**Basic Numeric Computations Programs in Java**

**1.Write a program to find the difference between sum of the squares and the square of the sums of n numbers?**

class Difference

{ int n1,n2,n3;

Difference(int n1,int n2,int n3)

{

this.n1=n1;

this.n2=n2;

this.n3=n3;

}

double squareOfTheSum(int m1,int m2,int m3)

{

double m4= m1+m2+m3;

double m5= Math.pow(m4,2);

return m5;

}

double sumOfTheSquare(int p1,int p2,int p3)

{

double p4=(Math.pow(p1,2))+(Math.pow(p2,2))+(Math.pow(p3,2));

return p4;

}

}

public class MainDifference

{

public static void main(String args[])

{

Difference d= new Difference( Integer.parseInt(args[0]),Integer.parseInt(args[1]),Integer.parseInt(args[2]));

double squareOfSum= d.squareOfTheSum(Integer.parseInt(args[0]),Integer.parseInt(args[1]),Integer.parseInt(args[2]));

double sumOfSquare=d.sumOfTheSquare(Integer.parseInt(args[0]),Integer.parseInt(args[1]),Integer.parseInt(args[2]));

double difference= sumOfSquare - squareOfSum;

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

}

}

**Develop a program that accepts the area of a square and will calculate its perimeter.**

public class AreaPerimeter

{

public static void main(String[] args)

{

double p=4\*Math.sqrt(Integer.parseInt(args[0]));

System.out.println("");

System.out.print("Perimeter of the Square : "+p);

System.out.println("");

}

}

**Utopias tax accountants always use programs that compute income taxes even though the tax rate is a solid, never-changing 15%. Write a program that calculates net pay of an employee based on number of hours the employee has worked. Assume an hourly rate of $12.**

public class Net

{

public static void main(String[] args)

{

int h=Integer.parseInt(args[0]);

System.out.println("Enter the no. of hours took leave : ");

double gross=(h\*12);

double tax= gross\*0.15;

double net=gross-tax;

System.out.println("");

System.out.println("Gross Pay (in $) : "+gross);

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

System.out.println("Net Pay (in $) : "+net);

}

}

**An old-style movie theater has a simple profit program. Each customer pays $5 per ticket. Every performance costs the theater $20, plus $.50 per attendee. Develop the program that accepts the number of attendees (of a show) and calculates how much income the show earns.**

public class TotalProfit

{

public static void main(String[] args)

{

int n=Integer.parseInt(args[0]);

double profit = (n\*5)-(20+(n\*0.5));

System.out.println("");

System.out.println("Total Profit of the theater per show (in $) is : " + profit);

}

}

**5.Develop the program which computes the height that a rocket reaches in a given amount of time. If the rocket accelerates at a constant rate g, it reaches a speed of g · t in t time units and a height of 1/2 \* v \* t where v is the speed at t.**

Code:

public class Height

{

public static void main(String[] args)

{

double t=Double.parseDouble(args[0]);

double v=9.8\*t;

double height=0.5\*v\*t;

System.out.println("");

System.out.println("Height reached (in meters) is : " + height);

}

}

**1.Develop a program that accepts the length and width of a rectangular floor and the edge length of a square tile and will compute the whole number of tiles needed to cover the floor completely.**

Code:

public class NoOfTiles

{

public static void main(String args[])

{

double n=(Double.parseDouble(args[0])\*Double.parseDouble(args[1]))/Double.parseDouble(args[2]);

System.out.println("No. of tiles"+n);

}

}

#### How to Calculate Day Of Week in Java?

Code:

public class DayOfWeek {

public static void main(String [] args)

{

int day = Integer.parseInt(args[0]);

int month = Integer.parseInt(args[1]);

int year = Integer.parseInt(args[2]); ;

int day\_week = (year - 1900) \* 365 + (year - 1900) / 4;

if (year % 4 == 0 && month <= 2) {

day\_week--;

}

switch (month) {

case 12: day\_week += 30;

case 11: day\_week+= 31;

case 10: day\_week+= 30;

case 9: day\_week+= 31;

case 8: day\_week+= 31;

case 7: day\_week+= 30;

case 6: day\_week+= 31;

case 5: day\_week+= 30;

case 4: day\_week+= 31;

case 3: day\_week+= 28;

case 2: day\_week+= 31;

}

day\_week = (day + day\_week) % 7;

switch (day\_week) {

case 0: System.out.println("Sunday");

break;

case 1: System.out.println("Monday");

break;

case 2: System.out.println("Tuesday");

break;

case 3: System.out.println("Wednesday");

break;

case 4: System.out.println("Thursday");

break;

case 5: System.out.println("Friday");

break;

case 6: System.out.println("Saterday");

break;

}

}

}

#### Basic Conditional Operational Programs in Java

**1.Find the maximum of three numbers**

Code:

public class Maximum

{

public static void main(String args[])

{

int x=Integer.parseInt(args[0]);

int y=Integer.parseInt(args[1]);

int z=Integer.parseInt(args[2]);

int max;

if(x>y)

{ if(x>z)

max=x;

else

max=z;

}

else

{ if(y>z)

max=y;

else

max=z;

}

System.out.println("Max: "+max);

}

}

**2 .Write a program to check whether the input alphabet is a vowel or not.**

Code:

public class Vowel

{

public static void main(String args[])

{

char alphabet\_finder=args[0].charAt(0);

switch(alphabet\_finder)

{ case 'a':

case 'e':

case 'i':

case 'o':

case 'u':System.out.println("It is vowel"); break;

default:System.out.println("Consonant! it is");

}

}

}

**3.Develop a program, that accepts a deposit amount and calculates amount of interest the deposit amount earns in an year. The bank pays a flat 4% for deposits of up to Rs.1000, a flat 4.5% per year for deposits of up to Rs.5000, and a flat 5% for deposits of more than Rs.5000.**

Code:

public class CalculateAmount

{

public static void main(String args[])

{

double deposit\_amount=Double.parseDouble(args[0]);

if(deposit\_amount<=1000)

System.out.println("The Interest is:"+(deposit\_amount\*0.04));

else if(deposit\_amount>1000&&deposit\_amount<=5000)

System.out.println("The interest is :"+(deposit\_amount\*0.045));

else if(deposit\_amount>5000)

System.out.println("The interest is:"+(deposit\_amount\*0.05));

}

}

**4.Develop the program, which accepts the gross pay and produces the amount of tax owed. For a gross pay of $240 or less, the tax is 0%; for over $. 240 and $. 480 or less, the tax rate is 15%; and for any pay over $480, the tax rate is 28%.**

Code:

public class TaxCalculator

{

public static void main(String args[])

{

double gross\_pay=Double.parseDouble(args[0]);

if(gross\_pay<=240)

System.out.println("The tax is :"+gross\_pay);

else if(gross\_pay>240&&gross\_pay<=480)

System.out.println("The tax is:"+(gross\_pay\*0.15));

else if(gross\_pay>480)

System.out.println("the tax is :"+(gross\_pay\*0.28));

}

}

**5.Some credit card companies pay back a small portion of the charges a customer makes over a year. A particular credit card company's pay back policy is as follows:  
1.0.25% for charges up to Rs. 500.  
2.0.50% for the next Rs.1000 (that is, the portion between Rs. 500 and Rs. 1500),   
3.0.75% for the next Rs.1000 (that is, the portion between Rs. 1500 and Rs. 2500),   
4.1.0% for charges above Rs2500.   
Thus, a customer who charges Rs. 400 a year receives Rs.1.00, which is 0.25 · 1/100 · 400, and one who charges Rs1, 400 a year receives Rs. 5.75, which is 1.25 = 0.25 · 1/100 · 500 for the first Rs. 500 and 0.50 · 1/100 · 900 = 4.50 for the next Rs. 900. Determine by hand the pay backs amount for a customer who charged Rs. 2000 and one who charged Rs. 2600.  
Define the program, which accepts a charge amount and computes the corresponding pay back amount.**

Code:

public class PayBack

{

public static void main(String args[])

{

float charge\_amount=Float.parseFloat(args[0]);

if(charge\_amount<=500)

System.out.println("PayBack Amount is:"+(0.25\*0.01\*charge\_amount));

if(charge\_amount>500&&charge\_amount<=1500)

System.out.println("PayBack Amount is:"+((0.25f\*0.01\*500)+(0.50f\*0.01\*charge\_amount )));

if(charge\_amount>1500&&charge\_amount<=2500)

System.out.println("payBack Amount is :"+((0.25f\*0.01\*500)+(0.50f\*0.01\*1000)+(0.75f\*0.01\*charge\_amount)));

if(charge\_amount>2500)

System.out.println("pay back amount is :"+((0.25f\*0.01\*500)+(0.50f\*0.01\*1000)+(0.75f\*0.01\*1000)+(1\*0.01\*charge\_amount)));

}

}

**6.Implement a method that returns the day of the week for a given day (1..31), month (1..12) and year.  
The day of the week of dates between March 1900 and February 2100 can be calculated as follows:   
1.Calculate the total number of days from 1900/1/1 to the given date (see below, for details).  
2.Divide this number by 7, the remainder is the day of the week (0 - sunday, 1 - monday, etc)  
To calculate the total number of days:  
1.Subtract 1900 from the given year and multiply the result by 365. Add the missing leaps years by adding (year - 1900) / 4.  
2.If the given year is a leap year and the month is January or February, subtract 1 from the previous result.   
3.Add no of days of the given year till the given month (in case of February always add 28, because the additional day for a leap year would have been already added).**

Code:

public class DayOfWeek {

public static void main(String [] args)

{

int day = Integer.parseInt(args[0]);

int month = Integer.parseInt(args[1]);

int year = Integer.parseInt(args[2]); ;

int day\_week = (year - 1900) \* 365 + (year - 1900) / 4;

if (year % 4 == 0 && month <= 2) {

day\_week--;

}

switch (month) {

case 12: day\_week += 30;

case 11: day\_week+= 31;

case 10: day\_week+= 30;

case 9: day\_week+= 31;

case 8: day\_week+= 31;

case 7: day\_week+= 30;

case 6: day\_week+= 31;

case 5: day\_week+= 30;

case 4: day\_week+= 31;

case 3: day\_week+= 28;

case 2: day\_week+= 31;

}

day\_week = (day + day\_week) % 7;

switch (day\_week) {

case 0: System.out.println("Sunday");

break;

case 1: System.out.println("Monday");

break;

case 2: System.out.println("Tuesday");

break;

case 3: System.out.println("Wednesday");

break;

case 4: System.out.println("Thursday");

break;

case 5: System.out.println("Friday");

break;

case 6: System.out.println("Saterday");

break;

