1. If Object A has reference to Object B and Object B refer to Object A, apart from that there is no live reference to either object A or B, Does they are eligible to Garbage collection ?

A. Yes, they are eligible for garbage collection.

1. In the below Class X, is ‘method’ properly overloaded? Justify your answer.

class X

{

int method(int i, int d)

{

return i+d;

}

static int method(int i, double d)

{

return (int)(i+d);

}

double method(double i, int d)

{

return i+d;

}

static double method(double i, double d)

{

return i+d;

}

}

A. Yes, the method is properly overloaded in the class X. In all the 4 methods mentioned above, just the method name ‘method’ is same but remaining features like return type and parameters are all different. Since static methods can also be overloaded but not overridden, The method is properly overloaded in class X.

**In Lab task:**

***Problem Description***

Create a class called book with attributes name, author and methods setdata() and display() to read and display values of attributes. Expand book class in to other two sub classes with the names McGraw-Hill and Oxford. Include new price () method in both sub classes to specify price of in that publication.

**Test Case1 :**

**Input** {“java programing”, “Robert”}

300

400

**Output:** Name= java programing

Author=Robert

McGraw-Hill price=300

Oxford price=400

**Writing space for In lab:***(For Student’s use only)*

**Program:**

//a Java program to implement method overloading and method overriding

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 08

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 20/01/2021

\*Program name : Practice20.java

\*Topics : Method overloading, Method Overriding, late/Dynamic/Run-time binding,

Early/Static/Compile-time binding, Polymorphism

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//importing packages

import java.lang.\*;

import java.util.\*;

class Book{

//Book properties

String data;

String bookname;

String authorname;

//No-argument constructor

public Book(){

data="Unknown";

bookname="Unknown";

authorname="Unknown";

}

//setData to setup desired values

public void setData(String data){

this.data=data;

String temp[]=data.split(",");

for(int i=0;i<temp.length-1;i++){

bookname=temp[i];

authorname=temp[i+1];

}

}

public void display(){

System.out.println("Book name: "+bookname);

System.out.println("Author name: "+authorname);

}

}

class McGrawHill extends Book{

double price;

public McGrawHill(){

super();

price=0.0;

}

public void setData(String data, double price){

super.setData(data);

this.price=price;

}

public void display(){

super.display();

System.out.println("McGrawHill Price: "+price);

}

}

class Oxford extends Book{

double price;

public Oxford(){

super();

price=0.0;

}

public void setData(String data, double price){

super.setData(data);

this.price=price;

}

public void display(){

System.out.println("Oxford Price: "+price);

}

}

public class Practice20{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.println("Enter Book & Authorname");

String data = sc.nextLine();

McGrawHill m1 = new McGrawHill();

Oxford o1 = new Oxford();

m1.setData(data, 300);

o1.setData(data, 400);

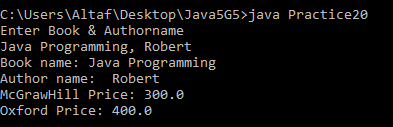
m1.display();

o1.display();

}

}

**Output:**



**Post Lab task:**

Create a class called IPLcricketmethodspalyed\_matches(), win() and loss().this class in class in to other two sub classes with the names Hyderabad\_sunrises and Chennai\_superkings, . Display number of matches win and loss by each team .

**Test case1: Input** {4,2,2, 3,2,1)

**Ourput:**Hyderabad\_sunrisesTeam summary:

Played Matches =4

Won = 2

Loss= 2

Chennai\_superkingsTeam summary:

Played Matches =3

Won = 2

Loss= 1

**Test case2: Input** {3,2,2, 3,2,1)

**Ourput:**Hyderabad\_sunrisesTeam summary:

Invalid win and loss data

Chennai\_superkingsTeam summary:

Played Matches =3

Won = 2

Loss= 1

**Writing space for Post lab:***(For Student’s use only)*

**Program:**

//a Java program to implement method overloading and method overriding

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 08

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 21/01/2021

\*Program name : Practice21.java

\*Topics : Method overloading, Method Overriding, late/Dynamic/Run-time binding,

Early/Static/Compile-time binding, Polymorphism

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//importing packages

import java.lang.\*;

import java.util.\*;

class IPLcricket{

public void played\_matches(){

System.out.println("Team Summary:");

