

# C# and .NET Frameworks

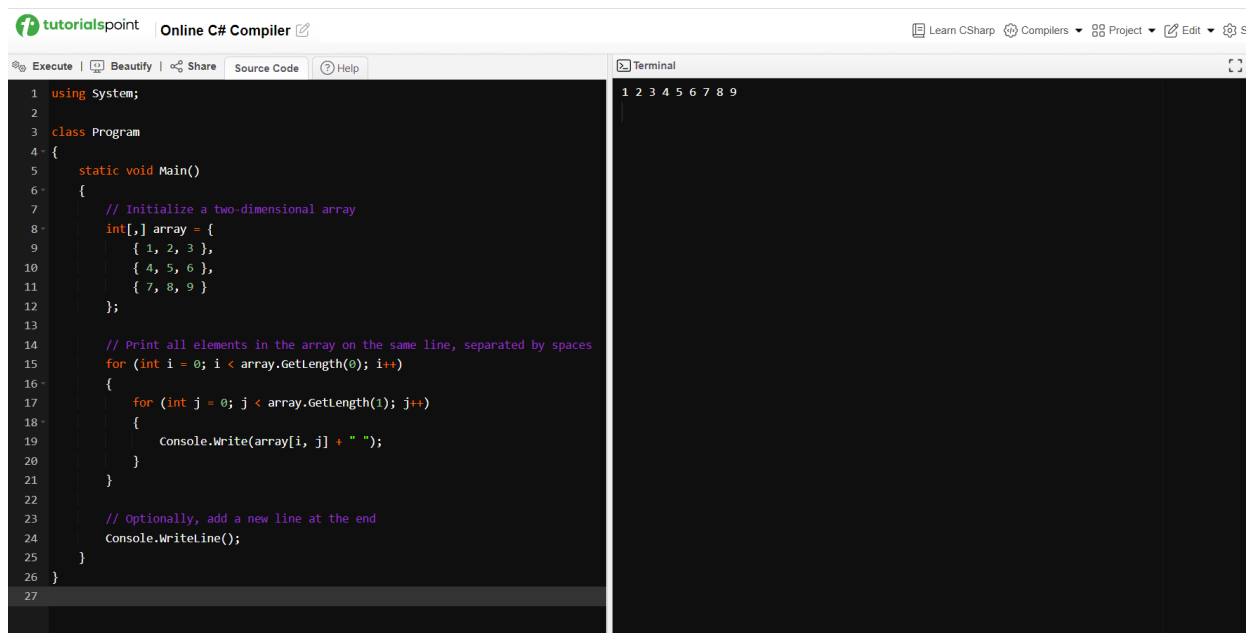
## Assignment 1

1. Develop the C# program to initialize two dimensional array and print all the elements of the array on the same line separated with space.

### AIM:

To create a C# program that initializes a two-dimensional array and prints all the elements in the array on the same line, separated by spaces.

### PROGRAM AND OUTPUT:



The screenshot shows the 'Online C# Compiler' interface. The left pane displays the C# code, and the right pane shows the terminal output.

```
1 using System;
2
3 class Program
4 {
5     static void Main()
6     {
7         // Initialize a two-dimensional array
8         int[,] array = {
9             { 1, 2, 3 },
10            { 4, 5, 6 },
11            { 7, 8, 9 }
12        };
13
14        // Print all elements in the array on the same line, separated by spaces
15        for (int i = 0; i < array.GetLength(0); i++)
16        {
17            for (int j = 0; j < array.GetLength(1); j++)
18            {
19                Console.Write(array[i, j] + " ");
20            }
21        }
22
23        // Optionally, add a new line at the end
24        Console.WriteLine();
25    }
26 }
27
```

The terminal output on the right shows the numbers 1 through 9 printed on a single line, separated by spaces: 1 2 3 4 5 6 7 8 9.

