PART -1 CONSOLE APPLICATIONS

AIM 1:**Write a program by using Console applications (Use ReadLine() &** **WriteLine()functions):WAP to enter Employee Name, Age, Joining Date, BASIC, DA, HRA, PF, calculate Grosspay & Net Pay and display it with Employee information.**

**Program :-**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Practical\_1

{

static void Main(string[] args)

{

int n;

Console.WriteLine("ENTER NO. OF EMPLOYEES IN COMPANY :"); n = Convert.ToInt32(Console.ReadLine());

Employee[] e = new Employee[n];

Console.WriteLine("\n");

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

{

e[i] = new Employee();

Console.WriteLine("EMPLOYEE NO :" + (i + 1));

Console.WriteLine("\n");

e[i].getdata();

}

Console.WriteLine("\n");

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

{

Console.WriteLine("\n");

Console.WriteLine("DETAILS OF " + (i + 1) + "th EMPLOYEE IS :"); e[i].display();

}

}

}

class Employee

{

String name,date;

int age;

double BASIC, HRA, PF, GROSS, NET,DA;

internal void getdata()

{

Console.WriteLine("ENTER NAME OF EMPLOYEE :"); name = Console.ReadLine(); Console.WriteLine("ENTER AGE :");

age = Convert.ToInt32(Console.ReadLine()); Console.WriteLine("ENTER JOINING DATE :"); date =Console.ReadLine(); Console.WriteLine("ENTER BASIC SALARY :"); BASIC = Convert.ToDouble(Console.ReadLine());

}

internal void display()

{

DA = 1.36 \* BASIC;

PF = 0.12 \* BASIC;

HRA = 0.2 \* BASIC;

GROSS = DA + HRA + BASIC;

NET = GROSS - PF;

Console.WriteLine("\n");

Console.WriteLine("NAME :"+name);

Console.WriteLine("AGE :"+age);

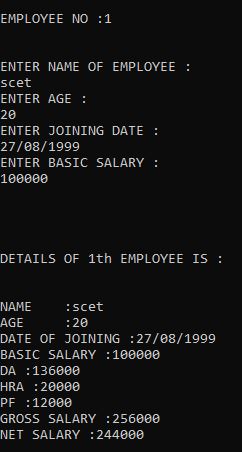
Console.WriteLine("DATE OF JOINING :"+ date); Console.WriteLine("BASIC SALARY :"+ BASIC); Console.WriteLine("DA :"+ DA); Console.WriteLine("HRA :"+ HRA); Console.WriteLine("PF :"+ PF); Console.WriteLine("GROSS SALARY :"+ GROSS); Console.WriteLine("NET SALARY :"+NET); Console.ReadKey();

}

}

}

**Output :-**



**AIM 2:Write C# menu based program to do the following**

**Convert binary to decimal**

1. **Convert decimal to hexadecimal**
2. **Convert decimal to binary**
3. **Convert decimal to octal.**

**Create a separate class for each functionality and put each class in a separate namespace in the same program.**

**PROGRAM:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Practical\_2

{

static void Main(string[] arg)

{

int opt;

string ch="y";

Console.Write("A MENU DRIVEN PROGRAM FOR NUMBER CONVERSION SYSTEM:\n");

while(ch.Equals("y"))

{

Console.Write("\nHERE ARE THE OPTIONS.. :\n");

Console.Write("1-Convert binary to decimal.\n2-Convert decimal tohexadecimal.\n3-Convertdecimal to binary.\n4-Convert decimal to octal.\n");

Console.Write("\nENTER YOUR CHOICE :");

opt = Convert.ToInt32(Console.ReadLine());

switch (opt)

{

case 1: Console.Write("Binary To Decimal\n");

binary\_decimal.binary\_decimal o1 = new binary\_decimal.binary\_decimal();

o1.b\_d();

break;

case 2: Console.Write("Decimal To Hexadecimal\n");

decimal\_hexadecimal.Program22 o2 = new decimal\_hexadecimal.Program22();

o2.d\_h();

break;

case 3: Console.Write("Decimal to Binary\n");

decimal\_binary.Program23 o3 = new decimal\_binary.Program23();

o3.d\_b();

break;

case 4: Console.Write("Decimal to Octal\n");

decimal\_octal.Program24 o4 = new decimal\_octal.Program24();

o4.d\_o();

break;

}

Console.Write("Do You Want To Repeat ??(Y/N)\n"); ch=Console.ReadLine();

}

Console.ReadLine();

}

}

}

namespace binary\_decimal

{

class binary\_decimal

{

internal void b\_d()