System.out.print("Played Matches: ");

}

public void won(){

System.out.print("Won: ");

}

public void loss(){

System.out.print("Loss: ");

}

}

class Hyderabad\_sunrises extends IPLcricket{

int win,loss,play;

public Hyderabad\_sunrises(){

win=0;

loss=0;

play=0;

System.out.print("Hyderabad\_sunrises ");

}

public void played\_matches(int hp){

super.played\_matches();

play=hp;

System.out.println(play);

}

public void won(int hw){

super.won();

win=hw;

System.out.println(win);

}

public void loss(int hl){

super.loss();

loss=hl;

System.out.println(loss);

}

}

class Chennai\_superkings extends IPLcricket{

int win,loss,play;

public Chennai\_superkings(){

win=0;

loss=0;

play=0;

System.out.print("Chennai\_superkings ");

}

public void played\_matches(int cp){

super.played\_matches();

play=cp;

System.out.println(play);

}

public void won(int cw){

super.won();

win=cw;

System.out.println(win);

}

public void loss(int cl){

super.loss();

loss=cl;

System.out.println(loss);

}

}

public class Practice21{

public static void main(String...args){

int hp,hw,hl,cp,cw,cl;

Scanner sc = new Scanner(System.in);

System.out.println("Enter the no.of matches, won, lost of both teams: ");

String data = sc.nextLine();

String temp[]=data.split(",");

hp=Integer.parseInt(temp[0]);

hw=Integer.parseInt(temp[1]);

hl=Integer.parseInt(temp[2]);

cp=Integer.parseInt(temp[3]);

cw=Integer.parseInt(temp[4]);

cl=Integer.parseInt(temp[5]);

Hyderabad\_sunrises h1 = new Hyderabad\_sunrises();

if(hw+hl==hp){

h1.played\_matches(hp);

h1.won(hw);

h1.loss(hl);

} else{

System.out.println("Invalid Win and Loss data");

}

Chennai\_superkings c1 = new Chennai\_superkings();

if(cw+cl==cp){

c1.played\_matches(cp);

c1.won(cw);

c1.loss(cl);

} else{

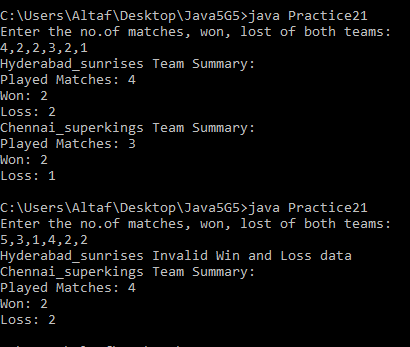
System.out.println("Invalid Win and Loss data");

}

}

}

**Output:**



Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

# Lab Session 09: Create a java application to copy content from one file to another using IO Streams

**Date of the Session:** 27/01/2021 **Time of the Session:** 11:20AM **to** 1:00PM

**Program Title:**Use I/O Streams in given application

**Pre Lab Task:**

1. What are the super most classes of all streams?

A. All the byte stream classes can be divided into two categories (input stream classes and output stream classes) and all character streams classes into two (reader classes and writer classes). There are four abstract classes from which all these streams are derived. The super most class of all byte stream classes is java.io.InputStream and for all output stream classes, java.io.OutputStream. Similarly for all reader classes is java.io.Reader and for all writer classes is java.io.Writer.

1. Differentiate Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy.

A. The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

1. List different file access modes.

A.

|  |  |
| --- | --- |
| **Mode** | **Description** |
| r | Read mode. Calling write methods will result in an IOException. |
| rw | Read and write mode. |
| rwd | Read and write mode - synchronously. All updates to file content is written to the disk synchronously. |
| rws | Read and write mode - synchronously. All updates to file content or meta data is written to the disk synchronously. |

1. Which streams are advised to use to have maximum performance in file copying?

A. BufferedInputStream and BufferedOutputStream on byte streams side and BufferedReader and BufferedWriter on character streams side.