2. Aravind wants to apply for competitive exam. He needs to know whether he is eligible to apply. The eligibility criteria is given below:
  - Age should be greater than 18 years, but not more than 30.

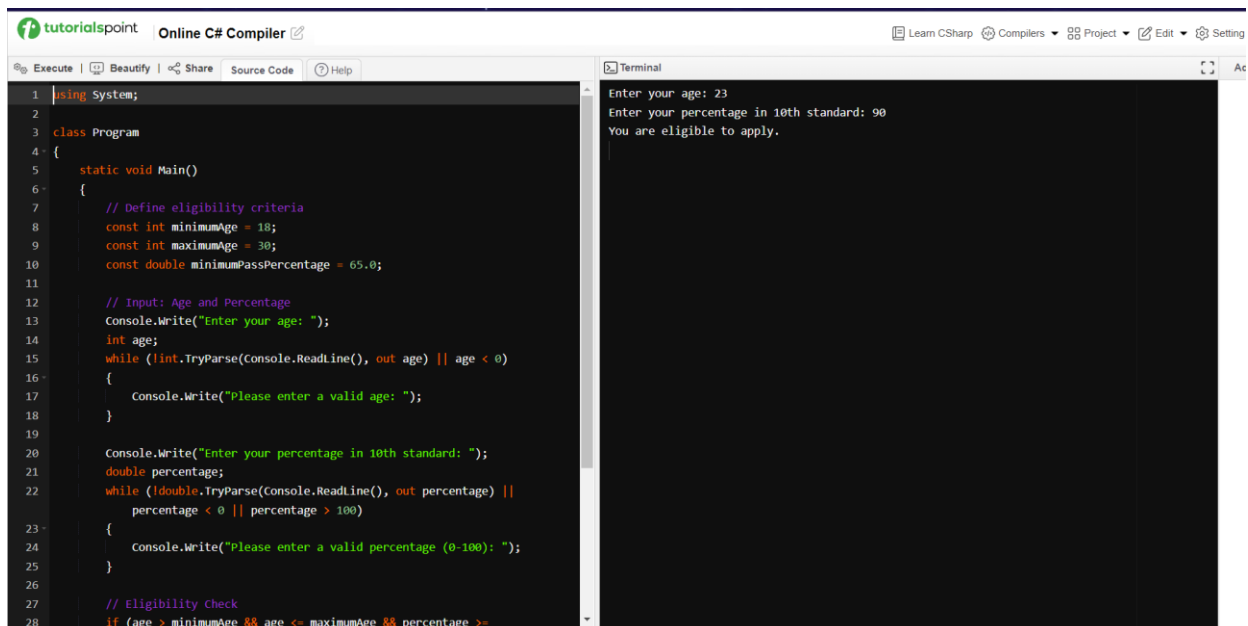
- The candidate should have passed 10 std with a minimum pass percentage of 65.

Design the C# program to help him to know his eligibility. If the criteria gets satisfied, print he is eligible else print he is not eligible.

### AIM:

To create a C# program that checks whether Aravind is eligible to apply for a competitive exam based on age and 10th standard percentage criteria.

### PROGRAM AND OUTPUT:



The screenshot shows the 'Online C# Compiler' interface. On the left, the source code is displayed with line numbers 1 through 28. The code defines a 'Program' class with a 'Main' method. It sets eligibility criteria (minimum age 18, maximum age 30, minimum pass percentage 65.0). It prompts the user for age and percentage, with validation loops for invalid inputs. Finally, it checks if the user is eligible based on the criteria.

```

1 using System;
2
3 class Program
4 {
5     static void Main()
6     {
7         // Define eligibility criteria
8         const int minimumAge = 18;
9         const int maximumAge = 30;
10        const double minimumPassPercentage = 65.0;
11
12        // Input: Age and Percentage
13        Console.WriteLine("Enter your age: ");
14        int age;
15        while (!int.TryParse(Console.ReadLine(), out age) || age < 0)
16        {
17            Console.WriteLine("Please enter a valid age: ");
18        }
19
20        Console.WriteLine("Enter your percentage in 10th standard: ");
21        double percentage;
22        while (!double.TryParse(Console.ReadLine(), out percentage) ||
23            percentage < 0 || percentage > 100)
24        {
25            Console.WriteLine("Please enter a valid percentage (0-100): ");
26        }
27
28        // Eligibility Check
29        if (age > minimumAge && age <= maximumAge && percentage >=