{

int num, binary\_val, decimal\_val = 0, base\_val = 1, rem; Console.Write("Enter a Binary Number(1s and 0s) : ");

num = Convert.ToInt32(Console.ReadLine()); /\* maximum five digits \*/

binary\_val = num;

while (num > 0)

{

rem = num % 10;

decimal\_val = decimal\_val + rem \* base\_val;

num = num / 10;

base\_val = base\_val \* 2;

}

Console.Write("The Binary Number is : " + binary\_val); Console.Write("\nIts Decimal Equivalent is : " + decimal\_val); Console.ReadLine();

}

}

}

namespace decimal\_hexadecimal

{

class Program22

{

internal void d\_h()

{

int decimalNumber, quotient;

int i = 1, j, temp = 0;

char[] hexadecimalNumber = new char[100];

char temp1;

Console.WriteLine("Enter a Decimal Number :");

decimalNumber = Convert.ToInt32(Console.ReadLine());

quotient = decimalNumber;

while (quotient != 0)

{

temp = quotient % 16;

if (temp < 10)

temp = temp + 48;

else

temp = temp + 55;

temp1 = Convert.ToChar(temp);

hexadecimalNumber[i++] = temp1;

quotient = quotient / 16;

}

Console.Write("The Decimal Number is : " + decimalNumber);

Console.Write("\nIts HexaDecimal Equivalent is : "); for (j = i - 1; j > 0; j--)

Console.Write(hexadecimalNumber[j]);

Console.Read();

}

}

}

namespace decimal\_binary

{

class Program23

{

internal void d\_b()

{

int num, no, i = 0,quot;

int[] bin= new int[100];

Console.Write("Enter a Decimal Number : ");

num = Convert.ToInt32(Console.ReadLine());

no = num;

float rem = 0;

while (num >= 1)

{

quot = num / 2;

rem = (num % 2);

num = quot;

bin[i] = Convert.ToInt32(rem);

i++;

}

Console.Write("The Decimal Number is : " + no);

Console.WriteLine("\nIts Binary Equivalent is :");

for (int j = i-1; j >= 0; j--)

{

Console.Write(+bin[j]);

}

Console.Read();

}

}

}

namespace decimal\_octal

{

class Program24

{

internal void d\_o()

{

int decimalNumber, quotient, i = 1, j;

int[] octalNumber = new int[100];

Console.WriteLine("Enter a Decimal Number :");

decimalNumber = Convert.ToInt32(Console.ReadLine());

quotient = decimalNumber;

while (quotient != 0)

{

octalNumber[i++] = quotient % 8;

quotient = quotient / 8;

}

Console.Write("The Decimal Number is : " + decimalNumber);

Console.Write("\nIts Octal Equivalent is : ");

for (j = i - 1; j > 0; j--)

Console.Write(octalNumber[j]);

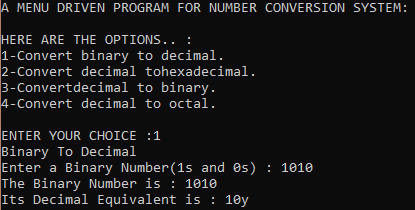
Console.Read();

}

}

}

**OUTPUT:**



**AIM 3:Create console applications to implement following C# Concepts.**

1. **Constructor & Destructor**
2. **Inheritance**
3. **Method Overloading**
4. **Operator Overloading**

**PROGRAM:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Practical\_3

{

static void Main(string[] args)

{

Console.WriteLine("---------------Demo of Constructor---------------\n");

Prac\_3\_1.Complex\_no c1 = new Prac\_3\_1.Complex\_no(20, 25);

c1.ShowXY();

Console.WriteLine("\n---------------Demo of Inheritance---------------\n");

Prac\_3\_2.Triangle t1 = new Prac\_3\_2.Triangle();

Prac\_3\_2.Triangle t2 = new Prac\_3\_2.Triangle();

t1.Width = 4.0;

t1.Height = 4.0;

Console.WriteLine("Info for t1: ");

t1.ShowDim();

Console.WriteLine("Area is " + t1.Area() + "\n\n");

t2.Width = 8.0;

t2.Height = 12.0;

Console.WriteLine("Info for t2: ");

t2.ShowDim();

Console.WriteLine("Area is " + t2.Area());

Console.WriteLine("\n---------------Demo of Method Overloading---------------\n");

Prac\_3\_3.Method\_overloading mthover = new Prac\_3\_3.Method\_overloading();

mthover.Area(2);

mthover.Area(5, 40);

mthover.Area(20.5);

Console.WriteLine("\n---------------Demo of Operator Overloading---------------\n");

Prac\_3\_4.overloadpgm d = new Prac\_3\_4.overloadpgm();

d++;