}

}

}

**1.Write a program to automate the following loan policy.  
Age category  
Gender  
Profession  
Personal assets  
Loan amount eligible  
16 -25  
M /F  
  
Self-Employed  
>25000  
10000  
  
  
Professional  
  
15000  
26 - 40  
M   
SelfEmployed / Professional  
> 40000  
25000  
  
F  
  
  
30000  
41 - 60  
M / F  
SelfEmployed / Professional  
> 50000  
40000  
> 60  
M/F  
Self Employed  
> 25000  
35000 – Age \* 100  
  
  
Retired  
  
25000 – Age \* 100  
  
Write a program that accepts age, gender, job status and assets, and return the eligible loan amount.**

Code:

public class CalculateLoan

{

public static void main(String args[])

{

int loan=0;

int age=Integer.parseInt(args[0]);

char gender=args[1].charAt(0);

String job=args[2];

int asset=Integer.parseInt(args[3]);

if ((age>=16)&&(age<=25))

{ if(gender=='M' || gender=='F')

if(asset>25000)

{ if(job.equals("selfemployed"))

loan=10000;

else

loan=15000;

}

}

if((age>=26)&&(age<=40))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(asset>40000)

{ if(gender=='M')

loan=25000;

else

loan=30000;

}

}

if((age>=41)&&(age<=60))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(gender=='M' || gender=='F')

if(asset>40000)

loan=40000;

}

if(age>60)

{ if(gender=='M' || gender=='F')

if(asset>25000)

if(job.equals("selfemployed"))

loan=35000-age\*100;

else

loan=25000-age\*100;

}

System.out.println("loan amount: " + loan);

}

}

#### Training Center Information System

**A reputed IT organization has recently opened its training centers across various cities in India to provide comprehensive training facilities to the organization's newly recruits on various domains on which its works on, such as Java, .NET, Unix, Oracle, etc. The organization has decided to develop an information system to manage the activities of all its training centers. The system analysis and design team has come up with the following model for the project as shown in the detailed class diagram:**  
[Click image for larger version. 

Name: image002.jpg 
Views: 360 
Size: 110.8 KB 
ID: 641](http://techforum4u.com/attachment.php?attachmentid=641&d=1327054337)  
Perform the code implementation of the above model, according to the given classes, their attributes and business methods. All the classes should have the necessary getters and setters for their attributes.  
  
  
  
Description of the constructors for all the classes:  
  
  
  
Class  
  
  
Fields of the constructor  
  
  
Class  
  
  
Fields of the constructor  
  
TrainingCenter  
  
  
code, city  
  
  
Trainee  
  
  
id, name  
  
TechnicalInstructor  
  
  
id, name, streamCode  
  
  
Batch  
  
  
batchId, stream  
  
SoftSkillsInstructor  
  
  
id, name  
  
  
Stream  
  
  
streamCode  
  
  
  
The detailed description of the business methods that need to be implemented are given below:  
  
  
  
1. addBatch(String batchId, Stream s): To add a new batch to the list of batches of the training center. The function will create a new batch with the given batch id and assign it to a given stream (using the constructor of the Batch class) and add it to the list of batches. The function will also add the stream to the stream list, if the stream is not found in the stream list from before; if the stream already exists in the stream list, then it will not be added. The function will also return the number of current batches in the training center; if a batch id is not unique, then the function will return -1.  
  
  
  
2. addTrainee(Trainee t): To add a given trainee to the list of trainees of the training center. The trainee id fields of each trainee should be unique. If the trainee id of the given trainee already exists in the list, then the method should display "Duplicate Trainee Id".  
  
  
  
3. addInstructor(Instructor i): To add a new instructor to a training center's list of instructors, only if the instructor id field is unique in the list of instructors; otherwise the method will display "Duplicate Instructor Id".  
  
  
  
4. assignTechLead(String batchId, int id): To assign a given technical instructor (identified by the instructor's id) to a batch (identified by the batch id), only if the batch has not been assigned a technical instructor before (the method will update the techLead field of the given batch). The method will also ensure that a technical instructor is not assigned to more than 2 batches. While assigning a batch to a technical instructor, the most important condition that has to be satisfied is that, steam code of the technical instructor should match with the stream code of the batch's stream. For any violation to these constraints, the method will throw an InvalidAssignmentException. If the batch id or the technical instructor's id is not found to exist in the respective lists, then the method will throw a RecordNotFoundException.  
  
  
  
5. assignBizLead(String batchId, int id): To assign a soft skills (business skill) instructor to the given batch (the method will update the bizLead field of the given batch). The method will ensure that a soft skills instructor should not be assigned to more than 3 batches. For any violation to this constraint, the method will throw an InvalidAssignmentException. If the batch id or the soft skills instructor's id is not found to exist in the respective lists, then the method will throw a RecordNotFoundException.  
  
  
  
6. assignBatch(int traineeId, String batchId): This method will take a trainee id and a batch id as arguments and assign the trainee to the given batch (the method will update the batchId field of the given trainee). The method will also ensure that 1 batch can have a maximum of 36 trainees, and that 1 trainee can only be assigned to 1 batch. For any violation to this constraint, the method will throw an InvalidAssignmentException. If the trainee id or the batch id does not exist, then the method will throw a RecordNotFoundException.  
  
  
  
7. findInstructor(String batchid, String instructorType): This method will take a batch id and type of instructor ("tech" or "biz") as parameters and return the name of the instructor of the given type. For any other string, other than "tech" or "biz", the function will return null. If the batch id is not found in the list of batches, then the function will throw a RecordNotFoundException.  
  
  
  
8. countTrainees(String code): This function will take a stream code as parameter and return the number of trainees assigned to the given stream. If the stream id does not exist in the stream list, then the function will return -1.  
  
  
  
9. findHighestBatch(): To return the list of batch ids which contain the maximum number of trainees.  
  
  
  
10. getModules(): This function will return the list of modules that an instructor has to conduct training sessions on. For a technical instructor it will return a list containing the modules, Application Understanding, User Interface, Problem Solving, and Information Management. If the stream code is J2EE or DOTNET, then it will also include, Web Site Development. For a soft skills instructor it will return a list, containing the subjects, Communication Skills, Presentation Skills, Writing Skills and Professional Etiquette.  
  
  
  
To make the above methods more user-friendly, the code development team has been requested to make sure that all the string-matching conditions are case-insensitive; e.g. if a stream code exists in the stream list as J2EE and the user searches using j2ee, then it should match with the existing stream code.  
  
  
  
Specification of packages for the above model entities:  
  
  
  
Package  
  
  
Classes in the package  
  
model  
  
  
TrainingCenter, Trainee, Batch, Stream, Instructor, TechnicalInstructor, SoftSkillsInstructor, BatchAssignment  
  
exceptions  
  
  
InvalidAssignmentException, RecordNotFoundException  
  
  
  
The InvalidAssignmentException is a user-defined exception which should display the message "The batch assignment is invalid" whenever it is handled.  
  
  
  
The RecordNotFoundException is also a user-defined exception which should display the message "Matching records not found" whenever it is handled.  
  
  
  
The methods: 1, 2 and 3, should have the NullPointerException handler that will display how the exception was propagated.  
  
  
  
  
Solution:

Code:

package assignment;

import java.io.\*;

import java.util.\*;

class Trainee

{

int id;

String name;

Batch batchId;

public Trainee(int id,String name)

{

this.id=id;

this.name=name;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Batch getBatchId() {

return batchId;

}

public void setBatchId(Batch batchId) {

this.batchId = batchId;

}

}

class Batch

{

String batchId;

Stream stream;

public Batch(String batchId,Stream stream)

{

this.batchId=batchId;

this.stream=stream;

}

public String getBatchId() {

return batchId;

}

public void setBatchId(String batchId) {

this.batchId = batchId;

}

public Stream getStream() {

return stream;

}

public void setStream(Stream stream) {

this.stream = stream;

}

public TechnicalInstructor getTechLead() {

return techLead;

}

public void setTechLead(TechnicalInstructor techLead) {

this.techLead = techLead;

}

public SoftSkillsInstructor getBizLead() {

return bizLead;

}

public void setBizLead(SoftSkillsInstructor bizLead) {

this.bizLead = bizLead;

}

TechnicalInstructor techLead;

SoftSkillsInstructor bizLead;

}

class Stream

{

String streamCode;

ArrayList<String> subjects=new ArrayList<String>();

public Stream(String streamCode)

{

this.streamCode=streamCode;

}

public String getStreamCode() {

return streamCode;

}

public void setStreamCode(String streamCode) {

this.streamCode = streamCode;

}

public ArrayList<String> getSubjects() {

return subjects;

}

public void setSubjects(ArrayList<String> subjects) {

this.subjects = subjects;

}

}

interface BatchAssignment

{

public void assignTechLead(String name,int id) throws Exception;

public void assignBizLead(String name,int id) throws Exception;

}

abstract class Instructor

{

int id;

String name;

public Instructor(){}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getSessions() {

return sessions;

}

public void setSessions(int sessions) {

this.sessions = sessions;

}

int sessions;

public Instructor(int id,String name)

{

this.id=id;

this.name=name;

}

public abstract ArrayList<String> getModules();

}

class TechnicalInstructor extends Instructor

{

String streamCode;

public TechnicalInstructor(){super();}

public TechnicalInstructor(int id,String name,String streamCode)

{

super(id,name);

this.streamCode=streamCode;

}

public String getStreamCode() {

return streamCode;

}

public void setStreamCode(String streamCode) {

this.streamCode = streamCode;

}

public ArrayList<String> getModules()

{

ArrayList<String> m=new ArrayList<String>();

m.add("Application Understanding");

m.add("User Interface");

m.add("Problem Solving");

m.add("Information Management");

return m;

}

}

class SoftSkillsInstructor extends Instructor

{

public SoftSkillsInstructor(){super();}

public SoftSkillsInstructor(int id,String name)

{

super(id,name);

}

public ArrayList<String> getModules()

{

ArrayList<String> m=new ArrayList<String>();

m.add("Communication");

m.add("Presentation");

m.add("Writing");

m.add("Professional");

return m;

}

}

class InvalidAssignmentException extends Exception

{

public String toString()

{

return "The batch assignment is not valid";

}

}

class RecordNotFoundException extends Exception{

public String toString()

{

return "Matching records not found";

}

}

public class TrainingCenter implements BatchAssignment

{

String city;