```

On the right, the 'Terminal' window shows the program's execution. It prompts for age (23) and percentage (90), and then outputs 'You are eligible to apply.'

```

Enter your age: 23
Enter your percentage in 10th standard: 90
You are eligible to apply.

```

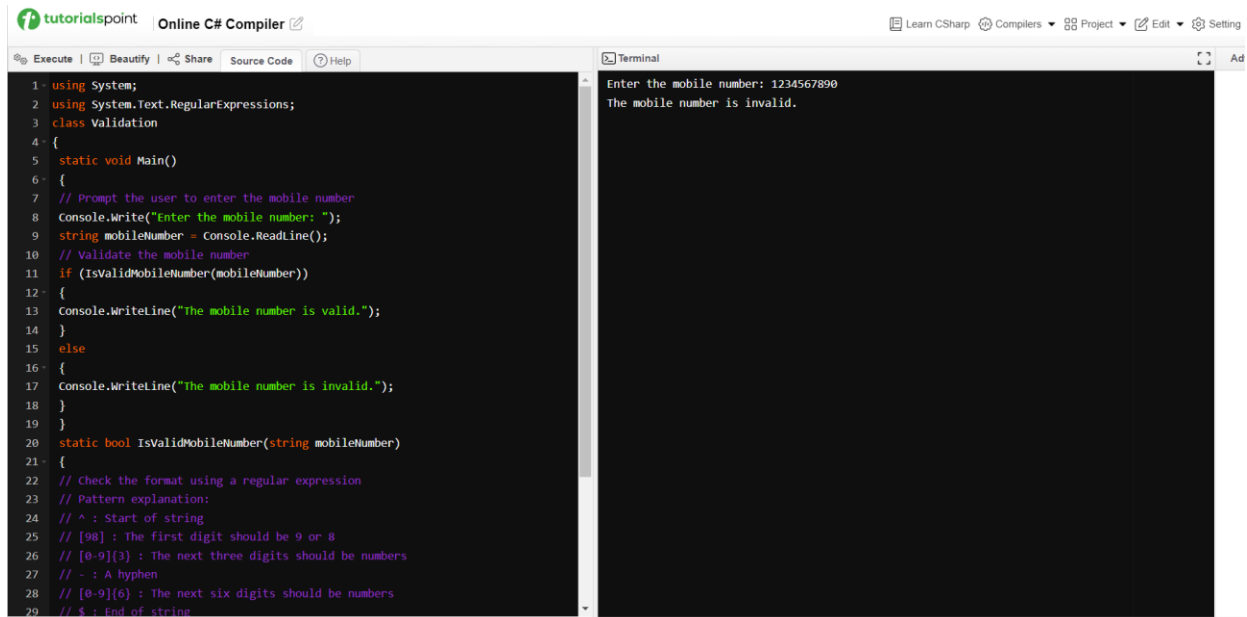
### 3. Design the C# console application named validation to get mobile number as input from the user. Validate the mobile number with the following cases:

- The first four number must be followed by then followed by next six numbers(eg:9894-256874) • Should contains only numbers
- Should be of length 10. • The first number should start only with 9 Or 8.

### AIM:

To create a C# console application that validates a mobile number based on specific criteria, including format, numeric content, length, and starting digit.