**In Lab task:**

***Problem Description***

The following is a list of scores for a game. Enter them into a text file.

14401

3094

39201

57192

4948

55854

84

95

430

5502

65816

4994

7712

Write a program that opens this file, read in each score as an integer, and remember the highest score. After all the scores have been read, output the highest score (65816 in this example, but it could be different if the file had different scores in it).Apply FileNotFoundException concept.

.

**Test Case 1: Input** Scores.txt 112 234 555 233 553 222 432

**Output** The Highest Score is 555.

**Test Case 1: Input** Score.txt 123 122 132 111 454 1212

**Output** FileNotFoundException: File is not found.

**Program**:

//A java program to check the max numberin the given file

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 09

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 27-01-2021

\*Program name : Practice23.java

\*Topics : Fundamentals of files, Fundamentals of streams

Types of streams(byte and character)

IO stream classes hierarchy, handling files

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

import java.io.\*;

public class Practice23{

public static void main(String...args) throws Exception{

try{

File f = new File("Scores.txt");

BufferedReader br = new BufferedReader(new FileReader(f));

int max;

String line = br.readLine();

max = Integer.parseInt(line);

while(line!=null){

int next = Integer.parseInt(line);

if(max<next){

max=next;

}

line = br.readLine();

}

br.close();

System.out.println("Maximum score: "+max);

}

catch(Exception e){

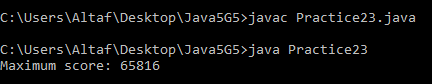
e.printStackTrace();

}

}

}

**Output**:



**Post Lab task:**

***Problem Description***

Write a program that will record that purchase made at a store. For each purchase, read from the keyboard an items name, its price , and the number bought. Compute the cost of the purchase ( number bought times price), and write all this data to a text file. Also , display this information and the current total cost on the screen. After all items have been entered, write the total cost to both the screen and the file. Since we want to remember all purchases made, you should append new data to the end of the file.

**Test Case 1: Input** Purchases.txt Laptop 12000 134

**Output:** Purchases File created :The Total expenditure is Rs. 10608000/-

**Program**:

//A java program that records a purchase made at a store in a file

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 09

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 27-01-2021

\*Program name : BillStore.java

\*Topics : Fundamentals of files, Fundamentals of streams

Types of streams(byte and character)

IO stream classes hierarchy, handling files

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

import java.io.\*;

public class BillStore{

public static void main(String[] args) {

try{

Scanner sc = new Scanner(System.in);

File p\_file = new File("Purchases.txt");

System.out.println("Enter file name, items name, its price and the number bought:");

String data = sc.nextLine();

String[] d = data.split(" ");

String file\_name = d[0];

String item\_name = d[1];

double price = Double.parseDouble(d[2]);

int quantity = Integer.parseInt(d[3]);

p\_file.createNewFile();

FileWriter fr = new FileWriter(p\_file, true);

BufferedWriter br = new BufferedWriter(fr);

br.write(item\_name + " " + price + " " + quantity + " total price: " + price \* quantity);

System.out.println(file\_name + " file created \nThe Total expenditure is Rs. " + price \* quantity);

br.close();

fr.close();

}catch(Exception e){

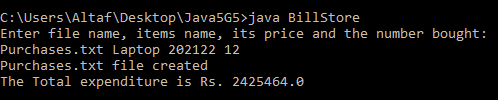
e.printStackTrace();

}

}

}

**Output**:



2. Which superstructure class allows reading data from an input byte stream in the format of primitive data types?

A. To read byte data (not strings), the DataInputStream class is used. InputStream classes reads data in terms of bytes but DataInputStream class is used to read data in terms of primitive data types such as short, char, int, float, double, boolean from an input stream. DataInputStream class is a filter class which is used to wrap any input stream to read primitive data types out of it.

3. What packages are stream classes in?

**A.** java.io.\*;

Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

# Lab Session 10: Implement Handling Exceptions to a given application

**Date of the Session: 03/02/2021 Time of the Session: 11:20AM to 01:00PM**

**Program Title:** Handle Exceptions to a given application

**Pre Lab Task:**

1. What are the Exception Handling Keywords in Java?

A. Exception Handling Keywords in Java are:

• try

• catch

• throw

• throws

• finally

1. Mention Java Exception Hierarchy?

A.