String code;

ArrayList<Batch> batch\_ass=new ArrayList<Batch>();

ArrayList<Stream> stream\_ass=new ArrayList<Stream>();

ArrayList<Trainee> trainee\_ass=new ArrayList<Trainee>();

ArrayList<Instructor> instructors\_ass=new ArrayList<Instructor>();

public int addBatch(Batch b)

{

int flag=0,count=0,f=0;

for(Batch bt:batch\_ass)

{

count++;

if (bt.getBatchId().equals(b.getBatchId()))

{

flag=1;

break;

}

}

if(flag==0)

{

for(Stream s:stream\_ass)

{

if(s.getStreamCode()==b.getStream().getStreamCode())

{

f=1;

break;

}

}

if(f==0)

stream\_ass.add(b.getStream());

batch\_ass.add(b);

return count;

}

else

return -1;

}

public void addTrainee(Trainee t)

{

int flag=0;

for(Trainee tn:trainee\_ass)

{

if (tn.getId()==t.getId())

{

System.out.println("\nYour Trainee Id is Duplicate");

flag=1;

}

}

if(flag==0)

{

trainee\_ass.add(t);

System.out.println("Registration of trainee is succesfull");

}

}

public void addInstructor(Instructor i)

{

int flag=0,fix=0;

for(Instructor map:instructors\_ass)

{

if (map.getId()==i.getId())

{

System.out.println("Your Trainee Id is Duplicate");

flag=1;

}

}

if(flag==0)

{

if(i.getClass().getName().equals("SoftSkillsInstructor"))

{

instructors\_ass.add(i);

System.out.println("SoftSkill Instructor is registered now");

}

else

{

for(Stream sop:stream\_ass)

{

TechnicalInstructor t=(TechnicalInstructor)i;

if(sop.getStreamCode()==t.getStreamCode())

{

fix=1;

instructors\_ass.add(i);

System.out.println("Technical Instructor is registered now ");

break;

}

}

if(fix==1)

System.out.println("Sorry!! Technical Instructor cannot be registered as to the Stream (no existence)");

}

}

}

public TrainingCenter(String city,String code) throws InvalidAssignmentException

{

this.city=city;

this.code=code;

}

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public ArrayList<Batch> getBatches() {

return batch\_ass;

}

public void setBatches(ArrayList<Batch> batches) {

this.batch\_ass = batches;

}

public void assignBatch(int id,String name)

{

int start1=0,start2=0,count=0;

Trainee tr=null;

Batch b=null;

for(Batch bat:batch\_ass)

{

if (bat.getBatchId().equals(name))

{

b=bat;

start1=1;

break;

}

}

for(Trainee t:trainee\_ass)

{

if (t.getId()==id)

{

tr=t;

start2=1;

break;

}

}

if(start1==0 || start2==0)

System.out.println("Sorry !!");

if(start1==1 && start2==1)

{

if(tr.getBatchId()==null)

{

for(Trainee top:trainee\_ass)

{

if(top.getBatchId().getBatchId()==b.getBatchId())

count++;

}

if(count<36)

{

tr.setBatchId(b);

System.out.println("Trainee is assigned a batch ");

}

else

try {

throw new InvalidAssignmentException();

} catch (InvalidAssignmentException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

else

System.out.println("Trainee already has been assigned batch ");

}

}

public void assignTechLead(String name,int id) throws InvalidAssignmentException,RecordNotFoundException

{

int start1=0,start2=0,count=0;

TechnicalInstructor t=null;

for(Batch bat1:batch\_ass)

{

if (bat1.getBatchId().equals(name))

{

start1=1;

break;

}

}

for(Instructor him:instructors\_ass)

{

if (him.getId()==id)

{

t=(TechnicalInstructor)him;

start2=1;

break;

}

}

if(start1==0 || start2==0){

throw new InvalidAssignmentException();

}

if(start1==1 && start2==1)

{

for(Batch bt:batch\_ass)

{

if (bt.getBatchId().equals(name))

{

if(bt.getTechLead().getId()==id)

count++;

}

}

if(count<2)

{

for(Batch bat1:batch\_ass)

{

if(t.getStreamCode().equals(bat1.getStream().getStreamCode()))

{

bat1.setTechLead(t);

System.out.println("Tech lead is assigned to !!"+bat1.getBatchId());

break;

}

}

}

else

System.out.println("Tech lead is assigned two batches before");

}

}

public void assignBizLead(String name,int id) throws RecordNotFoundException

{

int start1=0,start2=0,count=0;

SoftSkillsInstructor t=null;

for(Batch bat1:batch\_ass)

{

if (bat1.getBatchId().equals(name))

{

start1=1;

break;

}

}

for(Instructor pil:instructors\_ass)

{

if (pil.getId()==id)

{

t=(SoftSkillsInstructor)pil;

start2=1;

break;

}

}

if(start1==0 || start2==0)

throw new RecordNotFoundException();

if(start1==1 && start2==1)

{

for(Batch bat1:batch\_ass)

{

if (bat1.getBatchId().equals(name))

{

if(bat1.getBizLead().getId()==id)

count++;

}

}

if(count<3)

{

for(Batch bat1:batch\_ass)

{

if(bat1.getBizLead()==null)

{

bat1.setBizLead(t);

System.out.println("Biz lead has been assigned to "+bat1.getBatchId());

break;

}

}

}

else

System.out.println("Biz lead is allocated to three batches already");

}

}

public void get(int check)

{

ArrayList<String> dim=new ArrayList<String>();

SoftSkillsInstructor sim=new SoftSkillsInstructor();

TechnicalInstructor tim=new TechnicalInstructor();

if(check==1)

{

dim=tim.getModules();

}

else

dim=sim.getModules();

System.out.print("The modules are:"+dim);

}

void findHighestBatch()

{

ArrayList<Integer> a\_List=new ArrayList<Integer>();

ArrayList<String> b=new ArrayList<String>();

String batch="";

int max=0,index=0;

int count=0;

for(Batch bt:batch\_ass)

{

for(Trainee t:trainee\_ass)

{

if(t.getBatchId().getBatchId()==bt.getBatchId())

count++;

}

a\_List.add(count);

b.add(bt.getBatchId());

}

for(index=0;index<a\_List.size();index++)

{

if(a\_List.get(index)>max)

{

max=a\_List.get(index);

batch=b.get(index);

}

}

System.out.println("Batch name: "+batch+" and no of trainees: "+max);

}

public static void main(String[] args) throws IOException,InvalidAssignmentException,RecordNotFoundException

{BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

System.out.println("\t\t Welcome to TCS ILP LAB Daigonostic Session");

System.out.println("\t Enter your name");

String name23=br.readLine();

System.out.println("\t Enter your Employment id");

int emp\_no=Integer.parseInt(br.readLine());

System.out.println("Ready for the test!!");

System.out.println("Confirm with Yes or No");

String option=br.readLine();

if(option=="Yes")

{

System.out.println(((char) 27)+"[2J");

}

else

{

System.out.println("TCS to you : You have to give the test else you are fired");

System.out.println("Here your test starts");

}

System.out.println("\t\t\t\tHere is the question for you"+"Mr/Miss"+name23+" ("+emp\_no+")");

System.out.println("Visit 172.0.0.10/ps-1");

System.out.println("Question Solved: Yes or No");

String solve=br.readLine();

if(solve=="Yes"){

System.out.println("Would you like to run your code? Confirm with Yes or No only");

String con=br.readLine();

if(con=="yes"){

System.out.println(((char) 27)+"[2J");

}

else

{

System.out.println("You have to run your code else No salary!!!");

System.out.println("We are waiting for your code");

}

}

else {

System.out.println("No alternate option, you have to solve ");

}

System.out.println(((char) 27)+"[2J");

String ass\_id="",ass\_name="",ass="";

int tid,index;

int k=0;

TrainingCenter tn=new TrainingCenter("Mumbai","m5");

do

{

System.out.println("\t\t\tWelcome to daignostic of java Lab");

System.out.println("\n\t1.Batch to be added \n\t2.Register Trainee \n\t3.Include Instructor \n\t4.TechnicalLeader assignment \n\t5.Assign BizLead\n\t6.Assign batch to Trainee \n\t7.Frequency of Trainees\n\t8.Search Instructor \n\t9.Search Batch with maximum trainees \n\t10.See the modules \n\t11.Exit");

int ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1:System.out.println("Enter the batch id:");

ass\_id=br.readLine();

Stream stream=new Stream("s9");

int mew =tn.addBatch(new Batch(ass\_id,stream));

if(mew==-1)

System.out.println("Batch cannot be added!!");

else

System.out.println("Batch has been added"+"\n The Current no of batches: "+mew);

break;

case 2:System.out.println("Enter the trainee id:");

tid=Integer.parseInt(br.readLine());

System.out.println("Enter name:");

ass\_name=br.readLine();

tn.addTrainee(new Trainee(tid,ass\_name));

break;

case 3: System.out.println("Enter the instructor id:");

tid=Integer.parseInt(br.readLine());

System.out.println("Enter name:");

ass\_name=br.readLine();

System.out.println("1:technical instructor \n 2:soft skill instructor:");

index=Integer.parseInt(br.readLine());

if(index==1)

{

System.out.println("Enter stream code:");

ass=br.readLine();

tn.addInstructor(new TechnicalInstructor(tid,ass\_name,ass));

}

else

tn.addInstructor(new SoftSkillsInstructor(tid,ass\_name));

break;

case 4: System.out.println("Enter the batch id:");

ass\_id=br.readLine();

System.out.println("Enter the instructor id:");

tid=Integer.parseInt(br.readLine());

tn.assignTechLead(ass\_id,tid);

break;

case 5:System.out.println("Enter the batch id:");

ass\_id=br.readLine();

System.out.println("Enter the instructor id:");

tid=Integer.parseInt(br.readLine());

tn.assignBizLead(ass\_id,tid);

break;

case 6:System.out.println("Enter the batch id:");

ass\_id=br.readLine();

System.out.println("Enter the trainee id:");

tid=Integer.parseInt(br.readLine());

tn.assignBatch(tid,ass\_id);

break;

case 9:tn.findHighestBatch();

break;

case 10:System.out.println("1:technical instructor \n 2:soft skill instructor:");

index=Integer.parseInt(br.readLine());

tn.get(index);

break;

case 11:System.out.println("Sorry");

System.exit(0);

default:System.out.println("Invalid option!!");

}

}

while(k==0);