## PROGRAM AND OUTPUT:



```
1 using System;
2 using System.Text.RegularExpressions;
3 class Validation
4 {
5     static void Main()
6     {
7         // Prompt the user to enter the mobile number
8         Console.Write("Enter the mobile number: ");
9         string mobileNumber = Console.ReadLine();
10        // Validate the mobile number
11        if (IsValidMobileNumber(mobileNumber))
12        {
13            Console.WriteLine("The mobile number is valid.");
14        }
15        else
16        {
17            Console.WriteLine("The mobile number is invalid.");
18        }
19        }
20        static bool IsValidMobileNumber(string mobileNumber)
21        {
22            // Check the format using a regular expression
23            // Pattern explanation:
24            // ^ : Start of string
25            // [98] : The first digit should be 9 or 8
26            // [0-9]{3} : The next three digits should be numbers
27            // - : A hyphen
28            // [0-9]{6} : The next six digits should be numbers
29            // $ : End of string
```

4. Write the missing code snippets and the statements in the C# program given below.

Class person {

```
_____ name;
_____ age;
_____ weight;
```

Void printperson() {

```
    // write the code to print name, age and weight of a person
}
```

Class persondata {

```
Static void Main(string[] args) {
    person = _____;
```

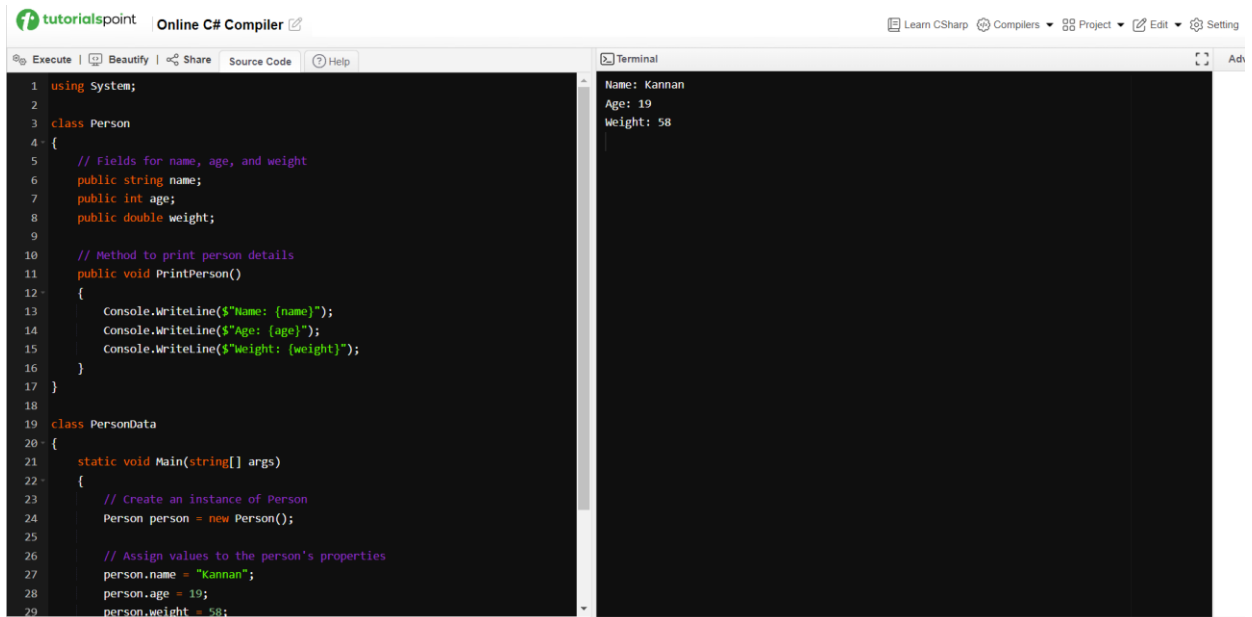
```
_____ .name = "Kannan";
_____ .age = 19;
_____ .weight = 58;
```

```
    // write the statement to access printperson() function
}
}
```

## AIM:

To create a C# program that defines a Person class with attributes name, age, and weight, and a method to print these values.

