1. A. Count the number of vowels in a given string

using System;

class CountVowels

{

static int Vowels(string str)

{

int i = 0;

int Vowels\_count = 0;

for (i = 0; i < str.Length; i++)

{

if ((str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u') ||

(str[i] == 'A' || str[i] == 'E' || str[i] == 'I' || str[i] == 'O' || str[i] == 'U'))

Vowels\_count++;

}

return Vowels\_count;

}

public static void Main()

{

int vowels = 0;

string str = "";

Console.Write("Enter the string: ");

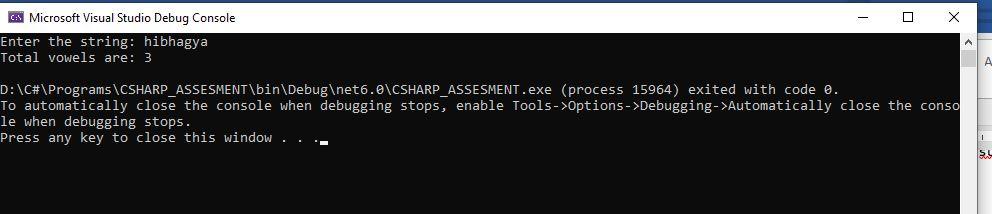
str = Console.ReadLine();

vowels = Vowels(str);

Console.WriteLine("Total vowels are: " + vowels);

}

}



1.B. Create a ATM class that would allow the end user to perform cash withdrawal and deposit operations in sequence not by choice and print the balance in the account after every operation.

Bank CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Bank

{

private string name;

private int accno;

private decimal balance;

private decimal amount;

private string acctype;

public Bank(string name, int accno,string type, decimal balance, decimal amount )

{

this.name = name;

this.accno = accno;

this.balance = balance;

this.Acctype = Acctype;

this.amount = amount;

}

public string Name { get => name; set => name = value; }

public int Accno { get => accno; set => accno = value; }

public decimal Amount { get => amount; set => amount = value; }

public decimal Balance { get => balance; set => balance = value; }

public string Acctype { get => acctype; set => acctype = value; }

}

}

 WITHDRAW CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.AccessControl;

using System.Text;

using System.Threading.Tasks;

using System.Xml.Linq;

namespace CSHARP\_ASSESMENT

{

internal class Withdraw\_amount:Bank

{

private int withdrawamount;

public int Withdrawamount { get => withdrawamount; set => withdrawamount = value; }

public Withdraw\_amount(string name, int accno, string acctype, decimal balance, decimal amount ,int withdrawamount) :base( name,accno,acctype,balance,amount) { }

internal decimal Withdraw(int withdrawamount)

{

if ((withdrawamount <= 0) || (withdrawamount > Amount))

{

Console.WriteLine("Enter valid amount");

return Amount;

}

else

{

Amount = Amount - withdrawamount;

Console.WriteLine($"Withdraw amount: {withdrawamount}");

Console.WriteLine("Balance amount is: " + Amount);

return Amount;

}

//throw new NotImplementedException();

}

}

}

 DEPOSIT CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Deposit\_amount:Bank

{

private int depositamount;

public Deposit\_amount(string name, int accno, string type, decimal balance, decimal amount, int depositamount) : base(name, accno, type, balance, amount)

{

this.Depositamount = Depositamount;

}

public int Depositamount { get => depositamount; set => depositamount = value; }

public void Deposit()

{

/\*if (depositamount <= 0)

{

Console.WriteLine("Enter valid amount");

}

else\*/

{

Console.WriteLine(+depositamount);

Amount = Amount + this.Depositamount;

Console.WriteLine($"Deposit amount: {depositamount}");

Console.WriteLine("Amount deposited from account" + depositamount);

Console.WriteLine("Balance amount is" + Amount);

}

}

internal void Deposit(int depositamount)

{

//throw new NotImplementedException();

if (depositamount <= 0)

{

Console.WriteLine("Enter valid amount");

}

else

{

Console.WriteLine(+depositamount);

Amount = Amount + depositamount;

Console.WriteLine($"Deposit amount: {depositamount}");

Console.WriteLine("Balance amount is : " + Amount);

}

}

internal void Deposit(decimal balance, int depositamount)

{

//throw new NotImplementedException();

if (depositamount <= 0)

{

Console.WriteLine("Enter valid amount");

}

else

{

Console.WriteLine(+depositamount);

balance = balance + depositamount;

Console.WriteLine($"Deposit amount: {depositamount}");

Console.WriteLine("Balance amount is : " + balance);

}

}

}

}

MAIN CLASS:

using CSHARP\_ASSESMENT;

using System;

public class Program

{

public static void Main(string[] args)

{

Console.WriteLine("Welcome");

Console.WriteLine("Enter your name");

string name=Console.ReadLine();

Console.WriteLine("Enter your Account NUmber");

int accno=Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter your Account Type");

string acctype=Console.ReadLine();

decimal amount=100000;

decimal balance = 0;

//withdraw

Console.WriteLine("Enter amount to withdraw");

int withdrawamount=Convert.ToInt32(Console.ReadLine());

Withdraw\_amount w= new Withdraw\_amount( name, accno, acctype, balance, amount ,withdrawamount);

balance = w.Withdraw(withdrawamount);

Console.WriteLine(+balance);

//deposit

Console.WriteLine("Enter amount to deposit");

int depositamount = Convert.ToInt32(Console.ReadLine());

Deposit\_amount d = new Deposit\_amount(name, accno, acctype, balance, amount, depositamount);

d.Deposit(balance,depositamount);