Object

Throwable

Error

Exception

Checked

Exception

Unchecked

Exception

1. What are important methods of Java Exception Class?

A. • **public String getMessage()**

Returns a detailed message about the exception that has occurred. This

message is initialized in the Throwable constructor.

• **public Throwable getCause()**

Returns the cause of the exception as represented by a Throwable object.

• **public void printStackTrace()**

Prints the result of toString() along with the stack trace to System.err, the error output stream.

1. Describe multi-catch block?

A. A single try block (code in try block) can create multiple errors, to catch different exceptions and handle them we use multiple catch blocks, they are also used to create user friendly exception messages;

try{

//code

}

catch(Exception1ClassName e) //Multiple catch blocks{

//handler code

}

catch(Exception2ClassName e){

//handler code

}

finally{

//code

}

1. Differentiate Checked and Unchecked Exception in Java?

A. **Checked Exceptions**: are the exceptions that are checked at compile time. If some code within a method throws a checked exception, then the method must either handle the exception or it must specify the exception using throws keyword.

**Unchecked Exceptions**: are the exceptions that are not checked at compiled time they will be done during runtime. In Java, all Unchecked exceptions are not forced by the compiler to either handle or specify the exception. It is up to the programmers to be civilized, and specify or catch the exceptions.

1. What is difference between throw and throws keyword in Java?

A.

|  |  |  |
| --- | --- | --- |
| **S.No** | **throw** | **throws** |
| 1 | Java throw keyword is used to explicitly throw an exception. | Java throws keyword is along with the method  implementation it can throw multiple  exceptions. |
| 2 | Checked exception cannot be propagated using throw only. | Checked exception can be propagated with  throws. |
| 3 | throw is followed by an instance. | throws is followed by method implementation  resided in class. |
| 4 | throw is used within the method. | throws is used with the method signature. |
| 5 | You cannot throw multiple exceptions. | You can declare multiple exceptions e.g.  public void method()throws  IOException,SQLException. |

1. How to write custom exception in Java?

A. Following steps are followed for the creation of Custom Exception.

• The user should create an exception class as a subclass of Exception class. Since all the exceptions are subclasses of Exception class, the user should also make his class a subclass of it. This is done as:

+ class MyException extends Exception

• We can write a default constructor in his own exception class.

+ MyException(){}

• To raise exception of user-defined type, we need to create an object to his exception class and throw it using throw clause, as:

+ MyException me = new MyException(“Exception details”);

+ throw me;

**In Lab task:**

***Problem Description:***

Develop an algorithm for a simple game of guessing at a secret five-digit code. When the user enters a guess at the code, the program returns two values: the number of digits in the guess that are in the correct position and the sum of those digits. For example, if the secret code is 53840 and the user guesses 83241, the digits 3 and 4 are in the correct position. Thus the program should respond with 2 and 7. Allow the user to guess a fixed number of times. If fixed number of times exceeds throw an exception.

Guess number is 53840

**Test Case 1: Input** 83241

**Output** 2 and 7

**Test Case 2: Input** 52841

**Output** 3 and 17

**Program:**

//Implement handling exceptions to a given application

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 10

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 03-02-2021

\*Program name : Practice22.java

\*Topics : Exception fundamentals, exception keywords, types of exceptions,

exception hierarchy, custom exceptions

try, catch, finally, throw, throws

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

class GameException extends Exception{

public GameException(String message){

super(message);

}

}

class Game{

Random r = new Random();

int secretCode=10000+r.nextInt(100000);

Scanner sc = new Scanner(System.in);

int sum, digit, count;

int n,t,g;

public void check(){

System.out.println("Your Secret Code is generated");

System.out.print("Enter Guess Code: ");

n=sc.nextInt();

int temp=n;

while(n>0){

count++;

n=n/10;

}

try{

if(count!=5){

//Usage of throw keyword

throw new GameException("Invalid Guess Code");

} else{

while(temp>0){

t=temp%10;

g=secretCode%10;

if(t==g){

digit++;

sum=sum+t;