}

}

#### Java Case Study1: Class RelationShip and Inheritance

**Sell Now manages an online advertisement site, where users can advertise about various products they want to sell. Users can post advertisements to sell various products. Users need to provide the category of the product (e.g – Electronics, Books, Textiles, and Vehicle etc), brand of the product (e.g – Nokia, Reebok, etc.). User can anytime remove the advertisement. Write a program to  
a.Register a user  
b.Post an advertisement  
c.Remove an advertisement.**

Code:

import java.io.\*;

class UserRegister{

String name;

UserRegister(){

}

UserRegister(String name)

{

this.name=name;

}

public void setName(String name) {

this.name = name;

}

public String getName()

{

return name;

}

}

class PostAdvertisement{

String userName;

String prodCategory;

String prodBrand;

PostAdvertisement(String userName,String prodCategory,String prodBrand)

{

this.userName=userName;

this.prodCategory=prodCategory;

this.prodBrand=prodBrand;

}

public String getUserName()

{

return userName;

}

public String getProdCategory()

{

return prodCategory;

}

public String getProdBrand()

{

return prodBrand;

}

public void removeA(String userName,String prodCategory,String prodBrand)

{

this.userName=null;

this.prodCategory=null;

this.prodBrand=null;

}

}

public class OnLineAdvertisement {

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

String array\_Name[]=new String[100];

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int opt=0;

do{

System.out.println(" 1-regiter");

System.out.println("2-post an advertisement");

System.out.println("3-remove an advertisement");

int num=Integer.parseInt(br.readLine());

switch(num)

{

case 1: System.out.println("How many users you want to register?");

int number\_register=Integer.parseInt(br.readLine());

UserRegister [] uob=new UserRegister[number\_register];

for(int index1=0;index1<number\_register;index1++)

{

uob[index1]=new UserRegister();

System.out.println(" Enter name");

uob[index1].setName(br.readLine());

array\_Name[index1]=uob[index1].getName();

System.out.println(uob[index1].getName()+ " is registered now");

System.out.println("Want to add more user : enter Y/N");

char check=br.readLine().charAt(0);

if(check=='N')

break;

else if(check=='Y')

continue;

else

break;

}

break;

case 2:

System.out.println(" Enter user name");

String uName=br.readLine();

int j;

for( j=0;j<array\_Name.length;j++)

{

if(array\_Name[j].equals(uName))

{

System.out.println("User Name Is Available");

break;

}

}

System.out.println("Number of advertisements to be posted");

int post\_number=Integer.parseInt(br.readLine());

PostAdvertisement pa[]=new PostAdvertisement[post\_number];

for(int index=0;index<post\_number;index++)

{

String uName1=array\_Name[index];

System.out.println(" Enter product category");

String productCategory=br.readLine();

System.out.println(" Enter brand name");

String productBrand=br.readLine();

pa[index]=new PostAdvertisement(uName1,productCategory,productBrand);

System.out.println(pa[index].getUserName()+ " your advertisement for " + pa[index].getProdCategory()+" with brand "+pa[index].getProdBrand()+" is posted");

System.out.println("Want to advertise for another product : enter Y/N");

char check=br.readLine().charAt(0);

if(check=='N')

break;

else if(check=='Y'||check=='y')

continue;

else

break;

}

break;

case 3: System.out.println(" Enter user name");

String uName1=br.readLine();

System.out.println(" Enter product category");

String productCategory1=br.readLine();

System.out.println(" Enter brand name");

String productBrand1=br.readLine();

PostAdvertisement rmAdv=new PostAdvertisement(uName1,productCategory1,productBrand1);

rmAdv.removeA(uName1,productCategory1,productBrand1);

System.out.println("Record deleted");

break;

default:System.out.println("Not a valid option");

break;

}

}while(opt!=25);

}

}

#### Java Case Study2: Class RelationShip and Inheritance

**1.All Smarts’ university has a library with extensive collection of books. A student can rent up to 5 books at a time. Write a program to manage:  
a.Issuing books  
b.Returning books  
c.Searching for book using title**

import java.io.\*;

public class Library

{

// Issuing Book

public static void issueBook(int array\_bookid[],int n)

{

int lent=array\_bookid.length;

int array\_bookidc[]=new int[100];

for(int g=0;g<lent;g++)

array\_bookidc[g]=array\_bookid[g];

int temp=0;

int f;

for( f=0;f<lent;f++)

{

if(array\_bookidc[f]==n)

{

temp=1;

}

}

if (temp==1){

System.out.println("Book Found and issued");

lent--;

array\_bookid[lent]=array\_bookidc[lent];

}

else{

System.out.println("Element not found");

}

System.out.println("number of books now"+lent);

}

public static void returnBook(int array\_bookid[],int n1)

{

int lent=array\_bookid.length;

int array\_bookidc[]=new int[100];

for(int g=0;g<lent;g++)

array\_bookidc[g]=array\_bookid[g];

int temp1=0;

int f;

for( f=0;f<lent;f++)

{

if(array\_bookidc[f]==n1)

{

temp1=1;

}

}

if (temp1==1){

System.out.println("Book Found and Accepted");

lent++;

array\_bookid[lent]= array\_bookidc[lent];}

else{

System.out.println("Element not found");

}

}

//Find books

public static void findBook(String array\_bookname[],String bName,int array\_bookid[],String array\_author[])

{ int j;

for( j=0;j<array\_bookname.length;j++)

{

if(array\_bookname[j].equals(bName))

{

System.out.println("Book Is Available");

System.out.println("\*\*\*Book Details\*\*\*");

System.out.println("Book Id : "+array\_bookname[j]);

System.out.println("Book Name : "+array\_bookid[j]);

System.out.println("Book Author : "+array\_author[j]);

}

}

System.out.println("Book Is Already Issued or not in library");

}

//Main

public static void main(String[] args) throws IOException

{

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("This is newly built library,\n\t Let's start keeping the books serially");

System.out.println("Enter the number of books you have");

int n=Integer.parseInt(cr.readLine());

int array\_bookid[]=new int[100];

String array\_bookname[]=new String[100];

String array\_author[]=new String[100];

Books [] book\_manage=new Books[n];

System.out.println("\t"+"Fill The Records");

int i;

for( i=0;i<n;i++)

{

book\_manage[i]=new Books();

System.out.println("Enter Book id");

book\_manage[i].setBookId(Integer.parseInt(cr.readLine()));

array\_bookid[i]=book\_manage[i].getBookId();

System.out.println("Enter Book name");

book\_manage[i].setBookName(cr.readLine());

array\_bookname[i]=book\_manage[i].getBookName();

System.out.println("Enter Book Author");

book\_manage[i].setBookAuthor(cr.readLine());

}

System.out.println("You have "+n+"books");

int opt=0;

do{

System.out.println("\t"+" Menu");

System.out.println("\n\t 1.Add Books \n\t 2.Issue Books \n\t 3.Return Books\n\t 4.Find Book");

int choice=Integer.parseInt(cr.readLine());

switch(choice)

{

case 1: System.out.println("How many more books you want to enter");

int morebook=Integer.parseInt(cr.readLine());

int sum=morebook+(book\_manage.length);

Books [] book\_manage1=new Books[sum];

for(int index=0;index<book\_manage.length;index++)

book\_manage1[index]=book\_manage[index];

System.out.println("\t\tFill The Records");

for(int k=book\_manage.length;k<sum;k++)

{

book\_manage1[k]=new Books();

System.out.println("Enter Book id");

book\_manage1[k].setBookId(Integer.parseInt(cr.readLine()));

array\_bookid[k]=book\_manage1[k].getBookId();

System.out.println("Enter Book name");

book\_manage1[k].setBookName(cr.readLine());

array\_bookname[k]=book\_manage1[k].getBookName();

System.out.println("Enter Book Author");

book\_manage1[k].setBookAuthor(cr.readLine());

array\_author[k]=book\_manage1[k].getBookAuthor();

}

System.out.println("you have"+(morebook+book\_manage1.length)+"books");

break;

case 2:

System.out.println("You have "+book\_manage.length);

System.out.println("Enter the book id");

int b\_id=Integer.parseInt(cr.readLine());

issueBook(array\_bookid,b\_id);

break;

case 3:System.out.println("you have"+array\_bookid.length+"books");

System.out.println("Enter the book id to return");

int b\_id1=Integer.parseInt(cr.readLine());

returnBook(array\_bookid,b\_id1);

break;

case 4:

System.out.println("Enter Book Name You Want To Find :");

String b\_name=cr.readLine();

findBook(array\_bookname,b\_name,array\_bookid,array\_author);

break;

default: System.out.println("Invalid Choice");

break;

}

}while(opt!=25);

}

//constructor

}

class Books

{

public int bookId;

public String bookName;

public String bookAuthor;

//getters and setters

/\*public Books(int bookId,String bookName,String bookAuthor){

this.bookId=bookId;

this.bookName=bookName;

this.bookAuthor=bookAuthor;

}\*/

public int getBookId() {

return bookId;

}

public String getBookAuthor() {

return bookAuthor;

}

public void setBookAuthor(String bookAuthor) {

this.bookAuthor = bookAuthor;

}

public void setBookId(int bookId) {

this.bookId = bookId;

}

public String getBookName() {

return bookName;

}

public void setBookName(String bookName) {

this.bookName = bookName;

}

//constructors

/\*public Books(int bookId, String bookName,String bookAuthor) {

super();

this.bookId = bookId;

this.bookName = bookName;

this.bookAuthor = bookAuthor;

}\*/

}

#### Java Case Study3: Class RelationShip and Inheritance

**1.Keep Talking is a leading mobile communication service provider in the country. They offer post-paid mobile connection services across cities. There are various tariff plans they offer to customers. Customers’ calls are charged based on the plan they have opted for. Write a program to:  
a.Calculate bill of a customer  
b.Find number of customer who have opted for a particular tariff plan  
c.Find the connection with the highest bill**

Code:

import java.io.\*;

class Customer{

int customerId;

String customerName;

int plantaken;

int callmade;

public int getCallmade() {

return callmade;

}

public void setCallmade(int callmade) {

this.callmade = callmade;

}

public int getCustomerId() {

return customerId;

}

public void setCustomerId(int customerId) {

this.customerId = customerId;

}

public String getCustomerName() {

return customerName;

}

public void setCustomerName(String customerName) {

this.customerName = customerName;

}

public int getPlantaken() {

return plantaken;

}

public void setPlantaken(int plantaken) {

this.plantaken = plantaken;

}

}

public class PostPaid {

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

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the number of records");

int n=Integer.parseInt(cr.readLine());

