NAME : YOGANA R

REG NO : 73772214224

DEPT : BE-CSE-B(III)

COURSE CODE : 60 IT L04

COURSE NAME : C# AND .NET FRAMEWORKS

ASSIGNMENT 1

1)

Aim

Develop a C# program to initialize a two-dimensional array and print all the elements on the same line, separated by space.

Algorithm

- ✓ Start.
- ✓ Declare and initialize a 2D array with some elements.
- ✓ Loop through the 2D array using nested loops.
- ✓ In the inner loop, print each element on the same line with a space.
- ✓ End.

Program

```
using System;

class Program
{
   static void Main()
   {

      int[,] array = {
         {1, 2, 3},
         {4, 5, 6},
         {7, 8, 9}
      };

      for (int i = 0; i < array.GetLength(0); i++)
      {
         for (int j = 0; j < array.GetLength(1); j++)
```

```
        {
            Console.Write(array[i, j] + " ");
        }
    }
    Console.WriteLine();
  }
}
```

Input

```
 {1, 2, 3},
 {4, 5, 6},
 {7, 8, 9}
```

Output

1 2 3

4 5 6

7 8 9

Result

The program iterates through each element of the 2D array using nested for loops and prints each element followed by a space. After printing each row, it moves to the next line. This results in the 2D array being displayed in a grid-like format on the console.

2)

Aim

To determine eligibility based on age and percentage criteria and print the result.

Algorithm:

- ✓ Start.
- ✓ Get the age and percentage of marks from the user.
- ✓ Check if the age is between 18 and 30.
- ✓ Check if the marks are 65% or more.
- ✓ If both conditions are satisfied, print "Eligible"; otherwise, print "Not eligible."
- ✓ End.

Program

```
using System;

class Program

{

   static void Main()

   {

      Console.Write("Enter Age: ");

      int age = int.Parse(Console.ReadLine());

      Console.Write("Enter 10th Grade Percentage: ");

      double percentage = double.Parse(Console.ReadLine());

      if (age > 18 && age < 30 && percentage >= 65)

      {

         Console.WriteLine("Eligible");

      }

      else

      {

         Console.WriteLine("Not eligible");

      }

   }

}
```

Input

- age: 10
- percentage: 50

Output

Not Eligible

Result

Thus the program executed successfully.

3)

Aim

Create a C# console application to validate a mobile number entered by the user.

Algorithm:

1. Start.
2. Get the mobile number from the user.
3. Check if the mobile number starts with 9 or 8.
4. Check if the format is valid (first four digits followed by "-" and then six digits).
5. If the mobile number meets the criteria, print "Valid Number"; otherwise, print "Invalid Number."
6. End.

Program

```
using System;

class Program

{

   static void Main()

   {

      Console.Write("Enter your mobile number: ");

      string mobileNumber = Console.ReadLine();

      if (IsValidMobile(mobileNumber))

      {

         Console.WriteLine("The mobile number is valid.");

      }

      else

      {

         Console.WriteLine("The mobile number is invalid.");

      }

      Console.ReadLine();

   }

   static bool IsValidMobile(string number)
```

```
    {
        if (number.Length == 10 && (number.StartsWith("9") || number.StartsWith("8")))
        {
            foreach (char c in number)
            {
                if (!char.IsDigit(c))
                {
                    return false;
                }
            }
            return true;
        }
        return false;
    }
}
```

Input

Enter your mobile number: 9087362530

Output

The mobile number is valid.

Result

Thus the program executed successfully.

4)

Aim

Write the missing code snippets in a C# program related to a `Person` class.

Program

```
using System;

class Person
{
   public string name;
   public int age;
   public double weight;

   public void printPerson()
   {
      Console.WriteLine("Name: " + name);
      Console.WriteLine("Age: " + age);
      Console.WriteLine("Weight: " + weight);
   }
}

class PersonData
{
   static void Main(string[] args)
   {

      Person person = new Person();
      person.name = "Hari";
      person.age = 19;
      person.weight = 58;

      person.printPerson();
   }
}
```

Input

- Name: Hari
- Age: 19
- Weight: 58

Output

Hari, Age: 19, Weight: 58

Result

Thus the program executed successfully.

5)

Aim

To manage and display patient information including details like name, admission and discharge dates, age, disease, and total bills paid.

Algorithm:

- ✓ Start.
- ✓ Define a Patient class with data members for:
  - name
  - dateOfAdmission
  - age
  - disease
  - dateOfDischarge
  - totalBillsPaid
- ✓ Create two methods inside the Patient class:
  - One method to input details (GetPatientInfo()).
  - Another method to display details (DisplayPatientInfo()).
- ✓ Define a Hospital class with the Main() method to:
  - Create an object of Patient class.
  - Call GetPatientInfo() to input details.
  - Call DisplayPatientInfo() to display details.
- ✓ End.