}

temp=temp/10;

secretCode=secretCode/10;

}

if(digit==0){

throw new GameException("You Lost");

} else{

System.out.println("No.of digits matched: "+digit+" and Sum: "+sum);

}

}

} catch(GameException e){

e.printStackTrace();

}

}

}

public class Practice22{

public static void main(String...args){

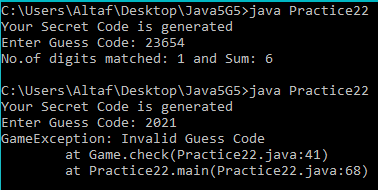
Game g = new Game();

g.check();

}

}

**Output**:



**Post Lab task:**

***Problem Description:***

A class called "Point" includes two instance variable - x (type int), y (type int). Provide a constructor that initialises the two instance variables.Add the following method to the Point class :

public int Slope(int num3, int num4)

The method returns the slope of the line drawn between the current point and the target point, using the formula (y2-y1)/(x2-x1) to determine the slope between two points (x1,y1) and (x2,y2).  
Note that this formula fails for the points with identical x-coordinates so throw an "ArithmeticException

**Test Case 1: Input** 4 9 10 21

**Output:** The slope is 2

**Program:**

//Implement handling exceptions to a given application

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 10

\*Type of lab : Post Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 04-02-2021

\*Program name : SlopeFinder.java

\*Topics : exception fundamentals, exception keywords, types of exceptions,

exception hierarchy, custom exceptions

try, catch, finally, throw, throws

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

class Point{

int x;

int y;

public Point(int x,int y){

this.x=x;

this.y=y;

}

public int slope(int a,int b){

int result=0;

try{

result=b/a;

}

catch(ArithmeticException e){

System.out.println("The slope is infinity");

System.exit(1);

}

return result;

}

}

public class SlopeFinder{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

System.out.print("Enter the points: ");

String data = sc.nextLine();

String[] d = data.split(" ");

int[] A = Arrays.stream(d).mapToInt(Integer::parseInt).toArray();

Point p = new Point(A[2]-A[0],A[3]-A[1]);

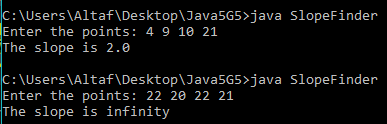
double output = p.slope(p.x,p.y);

System.out.println("The slope is "+output);

}

}

**Output**:



Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

# Lab Session 11: Apply Thread Capabilities on a given application

**Date of the Session: \_\_\_/\_\_\_/\_\_\_ Time of the Session:\_\_\_\_\_to\_\_\_\_\_\_**

**Program Title:**Handling Threads in Java

**Pre Lab Task:**

1. Mention two ways of creating a thread in java.
2. Create two threads, one thread will print odd numbers and second thread will print even numbers between 1 and 10 numbers with a delay of 100 ms after each number.
3. What are different states in lifecycle of Thread?
4. Can we call run() method of a Thread class?
5. How can we make sure main() is the last thread to finish in Java Program?

**In Lab task:**

***Problem Description:***

Write a java program using multithreading in which each thread reads the data from a text file and display the data of each file on Console alternatively such that one line from first input file is printed and then one line from another input file isprinted and so on.

**Detailed Description:**

You are required to develop a java program containing two threads that reads two files simultaneously.

In a program, first thread reads data from file“Personal\_Record.txt” and second thread reads the data from file “Academic\_Record.txt”.

When one thread reads a line from one file then it should allow another thread to read a line from another file.

After reading data from each file, program must write the output on console (Output screen) such that one line from first input file is printed and then one line from another input file is printed and so on.

**Test Case 1: Input** Personal\_Record.txt

Academic\_Record.txt

**Output Thread 1:**Pratap

**Thread 2:**Maths : 79

**Thread 1:**Mobile Number 12345

**Thread 2:**Programming in C : 77

**Thread 1:**Email : raghu@gmail.com

**Writing space for In lab:***(For Student’s use only)*

**Writing space for In lab:***(For Student’s use only)*

**Post Lab task:**

***Problem Description:***

In Java create a one thread to check if a given number is circular prime or not.