Customer [] obcustomer=new Customer[n];

int array\_customerid[]=new int[100];

String array\_customername[]=new String[100];

int array\_plan[]=new int[100];

int array\_callmade[]=new int[100];

int array\_cost[]=new int [100];

System.out.println("Fill The Records");

for(int i=0;i<n;i++)

{

obcustomer[i]=new Customer();

System.out.println("Enter Customer id");

obcustomer[i].setCustomerId(Integer.parseInt(cr.readLine()));

array\_customerid[i]=obcustomer[i].getCustomerId();

System.out.println("Enter Customer name");

obcustomer[i].setCustomerName(cr.readLine());

array\_customername[i]=obcustomer[i].getCustomerName();

System.out.println("Enter Plan you are in");

obcustomer[i].setPlantaken(Integer.parseInt(cr.readLine()));

array\_plan[i]=obcustomer[i].getPlantaken();

System.out.println("Enter no of calls made by customer");

obcustomer[i].setCallmade(Integer.parseInt(cr.readLine()));

array\_callmade[i]=obcustomer[i].getCallmade();

}

for(int k=0;k<array\_callmade.length;k++)

array\_cost[k]=array\_plan[k]\*array\_callmade[k];

int opt=0;

do

{

System.out.println("Menu");

System.out.println("\t1.Bill of a customer" + "\n"+"\t2.No. of Customer for XYZ plan"+"\n"+"\t3.Highest Bill"+"\n");

int choice=Integer.parseInt(cr.readLine());

switch(choice)

{

case 1:int temp=0,pos=0;

System.out.println("Enter the customer id ");

int element=Integer.parseInt(cr.readLine());

for(int i=0;i<array\_customerid.length;i++)

{

if(array\_customerid[i]==element)

{

temp=1;

pos=i;

}

}

if (temp==1)

System.out.println("Customer found"+"\n"+array\_customername[pos]);

else{

System.out.println("Element not found");

break;

}

System.out.println("The Bill to be paid is "+array\_plan[pos]\*array\_callmade[pos]);

break;

case 2:System.out.println("Enter the plan");

int plan\_t=Integer.parseInt(cr.readLine());

int count=0;

for(int j=0;j<array\_plan.length;j++)

{

if(array\_plan[j]==plan\_t)

count++;

}

System.out.println("The no of required plan takens "+count+" times");

break;

case 3:int max=0;

int index=0;

for(int i=0;i<array\_cost.length;i++)

{

if(array\_cost[i]>max)

{

max=array\_cost[i];

index=i;

}

}

System.out.println("The highest bill is "+array\_cost[index]+"and the name is "+array\_customername[index]);

break;

default :System.out.println("You have not entered Valid Option");

break;

}

}while(opt!=20);

}

}

#### Basic Iteration Program Concepts in Java

**1.Find the nth term in Tribonacci series (0, 1, 1, 2, 4, 7, 13…).**

Code:

import java.io.\*;

public class Tribonacci{

static void find\_tri(int n){

int t1,t2,t3,t4,i;

t1=0;t2=1;t3=1;

System.out.println(t1);

System.out.println(t2);

System.out.println(t3);

for(i=0;i<(n-3);i++)

{

t4=t1+t2+t3;

System.out.println(t4);

t1=t2;

t2=t3;

t3=t4;

}

}

public static void main(String args[]) throws IOException

{

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the Limit");

int tribo=Integer.parseInt(cr.readLine());

find\_tri(tribo);

}

}

**2.Find the number of occurrences of given word in a sentence.**

Code:

import java.io.\*;

public class SearchWord {

/\*\*

\* @param args

\*/

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

// TODO Auto-generated method stub

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

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

String sentence=br.readLine();

System.out.print("Enter the word to be searched: ");

String word=br.readLine();

String wordOfSentence ="";

int countWord=0;

sentence=sentence+" ";

for(int index=0;index<sentence.length();index++){

if(sentence.toUpperCase().charAt(index)>='A'&&sentence.toUpperCase().charAt(index)<='Z')

wordOfSentence+=sentence.charAt(index);

else{

if(wordOfSentence.equals(word))

countWord++;

wordOfSentence="";

}

}

System.out.print("The frequency of "+word+" is: "+countWord);

}

}

**3.Find the number of occurrences of a given digit in a number.**

Code:

import java.io.\*;

class Occurance{

static int check(int num,int che){

int rem;

int occ=0,quo=0;

quo=num;

while(quo>0){

rem=quo%10;

quo=quo/10;

if(che==rem)

{

occ++;

}

}

return occ;

}

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

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the digit");

int number=Integer.parseInt(cr.readLine());

System.out.println("Enter the digit you want to search");

int check\_digit=Integer.parseInt(cr.readLine());

int result=check(number,check\_digit);

System.out.println("The required digit occured "+result+" times.");

}

}

**4.Find the number of prime numbers below a given number.**

Code:

import java.io.\*;

class Prime\_no{

static int prime\_num( int num){

boolean isPrime=true;

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

for ( int j = 2; j <= num; j++){

if ( i!=j && i % j == 0 ){

isPrime=false;

break;

}

}

if (isPrime){

System.out.println(i);

}

isPrime=true;

}

return 0;

}

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

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter The Limit Number");

int n=Integer.parseInt(cr.readLine());

prime\_num(n);

}

}

**5.Ezee Shop has an online portal where customers can check about different products. Customers can search for products by giving a key word. Write a program that accepts a product name and a key word, and checks if the key word is present in the given product name.**

Code:

import java.io.\*;

public class SearchTheKeyWord {

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

BufferedReader cr = new BufferedReader(new InputStreamReader(System.in));

System.out.print("\nPlease enter product Name :");

int count = 0;

boolean flag = false;

String product\_name = cr.readLine();

System.out.print("\nPlease enter Key word:");

String key\_word = cr.readLine();

for(int i=0;i < product\_name.length();i++){

for(int j=0,k=i; j<key\_word.length();j++,k++){

if(k < product\_name.length()){

if(product\_name.charAt(k) == key\_word.charAt(j)){

flag = true;

count++;

}

}

}

if(count == key\_word.length()){

flag = true;

break;

}

else{

flag = false;

}

count = 0;

}

if(flag == true){

System.out.print("Yes, the key word is present in the product");

}

else{

System.out.print("No, The key word is NOT present in the product");

}

}

// TODO Auto-generated catch block

}

**6.Tech First has 72 offices spread across 30 countries. Each associate is assigned a seat at their base location. Every seat is coded as <country code><province code><building code><floor code><cubicle code>. The codes are decided by the admin department of Tech First. For example, cubicle number 365 of 3rd floor, Edge Techno Park located at Mumbai, India will be coded as INDMUMEDG003365.   
  
Tech First Solutions has an employee details search application. This will help in fetching various details of an employee, including location of that employee. Write a program that will accept a seat code and return the location code alone from the given seat code.  
Note: Avoid using inbuilt functions to fetch the substring.**

Code:

import java.io.\*;

public class LocationCodeFromSeatCode {

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

BufferedReader cr = new BufferedReader(new InputStreamReader(System.in));

System.out.print("\nPlease enter seat Code :");

String seat\_code=cr.readLine();

for(int i=6;i<=8;i++)

System.out.print(seat\_code.charAt(i));

}

}

**7.Tech First solutions has a savings scheme for its employees. An employee can contribute a certain amount every month. Every year the employee continues with the company and chooses to opt for the savings scheme, he/she needs to contribute 10% more than the previous year’s investment amount. The investment amount can be withdrawn only when the employee retires or resigns from the organization. The starting contribution that is the amount employee has to contribute when he/she joins company, has a maximum limit which is based on the designation of the employee at the time of joining. The company provides an interest of 9% on the investment amount. Interest is calculated on an annual basis and the same is added to the investment. At the end of a year interest is calculated as per the formulae invested amount \* interest rate. Invested amount at any point of time will include the entire contribution employee has made and interest credited in the previous years. Write a program that accepts starting contribution of an employee and number of years of investment and returns the amount currently available at an employees savings scheme account. Assume that calculation is done at year end. An example is given below:  
Starting Contribution: Rs. 1000/-   
Number of years: 3  
Year  
Employee Month Contribution (EC)  
Yearly Contribution (YC=ECX12)  
Amount eligible for interest(EI = CA of previous year+YA)  
Interest at 9%(I)  
Closing Amount at End of Year (CA = EI+I)  
Comments  
1  
1000  
12000  
12000  
1080  
13080  
Interest is 12000 X 0.09, amount at the end of year is amount invested + interested  
2  
1100  
13200  
26280  
2365.2  
28645.2  
Interest is (13080+13200)X0.09 and amount at the end of year is 13080+13200+2365.2  
3  
1210  
14520  
43165.2  
3884.868  
47050.068  
Interest is (28645.2+14520)X0.09 and amount at the end of year is 14520+3884.868+28645.2  
2.**

Code:

import java.io.\*;

public class InterestCalculate {

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

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Please Enter The contributed amount");

double contribution=Double.parseDouble(cr.readLine());

System.out.println("please enter the number of years");

int year=Integer.parseInt(cr.readLine());

double sum=0;

double final\_sum=0;

double total\_contribution=0;

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

sum=contribution\*12;

final\_sum=sum+total\_contribution;

total\_contribution=final\_sum+(final\_sum\*0.09);

contribution= contribution + (0.10\*contribution);

}

System.out.println("total contribution="+total\_contribution);

// TODO Auto-generated method stub

}

}

#### Advanced Iteration Concept Programs in Java

**1.Write a program to split a list of numbers based on condition; say even list and odd list.**

Code:

import java.io.\*;

class EvenOdd

{

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

boolean status=true;

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

int [] arr1=new int[20];

System.out.println("Enter the limit of mixed odd and even array");

int size=Integer.parseInt(cr.readLine());

System.out.println("Enter the values");

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

arr1[i]=Integer.parseInt(cr.readLine());

}

do{

System.out.println();

System.out.println("1 Display Even");

System.out.println("2 Display Odd");

System.out.println("3 Exit");

System.out.print("Choose Option: ");

int choice=Integer.parseInt(cr.readLine());

switch(choice){

case 1:

for(int j=0;j<arr1.length;j++){

if(arr1[j]%2==0){

System.out.println(arr1[j]);

}

}

break;

case 2:

for(int k=0;k<arr1.length;k++){

if(arr1[k]%2!=0){

System.out.println(arr1[k]);

}

}

break;

case 3:

status=true;

System.exit(0);

break;