## PROGRAM AND OUTPUT:



The screenshot displays the 'Online C# Compiler' interface. The left pane shows the source code for a C# program. The right pane shows the output of the program, which is printed to the console.

```
1 using System;
2
3 class Person
4 {
5     // Fields for name, age, and weight
6     public string name;
7     public int age;
8     public double weight;
9
10    // Method to print person details
11    public void PrintPerson()
12    {
13        Console.WriteLine($"Name: {name}");
14        Console.WriteLine($"Age: {age}");
15        Console.WriteLine($"Weight: {weight}");
16    }
17 }
18
19 class PersonData
20 {
21     static void Main(string[] args)
22     {
23         // Create an instance of Person
24         Person person = new Person();
25
26         // Assign values to the person's properties
27         person.name = "Kannan";
28         person.age = 19;
29         person.weight = 58;
```

Output:

```
Name: Kannan
Age: 19
Weight: 58
```

## 5.A hospital wants to create a console application to maintain its inpatient details.

The information to store includes:

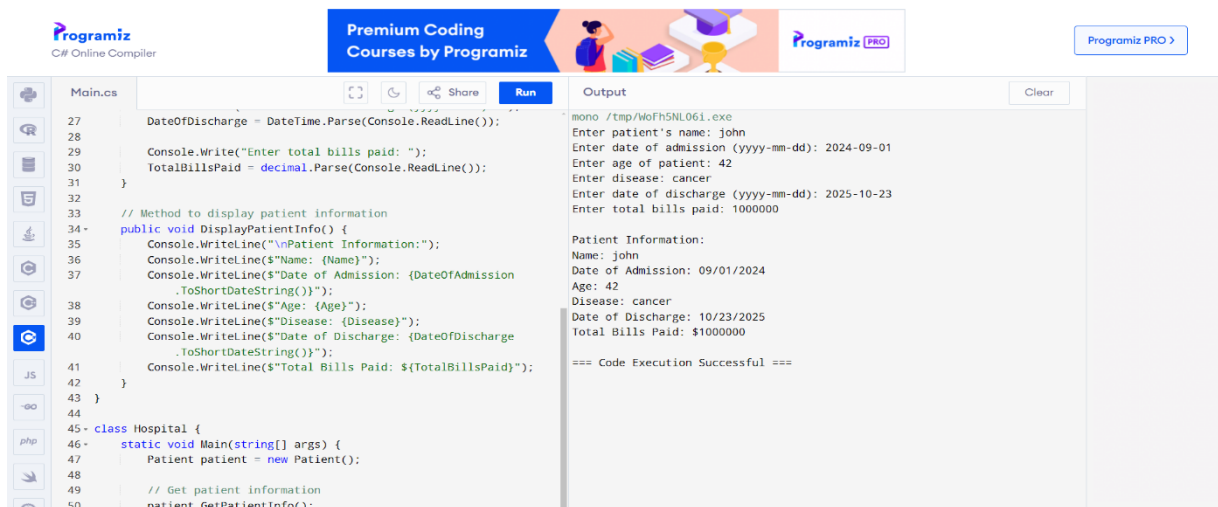
- Name of the patient
- Date of admission
- Age of patient
- Disease
- Date of discharge
- Total bills paid

Design the C# program with the class name patient with necessary data members to store the above information. The class should have two member functions, one to get the patients information and other to display the information. Create a main class called hospital to create necessary instances, methods calling statements and display all the details about the patient.

### AIM:

To create a C# console application that maintains and displays inpatient details including the patient's name, admission and discharge dates, age, disease, and total bills paid.