A circular prime is a prime number with the property that the number generated at each intermediate step when cyclically permuting its (base 10) digits will be prime. For example, 1193 is a circular prime,

since 1931, 9311 and 3119 all are also prime.

A circular prime with at least two digits can only consist of combinations of the digits 1, 3, 7 or 9, because having 0, 2, 4, 6 or 8 as the last digit makes the number divisible by 2, and having 0 or 5 as the last digit makes it divisible by 5.

and another thread is to keep track about all the prime numbers in between 2 and 1193

***Test Cases:***

**Test Case 1:**

Input a number: 1193

Output:

Thread 1 :

1193

Thread 2:

2

3

5

7

11

Thread 1:

1931

Thread 2:

13

17

19

23

29

31

37

41

Thread 1:

9311

3119

And so on

**Writing space for Post lab:***(For Student’s use only)*

**Writing space for Post lab:***(For Student’s use only)*

2. What is Thread Pool? How can we create Thread Pool in Java?

**Writing space for Post lab:***(For Student’s use only)*

3. Differences between synchronized and volatile keyword in Java?

**Writing space for Post lab:***(For Student’s use only)*

Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

**Date of the Session:** 16/02/2021 **Time of the Session:** 11:20 AM to 01:00PM

# Lab Session 12: Use Collections framework and create a java application.

**Program Title:** Use Collections framework and create a application.

**Pre Lab Task:**

1. How do you remove an entry from a Collection? and subsequently what is the difference between the remove() method of Collection and remove() method of Iterator, which one you will use while removing elements during iteration?

A. We can remove an entry from collections use remove method

Collection.remove(int element\_index\_to\_remove)

If we are iterating over a collection and use:

Collection.remove()

you can get runtime errors (specifically ConcurrentModifcationException) because you're changing the state of the object used previously to construct the explicit series of calls necessary to complete the loop.

If we use:

Iterator.remove()

you tell the runtime that you would like to change the underlying collection AND re-evaluate the explicit series of calls necessary to complete the loop.

1. What is the difference between Set and List in Java?

A. The main difference between List and Set is that Set is unordered and contains different elements, whereas the list is ordered and can contain the same elements in it.

1. Which one you will prefer between Array and ArrayList for Storing object and why?

A. I prefer ArrayList over Array when I need to store heterogeneous objects and also when I do not know the exact number of objects to be stored.

1. What is the difference between ArrayList and LinkedList?

A.

|  |  |
| --- | --- |
| ArrayList | LinkedList |
| This class uses a dynamic array to store the elements in it. With the introduction of [generics](https://www.geeksforgeeks.org/generics-in-java/), this class supports the storage of all types of objects. | This class uses a [doubly linked list](https://www.geeksforgeeks.org/doubly-linked-list/) to store the elements in it. Similar to the ArrayList, this class also supports the storage of all types of objects. |
| Manipulating ArrayList takes more time due to the internal implementation. Whenever we remove an element, internally, the array is traversed and the memory bits are shifted. | Manipulating LinkedList takes less time compared to ArrayList because, in a doubly-linked list, there is no concept of shifting the memory bits. The list is traversed and the reference link is changed. |
| This class implements a [List interface](https://www.geeksforgeeks.org/list-interface-java-examples/). Therefore, this acts as a list. | This class implements both the [List interface](https://www.geeksforgeeks.org/list-interface-java-examples/) and the [Deque interface](https://www.geeksforgeeks.org/deque-interface-java-example/). Therefore, it can act as a list and a deque. |
| This class works better when the application demands storing the data and accessing it. | This class works better when the application demands manipulation of the stored data. |

1. Difference between Set, List and Map Collection classes?

A. 1) **Duplicity:** List allows duplicate elements. Any number of duplicate elements can be inserted into the list without affecting the same existing values and their indexes.  
Set doesn’t allow duplicates. Set and all of the classes which implements Set interface should have unique elements.  
Map stored the elements as key & value pair. Map doesn’t allow duplicate keys while it allows duplicate values.

2)**Null values:** List allows any number of null values.  
Set allows single null value at most.  
Map can have single null key at most and any number of null values.