}

}

while(!status);

}

}

**2.Write a program to merge two given list into one list.**

Code:

import java.io.\*;

import java.io.InputStreamReader;

public class MergeList {

/\*\*

\* @param args

\*/

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

// TODO Auto-generated method stub

int i=0,j=0;

int k=0,p=0,n=0;

int arr1[]=new int[20];

int arr2[]=new int[20];

int arr3[]=new int[50];

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the number of elements:for 1st List");

int number\_input1=Integer.parseInt(cr.readLine());

System.out.println("Enter the number of elements:for 2nd List");

int number\_input2=Integer.parseInt(cr.readLine());

System.out.println("Enter the value for 1st List");

for(i=0;i<number\_input1;i++)

{

arr1[i]=Integer.parseInt(cr.readLine());

}

System.out.println("Enter the value for 2nd List");

for(j=0;j<number\_input2;j++)

{

arr2[j]=Integer.parseInt(cr.readLine());

}

System.out.println("Elements after combining");

for(p=0,k=0;(p<(number\_input1)||k<=(number\_input2));p++,k++){

if(p<number\_input1)

arr3[p]=arr1[p];

arr3[p+number\_input1]=arr2[k];

}

for(n=0;n<(number\_input1+number\_input2);n++)

System.out.println(arr3[n]);

}

}

**3.Write a program to find the availability of given number in the list.**

Code:

import java.io.\*;

public class SearchingElement {

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

int a[]=new int[20];

int i,temp=0,pos=0,element;

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Search an Element in Array");

System.out.println("Enter the number of elements: ");

int number\_input=Integer.parseInt(cr.readLine());

System.out.println("Enter the value for elements: ");

for(i=0;i<number\_input;i++)

{

a[i]=Integer.parseInt(cr.readLine());

}

System.out.println("Enter the searching element: ");

element=Integer.parseInt(cr.readLine());

for(i=0;i<number\_input;i++)

{

if(a[i]==element)

{

temp=1;

pos=i;

}

}

if (temp==1)

System.out.println("Element found"+element+"position="+pos);

else

System.out.println("Element not found");

}

}

**4.Make list of Students having name, roll no., age, score. Write a program to order the student in the class based on score.**

Code:

import java.io.\*;

public class OrderScore {

public static void bubblesort( int score[], int n ,String name[] ){

int i, j,t=0;

String name1="";

for(i = 0; i < n; i++){

for(j = 1; j < (n-i); j++){

if(score[j-1] > score[j]){

t = score[j-1];

name1=name[j-1];

score[j-1]=score[j];

name[j-1]=name[j];

score[j]=t;

name[j]=name1;

}

}

}

}

/\*\*

\* @param args

\*/

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

// TODO Auto-generated method stub

String name[]=new String[10];

int age[]=new int[10];

int roll[]=new int[10];

int score[]=new int[10];

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter the limit of students");

int len=Integer.parseInt(cr.readLine());

System.out.println("Enter The records ");

for(int k=0;k<len;k++)

{

System.out.println("Enter the name");

name[k]=cr.readLine();

System.out.println("Enter the age");

age[k]=Integer.parseInt(cr.readLine());

System.out.println("Enter roll");

roll[k]=Integer.parseInt(cr.readLine());

System.out.println("Enter Score");

score[k]=Integer.parseInt(cr.readLine());

}

bubblesort(score,len,name);

for(int p=0;p<len;p++)

System.out.println(score[p]+" " + "->"+name[p]);

}

}

**6.Make list of Students having name, roll no., age, score. Write a program to group the Students based on the average score [0-50],[50-65],[65-80],[80-100].**

Code:

import java.io.\*;

public class AverageScore {

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

String name[]=new String[5];

int roll[]=new int[5];

int age[]=new int[5];

int score[]=new int[5];

BufferedReader cr=new BufferedReader(new InputStreamReader(System.in));

System.out.println(" Enterh the number of elemets to be entered");

int num=Integer.parseInt(cr.readLine());

for(int k=0;k<num;k++)

{

System.out.println("Enter the name");

name[k]=cr.readLine();

System.out.println("Enter the age");

age[k]=Integer.parseInt(cr.readLine());

System.out.println("Enter roll");

roll[k]=Integer.parseInt(cr.readLine());

System.out.println("Enter Score");

score[k]=Integer.parseInt(cr.readLine());

}

for(int l=0;l<num;l++)

{

if(score[l]>0&&score[l]<=50)

System.out.println(" scorer in 0-50 range "+name[l]);

else if(score[l]>50&&score[l]<=65)

System.out.println(" scorer in 50-65 range "+name[l]);

else if(score[l]>65&&score[l]<=80)

System.out.println(" scorer in 65-80 range "+name[l]);

else

System.out.println(" scorer in 80-100 range "+name[l]);

}

}

}

#### Program to show Composition in Java

Code:

import java.io.\*;

import java.util.\*;

class Department {

/\*\*

\* @param args

\*/

private int Deptid;

private String DeptName;

public Department(int dept,String deptN){

Deptid=dept;

DeptName=deptN;

}

public Department(){

Deptid=200;

DeptName="Management";

}

public int getDeptid() {

return Deptid;

}

public void setDeptid(int deptid) {

Deptid = deptid;

}

public String getDeptName() {

return DeptName;

}

public void setDeptName(String deptName) {

DeptName = deptName;

}

}

class University{

private String uName;

private ArrayList<Department> deptList=new ArrayList<Department>();

public University(String name,int deptId,String deptName){

uName=name;

Department dept1=new Department(deptId,deptName);

deptList.add(dept1);

}

public String getuName() {

return uName;

}

public void setuName(String uName) {

this.uName = uName;

}

public void addDepartment(int deptId,String deptName){

deptList.add(new Department(deptId,deptName));

}

public void showList(){

if(deptList.size()==0)

System.out.println("Sorry list is empty");

else

for(Department dept:deptList)

System.out.println("Department Id is: "+dept.getDeptid()+" Department name is: "+dept.getDeptName());

}

}

public class Composition {

/\*\*

\* @param args

\*/

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

// TODO Auto-generated method stub

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

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

String univ=br.readLine();

University newUniv=new University(univ,100,"Physics");

int opt=0;

do{

System.out.println("\t\tMenu\n\t(1)--Add department\n\t(2)--Show university list\n\t(3)--Exit\n");

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

opt=Integer.parseInt(br.readLine());

switch(opt){

case 1:System.out.print("Enter the department id: ");

int id=Integer.parseInt(br.readLine());

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

String name=br.readLine();

newUniv.addDepartment(id,name);

break;

case 2:newUniv.showList();

break;

case 3:break;

default:System.out.print("Invalid option.");

}

}while(opt!=3);

}

}

#### Example Program to show Aggregation in Java

Code:

import java.io.\*;

import java.util.\*;

class Professor {

/\*\*

\* @param args

\*/

private int profId;

private String profName;

public Professor(int id,String name){

profId=id;

profName=name;

}

public int getProfId() {

return profId;

}

public void setProfId(int profId) {

this.profId = profId;

}

public String getProfName() {

return profName;

}

public void setProfName(String profName) {

this.profName = profName;

}

}

class University1{

private String uName;

private ArrayList<Professor> profList=new ArrayList<Professor>();

public University1(String name,Professor prof){

uName=name;

profList.add(prof);

}

public String getuName() {

return uName;

}

public void setuName(String uName) {

this.uName = uName;

}

public void addProfessor(Professor prof){

profList.add(prof);

}

public void showList(){

if(profList.size()==0)

System.out.println("Sorry list is empty");

else

for(Professor prof:profList)

System.out.println("Department Id is: "+prof.getProfId()+" Department name is: "+prof.getProfName());

}

}

public class Aggregation {

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

// TODO Auto-generated method stub

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

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

String univ=br.readLine();

Professor prof=new Professor(100,"S.Chatterjee");

University1 newUniv=new University1(univ,prof);

int opt=0;

do{

System.out.println("\t\tMenu\n\t(1)--Add department\n\t(2)--Show university list\n\t(3)--Exit\n");

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

opt=Integer.parseInt(br.readLine());

switch(opt){

case 1:System.out.print("Enter the professor id: ");

int id=Integer.parseInt(br.readLine());

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

String name=br.readLine();

Professor newProf=new Professor(id,name);

newUniv.addProfessor(newProf);

break;

case 2:newUniv.showList();

break;

case 3:break;

default:System.out.print("Invalid option.");

}

}while(opt!=3);

}

}

#### Write a program To Find The Top And Least Scorer In The Class.

**Make list of Students having name, roll no., age, score. Write a program to find the top and least scorer in the class.**

Code:

import java.io.\*;

import java.util.\*;

class findScore1

{

String name;

int rollNo;

int age;

int score;

findScore1(String name,int rollNo,int age,int score)

{

this.name=name;

this.rollNo=rollNo;

this.age=age;

this.score=score;

}

public String getName()

{

return name;

}

public int getrollNo()

{

return rollNo;

}

public int getAge()

{

return age;

}

public int getScore()

{

return score;

}

}

#### Java Case Study: Hospital Management System

**Care Well Hospital is a leading super specialty hospital in the city. They have various departments such as ENT, Cardiology etc. In every department doctors are available for OP consultation. Patients need to take appointment for consulting any doctor. First time a patient visits the hospital patient details will be recorded and an OP ID will be provided. For taking an appointment patient needs to provide OP ID to the clerk. Write a program to  
a.Register new patient details  
b.Book an appointment for a specific doctor  
c.Book an appointment for any available doctor in a department**

Code:

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.\*;

public class CareWell

{

ArrayList <Patient> pList=new ArrayList<Patient>();

ArrayList <Department> dList=new ArrayList<Department>();

ArrayList<Appointment> a=new ArrayList<Appointment>();

public void addPatient(Patient p)

{

pList.add(p);

}

public void addDept(Department d)

{

dList.add(d);

}

public void addDoc(Doctor d)

{

for(Department dept:dList)

{

if(dept.getDepName()==d.getDepName())

dept.addDoctor(d);

}

}

public void addAppoint(Appointment ap)

{

int flag=0;

for(Appointment aa:a)

{

if(aa.getDc().equals(ap.getDc()))

{

if(aa.getDt()==ap.getDt())

{

System.out.println("Doctor not free!!");

flag=1;

break;

}

}

}

if(flag==0)

{

System.out.println("The appointment is done!");

System.out.println("Appointment details are:"+"\nName: "+ap.getPt().getName()+" Doctor's Name: "+ap.getDc().getDocName()+" Date: "+ap.getDt());

a.add(ap);