Console.WriteLine("The value of 1st variable after unary operation: "+d.value);

d++;

Console.WriteLine("The value of 1st variable after unary operation: " + d.value);

Prac\_3\_4.overloadpgm g = new Prac\_3\_4.overloadpgm();

g++;

Console.WriteLine("The value of 2nd variable after unary operation: " + g.value);

Prac\_3\_4.overloadpgm t = d + g;

Console.WriteLine("The value after addition of 1st and 2nd variable is : "+t.value);

Console.WriteLine("\n---------------Demo of Destructor---------------");

GC.Collect();

Console.ReadLine();

}

}

}

namespace Prac\_3\_1

{

class Complex\_no

{

private int x;

private int y;

public Complex\_no(int i, int j)

{

x = i;

y = j;

}

public void ShowXY()

{

Console.WriteLine(x + "i+" + y);

}

~Complex\_no()

{

Console.WriteLine("Deleted...");

Console.Read();

}

}

}

namespace Prac\_3\_2

{

class Shape

{

public double Width;

public double Height;

public void ShowDim()

{

Console.WriteLine("Width and height are " + Width + " and " + Height);

}

}

class Triangle : Shape

{

public double Area()

{

return Width \* Height / 2;

}

}

}

namespace Prac\_3\_3

{

class Method\_overloading

{

public void Area(int side)

{

int squarearea = side \* side; Console.WriteLine("Area of Square :" + squarearea);

}

public void Area(int length, int breadth)

{

int rectarea = length \* breadth; Console.WriteLine("Area of Rectangle :" + rectarea);

}

public void Area(double radius)

{

double circlearea = 3.14 \* radius \* radius; Console.WriteLine("Area of Circle :" + circlearea);

}

}

}

namespace Prac\_3\_4

{

class overloadpgm

{

public int value;

public static overloadpgm operator +(overloadpgm a, overloadpgm b)

{

overloadpgm overloadpg = new overloadpgm();

overloadpg.value = a.value + b.value;

return overloadpg;

}

public static overloadpgm operator ++(overloadpgm c)

{

c.value++;

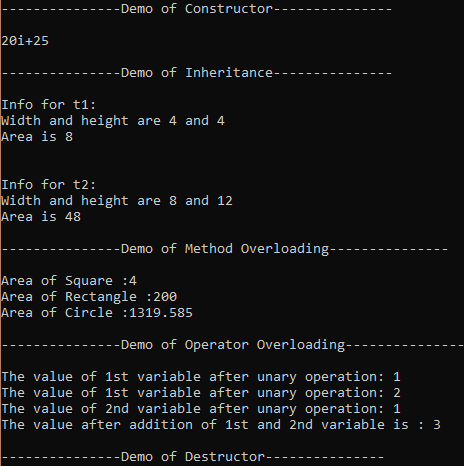
return c;

}

}

}

**OUTPUT:**



**AIM 4:Create a class to demonstrate static property by counting numbering of objects created.**

**PROGRAM:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Program\_4

{

Person s1 = new Person();

Console.WriteLine("Count: "+Person.count);

Person s2 = new Person();

Person s3 = new Person();

Person s4 = new Person();

Console.WriteLine("Count: " + Person.count);

Person s5 = new Person();

Console.WriteLine("Count: " + Person.count);

Console.ReadLine();

}

public class Person

{

static int count = 0;

public Person()