### PROGRAM AND OUTPUT:



The screenshot displays the Programiz C# Online Compiler interface. The code editor on the left shows a C# program with the following structure:

```
27 DateOfDischarge = DateTime.Parse(Console.ReadLine());
28
29 Console.WriteLine("Enter total bills paid: ");
30 TotalBillsPaid = decimal.Parse(Console.ReadLine());
31 }
32
33 // Method to display patient information
34 public void DisplayPatientInfo() {
35     Console.WriteLine("\nPatient Information:");
36     Console.WriteLine($"Name: {Name}");
37     Console.WriteLine($"Date of Admission: {DateOfAdmission}
38         .ToShortDateString()");
39     Console.WriteLine($"Age: {Age}");
40     Console.WriteLine($"Disease: {Disease}");
41     Console.WriteLine($"Date of Discharge: {DateOfDischarge}
42         .ToShortDateString()");
43     Console.WriteLine($"Total Bills Paid: {TotalBillsPaid}");
44 }
45
46 class Hospital {
47     static void Main(string[] args) {
48         Patient patient = new Patient();
49
50         // Get patient information
51         patient.GetPatientInfo();
```

The output window on the right shows the execution results:

```
mono /tmp/WoFh5NL061.exe
Enter patient's name: john
Enter date of admission (yyyy-mm-dd): 2024-09-01
Enter age of patient: 42
Enter disease: cancer
Enter date of discharge (yyyy-mm-dd): 2025-10-23
Enter total bills paid: 1000000

Patient Information:
Name: john
Date of Admission: 09/01/2024
Age: 42
Disease: cancer
Date of Discharge: 10/23/2025
Total Bills Paid: $1000000

=== Code Execution Successful ===
```

The screenshot displays the Programiz C# Online Compiler interface. The top header includes the Programiz logo, a banner for 'Premium Coding Courses by Programiz', and a 'Programiz PRO' button. The main workspace is divided into two panels: a code editor on the left and an output console on the right.

**Code Editor (Main.cs):**

```
1 using System;
2
3 class Patient {
4     // Properties for the Patient class
5     public string Name { get; set; }
6     public DateTime DateOfAdmission { get; set; }
7     public int Age { get; set; }
8     public string Disease { get; set; }
9     public DateTime DateOfDischarge { get; set; }
10    public decimal TotalBillsPaid { get; set; }
11
12    // Method to collect patient information
13    public void GetPatientInfo() {
14        Console.WriteLine("Enter patient's name: ");
15        Name = Console.ReadLine();
16
17        Console.WriteLine("Enter date of admission (yyyy-mm-dd): ");
18        DateOfAdmission = DateTime.Parse(Console.ReadLine());
19
20        Console.WriteLine("Enter age of patient: ");
21        Age = int.Parse(Console.ReadLine());
22
23        Console.WriteLine("Enter disease: ");
24        Disease = Console.ReadLine();
25
26        Console.WriteLine("Enter date of discharge (yyyy-mm-dd): ");
```

**Output Console:**

```
mono /tmp/woFh5NL06i.exe
Enter patient's name: john
Enter date of admission (yyyy-mm-dd): 2024-09-01
Enter age of patient: 42
Enter disease: cancer
Enter date of discharge (yyyy-mm-dd): 2025-10-23
Enter total bills paid: 1000000

Patient Information:
Name: john
Date of Admission: 09/01/2024
Age: 42
Disease: cancer
Date of Discharge: 10/23/2025
Total Bills Paid: $1000000

=== Code Execution Successful ===
```

**6. Implement the C# code to get two vector number as input, add them and print the sum as another vector. Make use of operator overloading to perform addition of vector numbers.**

#### AIM:

To create a C# program that uses operator overloading to add two vectors and print the resulting vector.