}

}

public void addAppoint(Patient p,int dt)

{

int flag=0;

for(Appointment aa:a)

{

if(aa.getDt()==dt)

{

for(Department d:dList)

{

for(Doctor dc:d.getDocList())

{

if (!(dc.getDocName().equals(aa.getDc())))

{

Appointment ap=new Appointment(p,dc,dt);

a.add(ap);

System.out.println("The appointment is done!");

System.out.println("Appointment details are:"+"\nName: "+ap.getPt().getName()+" Doctor's Name: "+ap.getDc().getDocName()+" Date: "+ap.getDt());

flag=1;

break;

}

}

}

}

}

if(flag==0)

{

for(Appointment aa:a)

{

if(aa.getDt()!=dt)

{

Doctor dc=aa.getDc();

Appointment ap=new Appointment(p,dc,dt);

a.add(ap);

System.out.println("The appointment is done!");

System.out.println("Appointment details are:"+"\nName: "+ap.getPt().getName()+" Doctor's Name: "+ap.getDc().getDocName()+" Date: "+ap.getDt());

}

}

}

}

public static void main(String[] args) throws IOException

{

int da=0,i,j=0,k=0,flag=0;

String name="",name1="";

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

CareWell cw=new CareWell();

cw.addDept(new ENT("ENT"));

cw.addDept(new Cardiology("Cardiology"));

cw.addDoc(new Doctor("x","ENT"));

cw.addDoc(new Doctor("y","ENT"));

cw.addDoc(new Doctor("z","Cardiology"));

System.out.println("\t\t\tWELCOME TO CARE WELL!!");

for(Department d:cw.dList)

{

for(Doctor dc:d.getDocList())

{

System.out.println("\nDoctor's name: "+dc.getDocName()+" Department's Name: "+dc.getDepName());

}

}

do

{

System.out.println("\n\t\t\tEnter the choice:");

System.out.println("1.Add patient\n2.Take appointment with specific doctor\n3.Take appointment with available doctor\n4.Exit");

int ch=Integer.parseInt(br.readLine());

switch(ch)

{

case 1: System.out.println("Enter the name:");

name=br.readLine();

j++;

cw.addPatient(new Patient(j,name));

System.out.println("\nOp ID: "+j);

break;

case 2: System.out.println("Enter the Patient's name:");

name1=br.readLine();

System.out.println("Enter the Doctor's name:");

name=br.readLine();

System.out.println("Enter date:");

da=Integer.parseInt(br.readLine());

Patient p=new Patient();

flag=0;

for(i=0;i<cw.pList.size();i++)

{

if(cw.pList.get(i).getName().equals(name1))

{

p=cw.pList.get(i);

}

}

for(Department d:cw.dList)

{

for(Doctor dc:d.getDocList())

{

if (dc.getDocName().equals(name))

{

cw.addAppoint(new Appointment(p,dc,da));

flag=1;

break;

}

}

}

if(flag==0)

System.out.println("Doctor does not exists!!");

break;

case 3: System.out.println("Enter the Patient's name:");

name1=br.readLine();

System.out.println("Enter date:");

da=Integer.parseInt(br.readLine());

Patient pc=new Patient();

for(i=0;i<cw.pList.size();i++)

{

if(cw.pList.get(i).getName().equals(name1))

{

pc=cw.pList.get(i);

}

}

cw.addAppoint(pc,da);

break;

case 4:System.out.println("Thank You!!");

System.exit(0);

default:System.out.println("Invalid option!!");

}

}

while(k==0);

}

}

class Department

{

ArrayList <Doctor> docList=new ArrayList<Doctor>();

private String depName;

public ArrayList<Doctor> getDocList() {

return docList;

}

public void setDocList(ArrayList<Doctor> docList) {

this.docList = docList;

}

public String getDepName() {

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

public void addDoctor(Doctor dc)

{

docList.add(dc);

}

}

class Doctor

{

private String docName;

private String depName;

public String getDocName() {

return docName;

}

public void setDocName(String docName) {

this.docName = docName;

}

public String getDepName() {

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

public Doctor(String docName,String depName)

{

this.depName=depName;

this.docName=docName;

}

}

class ENT extends Department

{

String depName;

public String getDepName() {

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

public ENT(String depName)

{

this.depName=depName;

}

}

class Cardiology extends Department

{

String depName;

public String getDepName() {

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

public Cardiology(String depName)

{

this.depName=depName;

}

}

class Patient

{

private int opId;

private String name;

public Patient()

{

}

public Patient(int opId,String name)

{

this.name=name;

this.opId=opId;

}

public int getOpId() {

return opId;

}

public void setOpId(int opId) {

this.opId = opId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

class Appointment

{

//private Date dt;

private Patient pt;

private int dt;

private Doctor dc;

public Patient getPt() {

return pt;

}

public Doctor getDc() {

return dc;

}

public void setDc(Doctor dc) {

this.dc = dc;

}

public void setPt(Patient pt) {

this.pt = pt;

}

public int getDt() {

return dt;

}

public void setDt(int dt) {

this.dt = dt;

}

public Appointment(Patient pt,Doctor dc,int dt)

{

this.dc=dc;

this.pt = pt;

this.dt=dt;

}

}

Same programs in comments someone wrote this code

import java.io.\*;

import java.util.\*;

class Doctor

{

int doctorId;

String Speciality;

boolean status=true;;

String name;

public Doctor(int doctorId, String speciality, boolean status, String name) {

this.doctorId = doctorId;

Speciality = speciality;

this.status = status;

this.name = name;

}

public boolean isStatus() {

return status;

}

public void setStatus(boolean status) {

this.status = status;

}

public int getDoctorId() {

return doctorId;

}

public String getSpeciality() {

return Speciality;

}

public String getName() {

return name;

}

}

class Patient

{

int opid;

String name;

String prefDoc;

String dept;

public Patient(int opid, String name, String dept) {

this.opid = opid;

this.name = name;

this.dept = dept;

}

public int getOpid() {

return opid;

}

public void setOpid(int opid) {

this.opid = opid;

}

public void setPrefDoc(String prefDoc) {

this.prefDoc = prefDoc;

}

public String getName() {

return name;

}

}

class Department

{

ArrayList<Doctor> doc= new ArrayList<Doctor>();

}

class Cardiology extends Department

{

String depName;

public String getDepName() {

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

public ArrayList<Doctor> getList()

{

return doc;

}

void addList(Doctor ob)

{

doc.add(ob);

}

}

class Ent extends Department{

String depName;

public String getDepname()

{

return depName;

}

public void setDepName(String depName) {

this.depName = depName;

}

void addList(Doctor ob)

{

doc.add(ob);

}

}

class Appointment

{

int id;

String docName;

String PatientName;

String date;

String dept;

public Appointment(int id, String docName, String patientName, String date,String dept) {

this.id = id;

this.docName = docName;

PatientName = patientName;

this.date = date;

this.dept=dept;

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getDocName() {

return docName;

}

public String getPatientName() {

return PatientName;

}

}

class Manage

{

ArrayList<Doctor> doc=new ArrayList<Doctor>();

ArrayList<Patient>ptn=new ArrayList<Patient>();

ArrayList<Appointment>apn=new ArrayList<Appointment>();

ArrayList<Cardiology> crd=new ArrayList<Cardiology>();

ArrayList<Ent> en=new ArrayList<Ent>();

void addDoc(int doctorId, String speciality, boolean status, String name)

{

Doctor ob=new Doctor(doctorId,speciality,status,name);

doc.add(ob);

//System.out.println("Doctor added to list");

if(speciality.equals("cardio"))

{

Cardiology ob1=new Cardiology();

ob1.setDepName(speciality);

ob1.addList(ob);

crd.add(ob1);

}

else if(speciality.equals("ent"))

{

Ent ob2=new Ent();

ob2.setDepName(speciality);

ob2.addList(ob);

en.add(ob2);

}

else

System.out.println("No such department exists");

}

void addPatient(int opid, String name, String dept)

{

Patient ob3=new Patient(opid, name, dept);

ptn.add(ob3);

System.out.println("Patients details added");

}

void addAppoint(int id, String docName, String patientName, String date,String dept)

{

Appointment ob4=new Appointment(id,docName,patientName,date,dept);

apn.add(ob4);

System.out.println("appiontment is fixed ");

}

void setStatus(int id,boolean status1)

{

for(Doctor dctr:doc)

{

if(dctr.getDoctorId()==id)

{

dctr.setStatus(status1);

}

}

}

}

public class Hospital {

/\*\*

\* @param args

\*/

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

// TODO Auto-generated method stub

int option=0;

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

Manage obj=new Manage();

obj.addDoc(1, "ent", false, "Banerjee");

obj.addDoc(2,"cardio",true,"das");

obj.addDoc(3,"cardio",true , "Majumder");

obj.addDoc(4,"ent",true,"Sharma");

obj.addDoc(5,"cardio",true,"Dutta");

do{

System.out.println("\*\*\*\*\*\*\*\*\*Doctor list :\*\*\*\*\* ");

for(Doctor dctr:obj.doc)

{

System.out.println(dctr.getName()+" "+dctr.getDoctorId()+" "+dctr.getSpeciality()+" "+dctr.isStatus());

}

System.out.println("\*\*\*\*\*\*\*\*Welcome Enter 1.to add doctor 2. to fix appointments 3. to fix doctor status ");

System.out.println("4. to add patients 5.to view patients 6.to view appointment 7.exit");

System.out.println("Enter the Option");

option=Integer.parseInt(br.readLine());

switch(option)

{

case 1:

{

boolean flag=true;

System.out.println("Enter id");

int doctorId=Integer.parseInt(br.readLine());

for(Doctor dc:obj.doc)

{

if(dc.getDoctorId()==doctorId)

{

flag=false;

break;

}

}

if(flag==true)

{

System.out.println("Enter Name");

String name=br.readLine();

System.out.println("Enter speciality cardio/ent");

String speciality=br.readLine();

boolean status=true;

obj.addDoc(doctorId, speciality, status, name);

}

else

System.out.println("Id already exists");

break;

}

case 2:

{

String patientName="";

boolean flag2=false;

int appId=0;

System.out.println("Enter patient id");

int ptnId=Integer.parseInt(br.readLine());

for(Patient ptn:obj.ptn)

{

if(ptn.getOpid()==ptnId)

{

patientName=ptn.getName();

break;

}

}

System.out.println("Enter 1 for preference ");

int op=Integer.parseInt(br.readLine());

switch(op)