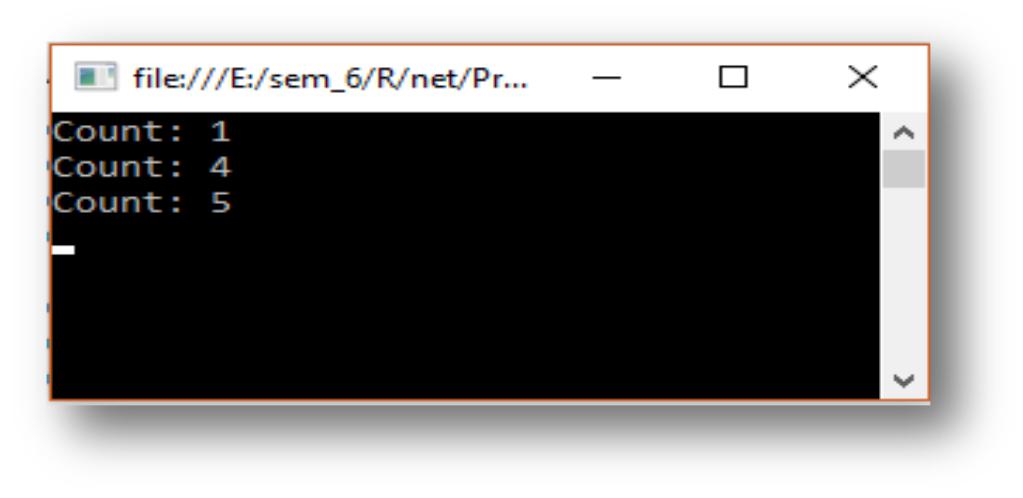
{

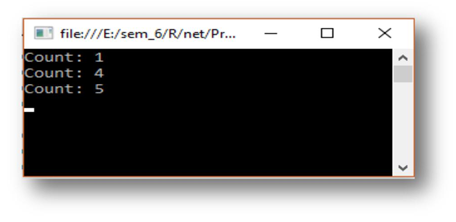
count++;

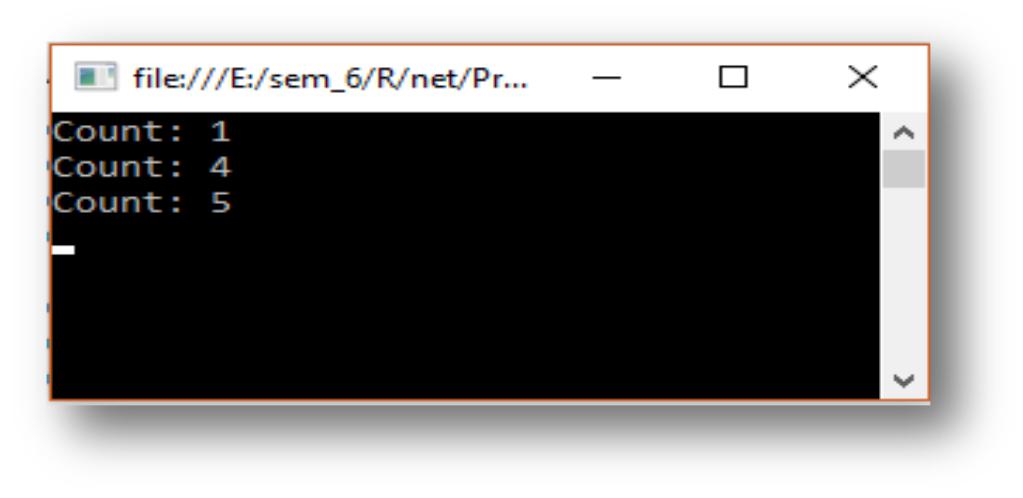
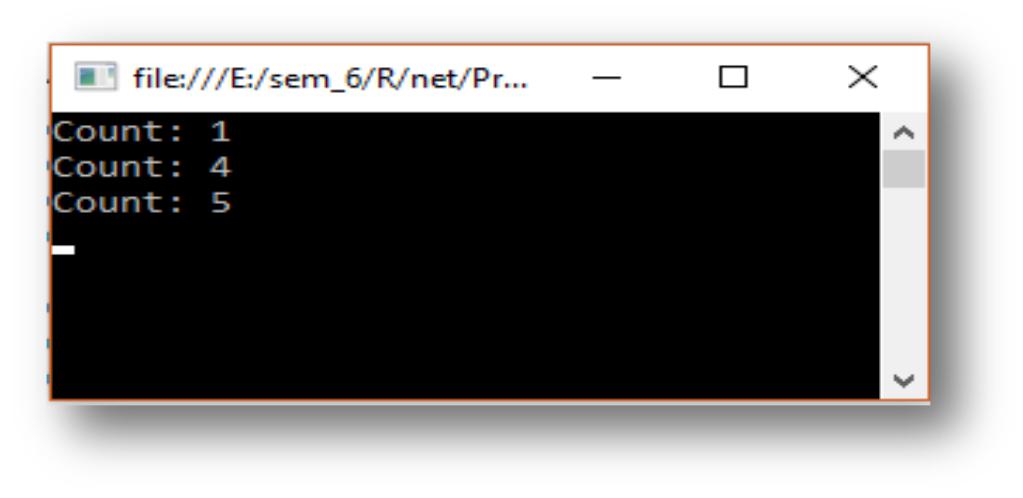
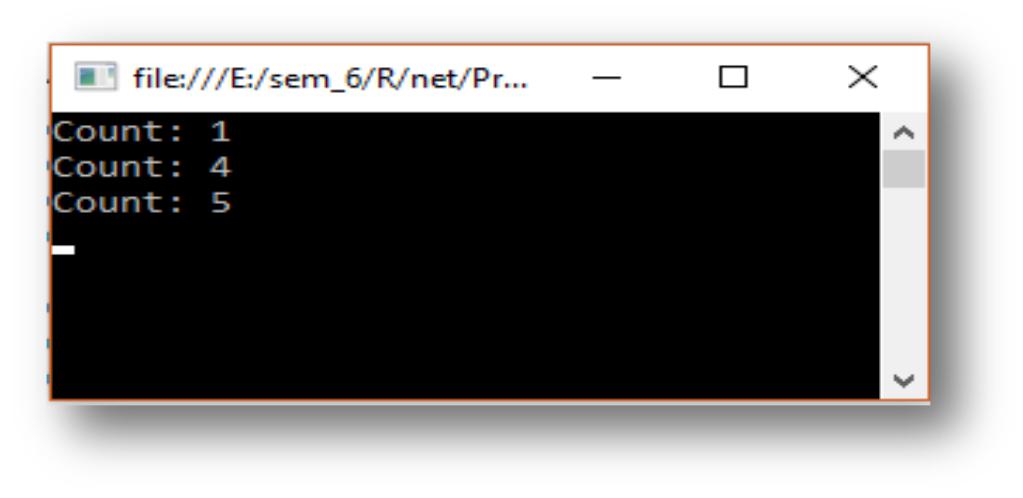
}