3) **Order:** List and all of its implementation classes maintains the insertion order.  
Set doesn’t maintain any order; still few of its classes sort the elements in an order such as LinkedHashSet maintains the elements in insertion order.  
Similar to Set Map also doesn’t stores the elements in an order, however few of its classes does the same. For e.g. TreeMap sorts the map in the ascending order of keys and LinkedHashMap sorts the elements in the insertion order, the order in which the elements got added to the LinkedHashMap.

**In Lab task:**

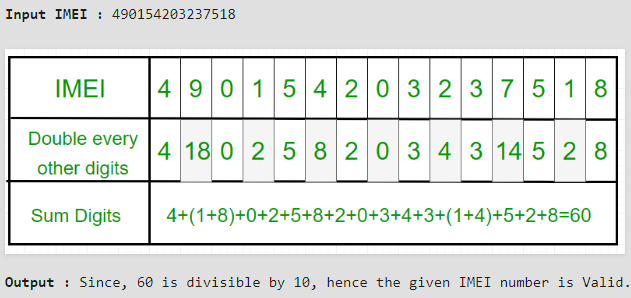
***Problem Description:***

Use any collection framework and code a program to check for a Valid IMEI Number

The IMEI (15 decimal digits: 14 digits plus a check digit) includes information on the origin, model, and serial number of the device.

The IMEI is validated in following steps:

1. Starting from the rightmost digit, double the value of every second digit (e.g., 7 becomes 14).
2. If doubling of a number results in a two digits number i.e greater than 9(e.g., 7 × 2 = 14), then add the digits of the product (e.g., 14: 1 + 4 = 5), to get a single digit number.
3. Now take the sum of all the digits.
4. Check if the sum is divisible by 10 i.e.(total modulo 10 is equal to 0) then the IMEI number is valid; else it is not valid.



***Test Cases:***

**Test Case 1: Input** 490154203237518

**Output** Valid EMI

**Program:**

//A java application toimplement Collections Framework

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Lab session number : 12

\*Type of lab : In Lab

\*Problem No. : 01

\*Programmer : Afrose Hussain

\*Date : 16-02-2021

\*Program name : CheckIMEI.java

\*Topics : Fundamentals of Collectons, Collections Framework Hierarchy,it's methods

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//import the required packages

import java.lang.\*;

import java.util.\*;

public class CheckIMEI{

public static void main(String...args){

Scanner sc = new Scanner(System.in);

int size = 15,sum=0;

int x=0;

System.out.print("Enter 15 digit IMEI code: ");

String data=sc.nextLine();

//Creaing ArrayList Collection

ArrayList<Integer> imei=new ArrayList<Integer>(size);

//Append the elements at the end of the list

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

x=data.charAt(i)-'0';

imei.add(x);

}

Integer[] temp = new Integer[imei.size()];

for(int j=0;j<temp.length;j++){

temp[j]=imei.get(j);

}

if(temp.length!=15){

System.out.println("Invalid Input");

} else{

int d=0;

for(int i=size-1;i>=0;i--){

d=temp[i];

if(i%2==1){

d=2\*d;

}

sum=sum+sumDig(d);

}

}

System.out.println("\nOutput Sum: "+sum);

if(sum%10==0){

System.out.println("\nValid IMEI number");

} else{

System.out.println("\nInvalid IMEI number");

}

}

public static int sumDig(int n){

int a=0;

while(n>0){

a=a+(n%10);

n=n/10;

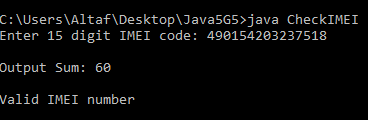
}

return a;

}

}

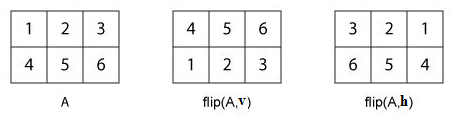
**Output**:



**Post Lab task:**

***Problem Description:***

Create a java program with a class FlipTest and add a method flip(A, dim) that takes a array of multi-dimension using any collections framework say it as A and a character dim that takes a value either “h” or “v” returns the array as a result of flipping the elements either horizontally or vertically. If dim variable is other than “h” or “v”, your function should return array unchanged. Note that your function should not change A itself, and should return the flipped array either horizontally or vertically.