#### PROGRAM AND OUTPUT:

The screenshot shows the Programiz C# Online Compiler interface. The code editor on the left contains the following C# code:

```
1 using System;
2
3 class Vector
4 {
5     public int X { get; set; }
6     public int Y { get; set; }
7
8     public Vector(int x, int y)
9     {
10         X = x;
11         Y = y;
12     }
13
14     public static Vector operator +(Vector v1, Vector v2)
15     {
16         return new Vector(v1.X + v2.X, v1.Y + v2.Y);
17     }
18
19     public void Display()
20     {
21         Console.WriteLine($"{X}, {Y}");
22     }
23 }
24
25 class Program
26 {
```

The output window on the right shows the execution results:

```
mono /tmp/MA7F279Vg5.exe
Enter the x component of the first vector: 2
Enter the y component of the first vector: 3
Enter the x component of the second vector: 4
Enter the y component of the second vector: 5
Sum of the vectors: (6, 8)

=== Code Execution Successful ===
```

**7. Create the class student with necessary members to maintain the basic details of a student such as name, age, address and mobile number. Add method getDate() to read the basic details and printData() to print the details of the student. Inherit the student class into the sub class called studentmark with necessary members to maintain student mark details. Override the getDate() and printData() in student mark class to read mark details and print the marks, respectively. Also, define a method to find the grade of the student based on his/her marks. Design the student main class to access the member of both the classes in C#.**

#### **AIM:**

To create a C# program that manages and displays student details including basic information and marks, with functionality to compute and display the student's grade. The program uses class inheritance and method overriding.

#### **PROGRAM AND OUTPUT:**

The screenshot displays the Visual Studio Code editor interface. On the left, the file explorer shows a project named "Main.cs". The central editor pane contains the following C# code:

```
using System;

class Student
{
    public string Name { get; set; }
    public int Age { get; set; }
    public string Address { get; set; }
    public string MobileNumber { get; set; }

    public virtual void GetData()
    {
        Console.WriteLine("Enter student's name: ");
        Name = Console.ReadLine();

        Console.WriteLine("Enter student's age: ");
        Age = int.Parse(Console.ReadLine());

        Console.WriteLine("Enter student's address: ");
        Address = Console.ReadLine();

        Console.WriteLine("Enter student's mobile number: ");
        MobileNumber = Console.ReadLine();
    }

    public virtual void PrintData()
    {
```

To the right of the code editor, there are icons for running the application (a green play button), debugging (a blue bug icon), and sharing (a share icon). Below these icons is a red "Run" button.

The output window on the far right shows the results of executing the program:

```
mono /tmp/e6gtyLX1z0.exe
Enter student's name: aswanth
Enter student's age: 23
Enter student's address: 7th cross street
Enter student's mobile number: 9087579257
Enter student's marks: 89

Student Details:
Name: aswanth
Age: 23
Address: 7th cross street
Mobile Number: 9087579257
Marks: 89
Grade: B

=== Code Execution Successful ===
```

Main.cs

Run

Clear

```
63
64 public void Display()
65 {
66     Console.WriteLine("\nEmployee Details:");
67     Console.WriteLine($"Employee Number: {empNo}");
68     Console.WriteLine($"Employee Name: {empName}");
69     Console.WriteLine($"Basic Salary: Rs. {basicSalary}");
70     Console.WriteLine($"HRA: Rs. {hra}");
71     Console.WriteLine($"DA: Rs. {da}");
72     Console.WriteLine($"Loan: Rs. {loan}");
73     Console.WriteLine($"PF: Rs. {pf}");
74     Console.WriteLine($"Net Salary: Rs. {netSalary}");
75 }
76 }
77
78 class Program
79 {
80     static void Main(string[] args)
81     {
82         Employee employee = new Employee(0, 0, 0, 0, 0);
83         employee.Input();
84         employee.CalculateSalary();
85         employee.Display();
86     }
87 }
88
```

Output

mono /tmp/760fp0uC8s.exe
Enter employee number: 8
Enter employee name: Kavin
Enter job category (1 for Table-I, 2 for Table-II): 2

Employee Details:
Employee Number: 8
Employee Name: Kavin
Basic Salary: Rs. 15000
HRA: Rs. 3000.00
DA: Rs. 4500.00
Loan: Rs. 600
PF: Rs. 1000
Net Salary: Rs. 20900.00