}

}

**OUTPUT:**

****

****

**AIM 5:Create a class MyStringIndexer with data members title, author, subject.**

**It should contain**

1. **Constructor with arguments**
2. **Indexer with string index**

**Create a class IndexerDemo for main function. Write code to get and set values of any data members of MyStringIndexer class.**

**Program :**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Practical\_5

{

static void Main(string[] args)

{

MyStringIndexer ind\_demo = new MyStringIndexer("Developing Web Applications", "Ralph Moseley", "Web Technology");

Console.WriteLine("Title is: " + ind\_demo[0]);

Console.WriteLine("Auther Name: " + ind\_demo[1]);

Console.WriteLine("Subject: " + ind\_demo[2]); ind\_demo[0] = "Introduction to dot net"; Console.WriteLine("Title is: " + ind\_demo[0]); Console.ReadLine();

}

}

class MyStringIndexer

{

string title,auther,subject;

public MyStringIndexer(string title, string auther, string subject)

{

this.title = title;

this.auther = auther;

this.subject = subject;

}

public string this[int index]

{

get

{

if (index == 0)

return title;

else if (index == 1)

return auther;

else if (index == 2)

return subject;

return "";

}

set

{

if (index == 0)

{

title = value;

}

else if (index == 1)

{

auther = value;

}

else if (index == 2)

{

subject = value;

}

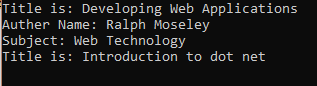
}

}

}

}

**OUTPUT :**



**AIM 6:Create a C# program to test Attributes and Reflection.**

**PROGRAM:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApp

{

class Program

{

static void Main(string[] args)

{

Type tpy = typeof(Developer);

Console.WriteLine("Class Attribute");

Attribute[] attr = Attribute.GetCustomAttributes(tpy);

foreach (Attribute a in attr)

{

MyAttribute devloper = (MyAttribute)a;

Console.WriteLine(devloper.getPublisherName() + "\t" + devloper.version);

}

Console.WriteLine("Method Attribute");

Type [] type=new Type[1];

type[0]=typeof(Int32);

Attribute[] attr\_method = Attribute.GetCustomAttributes(tpy.GetMethod("getData",type));

foreach (Attribute a in attr\_method)

{

MyAttribute devloper = (MyAttribute)a;

Console.WriteLine(devloper.getPublisherName() + "\t" + devloper.version);

}

Console.ReadLine();

}

}

[MyAttribute("Parth Roy", version = 2.0f)]

[MyAttribute("Royal Dreams")]

class Developer

{

int x;

[MyAttribute("Keval Navadiya", version = 5.12f)]

public int getData()

{

return x;

}

[MyAttribute("Keval Navadiya", version = 5.12f)]

public int getData(int a,int y)

{

return x;

}

[MyAttribute("Parth Roy", version = 1.20f)]

public int getData(int x)

{

return x;

}

[MyAttribute("Darshit Akbari")]

public void putData(int x)

{

this.x=x;

}

}

[AttributeUsage (AttributeTargets.Class|AttributeTargets.Method,AllowMultiple=true)]

class MyAttribute : Attribute

{

public string publisher;

public float version;

public MyAttribute()

{

publisher = "Royal Dreams";

version = 0.0f;

}

public MyAttribute(string str)

{

this.publisher = str;

}

public string getPublisherName() { return (string)publisher; }

}

}

**OUTPUT:**