Console.WriteLine("Thank you");

}

}



1. Implement hierarchical inheritance in a Company with employees of different departments segregated in the child classes. Write method to calculate the employees bonus (assume bonus is different for every department and is calculated from basic salary). (basic = 10000 bonus=10% bonus amt=10000\*10% = 1000)

DEPARTMENT CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Department

{

private int employeeId;

private string employeeName;

public int EmployeeId { get => employeeId; set => employeeId = value; }

public string EmployeeName { get => employeeName; set => employeeName = value; }

public Department(string employeeName, int employeeId)

{

this.EmployeeName = employeeName;

this.EmployeeId = employeeId;

}

}

}

SALES CLASS:

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Sales : Department

{

double salary;

static double bonus = 0.8;

public Sales(double salary, string emp\_name, int emp\_id) : base(emp\_name, emp\_id)

{

this.Salary = salary;

}

public double Salary { get => salary; set => salary = value; }

public double calculate\_bonus()

{

return this.Salary \* (bonus / 100);

}

}

}

MARKETING CLASS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Marketing : Department

{

double salary;

static double bonus = 0.6;

public Marketing(double salary, string emp\_name, int emp\_id) : base(emp\_name, emp\_id)

{

this.Salary = salary;

}

public double Salary { get => salary; set => salary = value; }

public double calculate\_bonus()

{

return this.Salary \* (bonus / 100);

}

}

}

ACCOUNTS CLASS:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CSHARP\_ASSESMENT

{

internal class Accounts : Department

{

double salary ;

static double bonus = 12.4;

public Accounts(double salary, string emp\_name, int emp\_id) : base(emp\_name, emp\_id)

{

this.Salary = salary;

}

public double Salary { get => salary; set => salary = value; }

public double calculate\_bonus()

{

return this.Salary \* (bonus / 100);

}

}

}

PROGRAM CLASS:

using CSHARP\_ASSESMENT;

using System;

public class Program

{

public static void Main(string[] args)

{

Console.WriteLine("Enter employee id");

int employeeId=Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter employee name");

string employeeName=Console.ReadLine();

double salary=0;

Console.WriteLine("Choose your department");

Console.WriteLine("1.Accounts" +

"2.Sales" +

"3.Marketing");

int ch=Convert.ToInt32(Console.ReadLine());

switch(ch)

{

case 1:

Console.WriteLine("DEPARTMENT: Accounts");

Console.WriteLine("salary :");

salary = Convert.ToDouble(Console.ReadLine());

Accounts accounts = new Accounts(salary,employeeName, employeeId);

double ab=accounts.calculate\_bonus();

Console.WriteLine("Bonus :" + ab);

break;

case 2:

Console.WriteLine("DEPARTMENT: Sales");

Console.WriteLine("salary :");

salary = Convert.ToDouble(Console.ReadLine());

Sales sales = new Sales(salary, employeeName, employeeId);

double sb= sales.calculate\_bonus();

Console.WriteLine("Bonus is :" + sb);

break;

case 3:

Console.WriteLine("DEPARTMENT: Marketing");

Console.WriteLine("salary :");

salary = Convert.ToDouble(Console.ReadLine());

Marketing marketing = new Marketing(salary, employeeName, employeeId);

double mb=marketing.calculate\_bonus();

Console.WriteLine("Bonus is :" + mb);

break;

default:

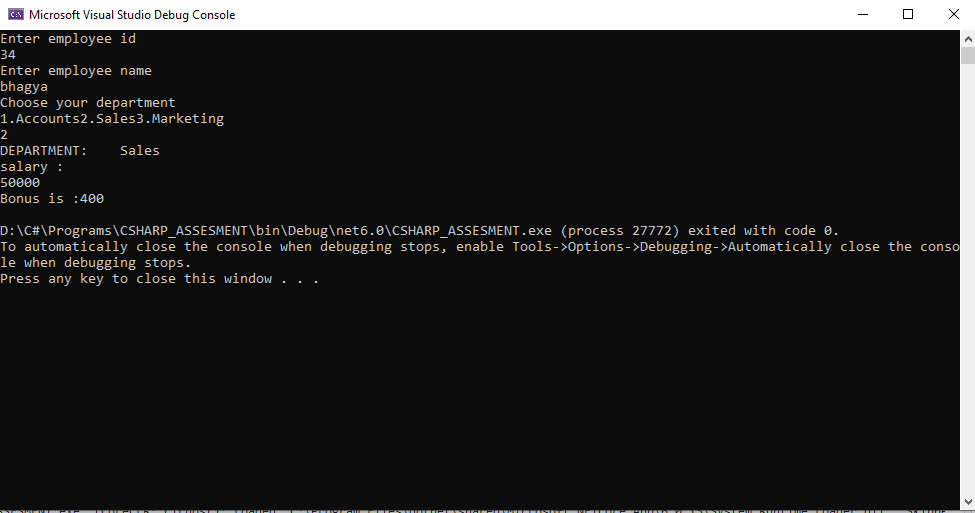
Console.WriteLine("Enter valid department");

break;

}

}

}



3.

a.

Create a class named SpaceStation that is abstract

On that abstract class, add a abstract method called FireLaser

Create a derived class called DeathStar that implements the FireLaser method to write "Pew Pew" to the Console followed by a new line

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assessment

{

abstract class Spacestation

{

public abstract void FireLaser();

}

}

b.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Assessment

{

internal class DeathStar : SpaceStation

{

public override void FireLaser()

{

Console.WriteLine("Pew Pew");

}

}

}

c.

DeathStar ds = new DeathStar();

ds.FireLaser();