using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assignment2

{

class Program

{

public delegate void Delegateobj();

static void Main(string[] args)

{

CEO Obj = new CEO("raj ",12,10000);

Console.WriteLine(Obj.calcNetsalary());

Console.WriteLine(Obj.deptno);

}

}

public interface IdFunction

{

void update();

void Delete();

void insert();

}

public abstract class Employee

{

protected string name;

protected short deptNo = 0;

protected int empNo;

protected decimal Basicsalary;

static int id = 0;

public abstract decimal calcNetsalary();

protected decimal basic;

public abstract decimal baSic

{

set;

get;

}

public abstract short deptno

{

set;

get;

}

public Employee(string name = "noname", short deptno = 10, decimal basic = 1000)

{

id++;

this.empNo = id;

this.name = name;

this.basic = basic;

this.deptNo = deptno;

}

}

public class Manager : Employee, IdFunction

{

public string designation;

public string Designation

{

set

{

if (value == "md")

designation = value;

else

Console.WriteLine("invalid credintial");

}

get

{

return designation;

}

}

public Manager(string name = "noname", short deptno = 10, decimal basic = 1000, string designation = "md") : base(name, deptno, basic)

{

this.name = name;

this.basic = basic;

this.deptNo = deptno;

}

public override decimal baSic

{

set

{

if (value > 1000 && value < 100000)

baSic = value;

else

Console.WriteLine("invalid credential");

}

get

{

return baSic;

}

}

private short deptNo;

public override short deptno

{

set

{

if (value > 0)

value = deptno;

else

{

Console.WriteLine("invalid credintial");

}

}

get

{

return deptno;

}

}

public override decimal calcNetsalary()

{

decimal tax = 500;

decimal pf = 1500;

decimal netsalary = basic - pf - tax;

return netsalary;

}

void IdFunction.update()

{

Console.WriteLine("interface method");

}

void IdFunction.Delete()

{

throw new NotImplementedException();

}

void IdFunction.insert()

{

throw new NotImplementedException();

}

}

class Generalmanager : Manager, IdFunction

{

public string perks;

public string Perks

{

set{

if (value =="")

perks = value;

else

{

Console.WriteLine("invalid credintial");

}

}

get

{

return perks;

}

}

public Generalmanager(string name = "noname", short deptno = 10, decimal basic = 10000, string designation = "md", string perks = " ") : base(name, deptno, basic, designation)

{

this.name = name;

this.basic = basic;

this.deptNo = deptno;

}

public override decimal baSic

{

set

{

if (value > 1000 && value < 100000)

baSic = value;

else

Console.WriteLine("invalid credential");

}

get

{

return baSic;

}

}

public override sealed decimal calcNetsalary()

{

decimal tax = 500;

decimal pf = 1500;

decimal netsalary = basic - pf - tax;

return netsalary;

}

}

class CEO : Employee, IdFunction

{

public CEO(string name = "noname", short deptno = 12, decimal basic = 10000):base(name,deptno,basic)

{

this.name = name;

this.basic = basic;

this.deptNo = deptno;

}

public override decimal baSic

{

set

{

if (value > 1000 && value < 100000)

baSic = value;

else

Console.WriteLine("invalid credential");

}

get

{

return baSic;

}

}

public override decimal calcNetsalary()

{

decimal tax = 500;

decimal pf = 1500;

decimal netsalary = basic - pf - tax;

return netsalary;

}

void IdFunction.update()

{

Console.WriteLine("this is ceo class");

}

void IdFunction.Delete()

{

throw new NotImplementedException();

}

void IdFunction.insert()

{

throw new NotImplementedException();

}

private short deptNo;

public override short deptno

{

set

{

if (value > 0)

value = deptno;

else

{

Console.WriteLine("invalid credintial");

}

}

get

{

return deptno;

}

}

}

}