**Test Case 1: Input**

1 2 3

4 5 6 and v

**Output:**

4 5 6

1 2 3

**Writing space for Post lab:***(For Student’s use only)*

**Writing space for Post lab:***(For Student’s use only)*

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |

Students Signature

## (For Evaluator’s use only)

# Lab Session 13: Develop GUI application using Applets

**Date of the Session: \_\_\_/\_\_\_/\_\_\_ Time of the Session:\_\_\_\_\_to\_\_\_\_\_\_**

**Program Title:** Use Applets and create a gui application.

**Pre Lab Task:**

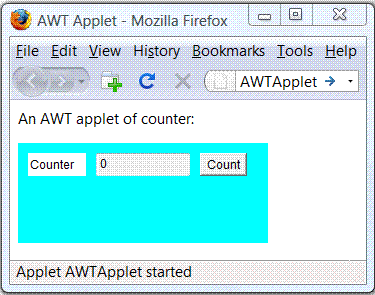
1. How do Applets differ from Applications?
2. What is the difference between applets loaded over the internet and applets loaded via the file system?
3. How to insert an applet into a HTML page?
4. What is the order of method invocation in Applet?
5. Can we pass parameters to an applet from HTML page to an applet? How?

**In Lab task:**

***Problem Description:***

Write an applet with a TextArea where the use can enter some text. The applet should have a button. When the user clicks on the button, the applet should count the number of lines in the user's input, the number of words in the user's input, and the number of characters in the user's input. This information should be displayed on three labels in the applet. Recall that if textInput is a TextArea, then you can get the contents of the TextArea by calling the function textInput.getText(). This function returns a String containing all the text from the TextArea. The number of characters is just the length of this String. Lines in the String are separated by the new line character, '\n', so the number of lines is just the number of new line characters in the String, plus one.

***Expected Output***



**Writing space for In lab:***(For Student’s use only)*

**Writing space for In lab:***(For Student’s use only)*

**Writing space for In lab:***(For Student’s use only)*

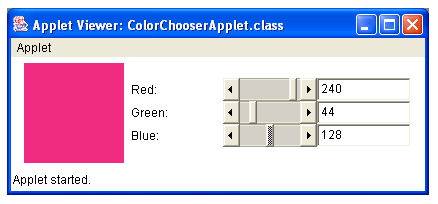
**Post Lab task:**

***Problem Description:***

Create a class [RGBColorChooser](http://math.hws.edu/eck/cs124/javanotes3/source/RGBColorChooser.java) that extends applet, lets the user set the red, green, and blue levels in a color by manipulating scroll bars. Something like this could make a useful custom component. Such a component could be included in a program to allow the user to specify a drawing color, for example. Rewrite the RGBColorChooser as a component. Make it a subclass of Panel instead of Applet. Instead of doing the initialization in an init() method, you'll have to do it in a constructor. The component should have a method, getColor(), that returns the color currently displayed on the component. It should also have a method, setColor(Color c), to set the color to a specified value. Both these methods would be useful to a program that uses your component.

In order to write the setColor(Color c) method, you need to know that if c is a variable of type Color, then c.getRed() is a function that returns an integer in the range 0 to 255 that gives the red level of the color. Similarly, the functions c.getGreen() and c.getBlue() return the blue and green components.

Test your component by using it in a simple applet that sets the component to a random color when the user clicks on a button,



**Writing space for Post lab:***(For Student’s use only)*

**Writing space for Post lab:***(For Student’s use only)*

**Writing space for Post lab:***(For Student’s use only)*

Students Signature

## (For Evaluator’s use only)

|  |  |  |  |
| --- | --- | --- | --- |
|  | |  | |
|  | Comment of the Evaluator (if Any) | Evaluator’s Observation  Marks Secured:\_\_\_\_\_\_\_ out of \_\_\_\_\_\_\_\_  Full Name of the Evaluator:  Signature of the Evaluator Date of Evaluation: |
|  |  | |