{

case 1:

{

System.out.println("Enter doctor's name");

String name1=br.readLine();

for(Doctor dtr:obj.doc)

{

if(dtr.getName().equals(name1) && dtr.isStatus()==true)

{

String docName=dtr.getName();

appId=obj.apn.size()+1;

String dept=dtr.getSpeciality();

String date="1/19/2012";

obj.addAppoint(appId, docName, patientName, date, dept);

flag2=true;

break;

}

}

if(flag2==true)

System.out.println("Doctor found id is"+appId);

else

System.out.println("Doctor not found");

break;

}

default:

{

System.out.println("Enter speciality");

String depName=br.readLine();

if(depName.equals("ent"))

{

for(Doctor dct2:obj.doc)

{

if(dct2.getSpeciality().equals(depName) && dct2.isStatus()==true)

{

String doctorName=dct2.getName();

String speciality=dct2.getSpeciality();

String date="1/19/2012";

appId=obj.apn.size()+1;

obj.addAppoint(appId, doctorName, patientName, date, speciality);

}

}

}

}

}

break;

}

case 3:

{

System.out.println("Enter doctors id");

int id=Integer.parseInt(br.readLine());

System.out.println("Enter true /false");

boolean status1=Boolean.parseBoolean(br.readLine());

obj.setStatus(id, status1);

break;

}

case 4:

{

int ptid=0;

boolean flag=false;

System.out.println("Enter name");

String name=br.readLine();

for(Patient ptt:obj.ptn)

{

if(ptt.getName().equals(name))

{

flag=true;

ptid=ptt.getOpid();

break;

}

}

if(flag==false)

{

ptid=obj.ptn.size()+1;

}

System.out.println("Patients id"+ptid);

System.out.println("Enter the department");

String dept=br.readLine();

obj.addPatient(ptid, name, dept);

break;

}

case 5:

{

for(Patient ptn1:obj.ptn)

System.out.println(ptn1.getName()+" "+ptn1.getOpid());

break;

}

case 6:

{

for(Appointment ap:obj.apn)

{

System.out.println(ap.getId()+" "+ap.getDocName()+" "+ap.getPatientName());

}

break;

}

case 7:

break;

default:

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

}

}while(option!=7);

}

}

#### Routine (Methods) Concepts Programs in Java

**1.Find the maximum of three numbers**

Code:

public class Maximum

{

public static void main(String args[])

{

int x=Integer.parseInt(args[0]);

int y=Integer.parseInt(args[1]);

int z=Integer.parseInt(args[2]);

System.out.println("Max: "+calMax(x,y,z));

}

static int calMax(int x,int y,int z)

{

int max=0;

if(x>y)

{ if(x>z)

max=x;

else

max=z;

}

else

{ if(y>z)

max=y;

else

max=z;

}

return max;

}

}

**2 .Write a program to check whether the input alphabet is a vowel or not.**

Code:

public class Vowel

{

public static void main(String args[])

{

String c=args[0].toLowerCase();

char alph=c.charAt(0);

checkAlpha(alph);

}

static void checkAlpha(char alph)

{

switch(alph)

{ case 'a':

case 'e':

case 'i':

case 'o':

case 'u':System.out.println("vowel!!"); break;

default:System.out.println("Consonant!!");

}

}

}

**3.Develop a program, that accepts a deposit amount and calculates amount of interest the deposit amount earns in an year. The bank pays a flat 4% for deposits of up to Rs.1000, a flat 4.5% per year for deposits of up to Rs.5000, and a flat 5% for deposits of more than Rs.5000.**

Code:

public class Interest

{

public static void main(String args[])

{

int amt=Integer.parseInt(args[0]);

System.out.print("Interest: " + calInterest(amt));

}

static double calInterest(int amt)

{

double r=0;

if (amt<=1000)

r=0.04;

if (amt>1000 && amt<=5000)

r=0.045;

if(amt>5000)

r=0.05;

double interest=r\*amt;

return interest;

}

}

**4.Develop the program, which accepts the gross pay and produces the amount of tax owed. For a gross pay of $240 or less, the tax is 0%; for over $. 240 and $. 480 or less, the tax rate is 15%; and for any pay over $480, the tax rate is 28%.**

Code:

public class GrossPay

{

public static void main(String args[])

{

int gp=Integer.parseInt(args[0]);

System.out.print("Tax Owed: " + calTax(gp));

}

static double calTax(int gp)

{

double tax=0;

if (gp<=240)

tax=0;

if (gp>240 && gp<=480)

tax=0.15;

if (gp>480)

tax=0.28;

double owedTax=tax\*gp;

return owedTax;

}

}

**5.Some credit card companies pay back a small portion of the charges a customer makes over a year. A particular credit card company's pay back policy is as follows:  
1.0.25% for charges up to Rs. 500.  
2.0.50% for the next Rs.1000 (that is, the portion between Rs. 500 and Rs. 1500),  
3.0.75% for the next Rs.1000 (that is, the portion between Rs. 1500 and Rs. 2500),  
4.1.0% for charges above Rs2500.  
Thus, a customer who charges Rs. 400 a year receives Rs.1.00, which is 0.25 · 1/100 · 400, and one who charges Rs1, 400 a year receives Rs. 5.75, which is 1.25 = 0.25 · 1/100 · 500 for the first Rs. 500 and 0.50 · 1/100 · 900 = 4.50 for the next Rs. 900. Determine by hand the pay backs amount for a customer who charged Rs. 2000 and one who charged Rs. 2600.  
Define the program, which accepts a charge amount and computes the corresponding pay back amount.**

Code:

public class Charge

{

public static void main(String args[])

{

int charge=Integer.parseInt(args[0]);

System.out.print("Pay Back amount: " +calculatePay(charge));

}

static double calculatePay(int charge)

{

double pay=0;

if (charge<=500)

pay=0.0025\*charge;

if (charge>500 && charge<=1500)

pay=0.0025\*500+0.0050\*(charge-500);

if (charge>1500 && charge<=2500)

pay=0.0025\*500+0.0050\*1000+0.0075\*(charge-1500);

if (charge>2500)

pay=0.0025\*500+0.0050\*1000+0.0075\*1000+0.01\*(charge-2500);

return pay;

}

}

**6.Implement a method that returns the day of the week for a given day (1..31), month (1..12) and year.  
The day of the week of dates between March 1900 and February 2100 can be calculated as follows:  
1.Calculate the total number of days from 1900/1/1 to the given date (see below, for details).  
2.Divide this number by 7, the remainder is the day of the week (0 - sunday, 1 - monday, etc)  
To calculate the total number of days:  
1.Subtract 1900 from the given year and multiply the result by 365. Add the missing leaps years by adding (year - 1900) / 4.  
2.If the given year is a leap year and the month is January or February, subtract 1 from the previous result.  
3.Add no of days of the given year till the given month (in case of February always add 28, because the additional day for a leap year would have been already added).**

Code:

public class Days

{

public static void main(String args[])

{

int d=Integer.parseInt(args[0]);

int m=Integer.parseInt(args[1]);

int y=Integer.parseInt(args[2]);

int flag=checkLeapYear(y);

int x=y-1900;

x=x\*365;

x=x+((y-1900)/4);

if ((flag==1)&&(m==1 || m==2))

x=x-1;

x=x+noOfDays(m,d);

dayToDay(x%7);

}

static int checkLeapYear(int y)

{

int flag=0;

if(y%100==0)

{ if(y%400==0)

flag=1;

}

else

{ if(y%4==0)

flag=1;

}

return flag;

}

static void dayToDay(int rem)

{

switch(rem)

{ case 0:System.out.print("Sunday");

break;

case 1:System.out.print("Monday");

break;

case 2:System.out.print("Tuesday");

break;

case 3:System.out.print("Wednesday");

break;

case 4:System.out.print("Thursday");

break;

case 5:System.out.print("Friday");

break;

case 6:System.out.print("Saturday");

break;

}

}

static int noOfDays(int m,int d)

{ int days=0;

switch(m)

{ case 1:days=d;

break;

case 2:days=31+d;

break;

case 3:days=31+28+d;

break;

case 4:days=31+28+31+d;

break;

case 5:days=31+28+31+30+d;

break;

case 6:days=31+28+31+30+31+d;

break;

case 7:days=31+28+31+30+31+30+d;

break;

case 8:days=31+28+31+30+31+30+31+d;

break;

case 9:days=31+28+31+30+31+30+31+31+d;

break;

case 10:days=31+28+31+30+31+30+31+31+30+d;

break;

case 11:days=31+28+31+30+31+30+31+31+30+31+d;

break;

case 12:days=31+28+31+30+31+30+31+31+30+31+30+d;

}

return days;

}

}

**7. Write a program to automate the following loan policy.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Age category | Gender | Profession | Personal assets | Loan amount eligible |
| 16 -25 | M /F | Self-Employed | >25000 | 10000 |
| Professional | 15000 |  |  |  |
| 26 - 40 | M | SelfEmployed / Professional | > 40000 | 25000 |
| F | 30000 |  |  |  |
| 41 - 60 | M / F | SelfEmployed / Professional | > 50000 | 40000 |
| > 60 | M/F | Self Employed | > 25000 | 35000 – Age \* 100 |
| Retired | 25000 – Age \* 100 |  |  |  |

**Write a program that accepts age, gender, job status and assets, and return the eligible loan amount.**

Code:

public class Loan

{

public static void main(String args[])

{

int age=Integer.parseInt(args[0]);

String c=args[1].toUpperCase();

char gender=c.charAt(0);

String job=args[2];

int asset=Integer.parseInt(args[3]);

System.out.println("loan amount: " + calLoan(age,gender,job,asset));

}

static int calLoan(int age,char gender,String job,int asset)

{

int loan=0;

if ((age>=16)&&(age<=25))

{ if(gender=='M' || gender=='F')

if(asset>25000)

{ if(job.equals("selfemployed"))

loan=10000;

else

loan=15000;

}

}

if((age>=26)&&(age<=40))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(asset>40000)

{ if(gender=='M')

loan=25000;

else

loan=30000;

}

}

if((age>=41)&&(age<=60))

{ if ((job.equals("selfemployed"))||(job.equals("professional")))

if(gender=='M' || gender=='F')

if(asset>40000)

loan=40000;

}

if(age>60)

{ if(gender=='M' || gender=='F')

if(asset>25000)

if(job.equals("selfemployed"))

loan=35000-age\*100;

else

loan=25000-age\*100;

}

return loan;

}

}