public class Company {

private String name;

private Partner[] arrPartner;

private int nbp;

Company(String name, int size){

this.name = name;

arrPartner = new Partner[size];

nbp = 0;

}

public boolean addPartner(Partner p){

if(nbp >= arrPartner.length)

return false;

if(p instanceof ProjectBased)

arrPartner[nbp++] = new ProjectBased((ProjectBased)p);

else if(p instanceof Exclusive)

arrPartner[nbp++] = new Exclusive((Exclusive)p);

else

arrPartner[nbp++] = new Partner(p);

return true;

}

public void displayAll(){

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

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

arrPartner[i].display();

}

public int countProjectBased(){

int count=0;

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

if(arrPartner[i] instanceof ProjectBased)

count++;

return count;

}

public Partner[] getHighRevenue(double s){

Partner[] res = new Partner[nbp];

int j=0;

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

if( arrPartner[i].calculateRevenue() > s)

res[j++]= arrPartner[i];

return res;

}

public Exclusive[] getExclusive(int n){

Exclusive[] res = new Exclusive[nbp];

int j=0;

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

if(arrPartner[i] instanceof Exclusive)

if(((Exclusive)arrPartner[i]).getNbYears()>n)

res[j++]=(Exclusive) arrPartner[i];

return res;

}

}

///////////////////

public class Exclusive extends Partner{

private int nbYears;

Exclusive(String name, int nbEmps) {

super(name);

this.nbYears = nbEmps;

}

Exclusive(Exclusive e){

super(e);

this.nbYears = e.nbYears;

}

public int getNbYears() {

return nbYears;

}

public void display(){

super.display();

System.out.println(" Number of Years: "+nbYears);

}

public double calculateRevenue(){

return nbYears \* 10000 + 300000;

}

}

///////////////////////

public class Partner {

private String name;

Partner(String name)

{

this.name = name;

}

Partner(Partner p){

name = p.name;

}

public String getName() {

return name;

}

public void display(){

System.out.print("name: "+name);

}

public double calculateRevenue()

{

return 100000;

}

}

///////////////////////////

public class ProjectBased extends Partner{

private int nbProjects ;

ProjectBased(String name, int nbProjects ) {

super(name);

this.nbProjects = nbProjects ;

}

ProjectBased(ProjectBased p){

super(p);

nbProjects= p.nbProjects;

}

public int getNbProjects() {

return nbProjects;

}

public void display(){

super.display();

System.out.println(" number of projects: "+nbProjects);

}

public double calculateSalary(){

return nbProjects \* 200000;

}

}

//////////////////////////

public class Test {

public static void main(String[] args) {

Partner stc = new Partner("STC");

Exclusive mobily = new Exclusive("mobily",10);

ProjectBased sale = new ProjectBased("sale",19);

Company elm = new Company("ELM", 10);

elm.addPartner(stc);

elm.addPartner(mobily);

elm.addPartner(sale);

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

elm.displayAll();

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

System.out.println("Number of Project Based = "+elm.countProjectBased());

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

Partner[] a = elm.getHighRevenue(1000000);

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

if(a[i] != null)

a[i].display();

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

Exclusive[] m = elm.getExclusive(10);

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

if(m[i] != null)

m[i].display();

}

}