=== Code Execution Successful ===



**8. Design sample C# program with class name employee to compute netsalary of the employee using the basic salary, if for the job\_catg is 1 use table-I else use table-II. Use constructor to initialize basic salary,hra,da,pf and loan. The employee class should contain input() method to get input for job\_catg, empno, empname, calculateSalary() method to compute salary and display() method to print the details.**

Table-I	Table-II
<b>BASIC=Rs. 8,000</b> <b>HRA=10% of basic</b> <b>DA=20% of basic</b> <b>LOAN=Rs. 300</b> <b>PF=Rs. 500</b>	<b>BASIC=Rs. 15,000</b> <b>HRA=20% of basic</b> <b>DA=30% of basic</b> <b>LOAN=Rs. 600</b> <b>PF=1000</b>

**AIM:**

To create a C# program that calculates and displays the net salary of an employee based on their job category using predefined salary tables. The program uses constructors for initialization and methods for input, salary calculation, and displaying details.

**PROGRAM AND OUTPUT:**



Main.cs

```
63
64 public void Display()
65 {
66     Console.WriteLine("\nEmployee Details:");
67     Console.WriteLine($"Employee Number: {empNo}");
68     Console.WriteLine($"Employee Name: {empName}");
69     Console.WriteLine($"Basic Salary: Rs. {basicSalary}");
70     Console.WriteLine($"HRA: Rs. {hra}");
71     Console.WriteLine($"DA: Rs. {da}");
72     Console.WriteLine($"Loan: Rs. {loan}");
73     Console.WriteLine($"PF: Rs. {pf}");
74     Console.WriteLine($"Net Salary: Rs. {netSalary}");
75 }
76 }
77
78 class Program
79 {
80     static void Main(string[] args)
81     {
82         Employee employee = new Employee(0, 0, 0, 0, 0);
83         employee.Input();
84         employee.CalculateSalary();
85         employee.Display();
86     }
87 }
88
```

Share Run

Output

Clear

```
mono /tmp/760fp0u0Cs.exe
Enter employee number: 8
Enter employee name: Kavin
Enter job category (1 for Table-I, 2 for Table-II): 2

Employee Details:
Employee Number: 8
Employee Name: Kavin
Basic Salary: Rs. 15000
HRA: Rs. 3000.00
DA: Rs. 4500.00
Loan: Rs. 600
PF: Rs. 1000
Net Salary: Rs. 20900.00

=== Code Execution Successful ===
```



Main.cs

Run

Clear

```
63
64 public void Display()
65 {
66     Console.WriteLine("\nEmployee Details:");
67     Console.WriteLine($"Employee Number: {empNo}");
68     Console.WriteLine($"Employee Name: {empName}");
69     Console.WriteLine($"Basic Salary: Rs. {basicSalary}");
70     Console.WriteLine($"HRA: Rs. {hra}");
71     Console.WriteLine($"DA: Rs. {da}");
72     Console.WriteLine($"Loan: Rs. {loan}");
73     Console.WriteLine($"PF: Rs. {pf}");
74     Console.WriteLine($"Net Salary: Rs. {netSalary}");
75 }
76 }
77
78 class Program
79 {
80     static void Main(string[] args)
81     {
82         Employee employee = new Employee(0, 0, 0, 0, 0);
83         employee.Input();
84         employee.CalculateSalary();
85         employee.Display();
86     }
87 }
88
```

Output

mono /tmp/760fp0uC8s.exe
Enter employee number: 8
Enter employee name: Kavin
Enter job category (1 for Table-I, 2 for Table-II): 2

Employee Details:
Employee Number: 8
Employee Name: Kavin
Basic Salary: Rs. 15000
HRA: Rs. 3000.00
DA: Rs. 4500.00
Loan: Rs. 600
PF: Rs. 1000
Net Salary: Rs. 20900.00

=== Code Execution Successful ===

**BY:**

MAGHAASWANTH M S (73772226130) III – B.TECH AI&DS