Program

```
using System;

class Patient
{
   public string name;
   public string dateOfAdmission;
   public int age;
   public string disease;
   public string dateOfDischarge;
   public double totalBillsPaid;

   public void GetPatientInfo()
   {
      Console.Write("Enter Patient Name: ");
      name = Console.ReadLine();

      Console.Write("Enter Date of Admission: ");
      dateOfAdmission = Console.ReadLine();

      Console.Write("Enter Age of Patient: ");
      age = int.Parse(Console.ReadLine());
```

```
        Console.Write("Enter Disease: ");
        disease = Console.ReadLine();

        Console.Write("Enter Date of Discharge: ");
        dateOfDischarge = Console.ReadLine();

        Console.Write("Enter Total Bills Paid: ");
        totalBillsPaid = double.Parse(Console.ReadLine());
    }

    public void DisplayPatientInfo()
    {
        Console.WriteLine("\n--- Patient Information ---");
        Console.WriteLine("Name: " + name);
        Console.WriteLine("Date of Admission: " + dateOfAdmission);
        Console.WriteLine("Age: " + age);
        Console.WriteLine("Disease: " + disease);
        Console.WriteLine("Date of Discharge: " + dateOfDischarge);
        Console.WriteLine("Total Bills Paid: Rs. " + totalBillsPaid);
    }
}

class Hospital
{
    static void Main(string[] args)
    {
        Patient patient = new Patient();
        patient.GetPatientInfo();
        patient.DisplayPatientInfo();
    }
}
```

Input

Enter Name: Alice
Enter Age: 45
Enter Disease: Pneumonia
Enter Date of Admission (yyyy-mm-dd): 2023-09-01
Enter Date of Discharge (yyyy-mm-dd): 2023-09-10
Enter Total Bills Paid: 15000

Output

Name: Alice
Age: 45
Disease: Pneumonia
Date of Admission: 9/1/2023 12:00:00 AM
Date of Discharge: 9/10/2023 12:00:00 AM
Total Bill: 15000

Result

Thus the program executed successfully.

6)

Aim

Implement a C# program to input two vector numbers, add them, and print the sum. Use operator overloading to perform the addition of vector numbers.

Algorithm:

- ✓ Start.
- ✓ Define a Vector class with members x, y, and z to represent 3D vectors.
- ✓ Create a constructor to initialize vector values.
- ✓ Overload the + operator to add two vectors.
- ✓ Define a method to display the result.
- ✓ In the Main method, create two vector objects, add them using the overloaded operator, and display the result.
- ✓ End.

Program

```
using System;

class Vector
{
    public int x, y, z;

    public Vector(int x, int y, int z)
    {
        this.x = x;

        this.y = y;

        this.z = z;
    }

    public static Vector operator +(Vector v1, Vector v2)
    {
        return new Vector(v1.x + v2.x, v1.y + v2.y, v1.z + v2.z);
    }

    public void Display()
    {
        Console.WriteLine($"Vector: {x}i + {y}j + {z}k");
    }
}
```

```
class Program
{
   static void Main(string[] args)
   {
      Vector v1 = new Vector(1, 2, 3);
      Vector v2 = new Vector(4, 5, 6);
     Vector v3 = v1 + v2;
      v3.Display();
   }
}
```

Input:

Two vectors: v1(1, 2, 3) and v2(4, 5, 6).

Output:

Sum of vectors: 5i + 7j + 9k.

Resullt

Thus the program executed successfully.

7)

Aim

Create a C# program to manage student details, with inheritance to maintain marks. Include methods to get and display details and marks, and override necessary methods.

Algorithm:

- ✓ Start.
- ✓ Define a `Student` class to store student details such as name, age, address, and mobile number.
- ✓ Create a method to get and print student details.
- ✓ Define a `Mark` class (subclass) that inherits `Student` and adds attributes for marks.
- ✓ Override methods to get and display marks.
- ✓ In the `Main` method, create an instance of `Mark`, input and display the student's details and marks.
- ✓ End.

Program

```
using System;

class Student
{
   public string name;
   public int age;
   public string address;
   public string mobileNumber;

   public virtual void GetDetails()
   {
      Console.Write("Enter Name: ");
      name = Console.ReadLine();

      Console.Write("Enter Age: ");
      age = int.Parse(Console.ReadLine());

      Console.Write("Enter Address: ");
      address = Console.ReadLine();

      Console.Write("Enter Mobile Number: ");
      mobileNumber = Console.ReadLine();
   }
```

```
    public virtual void PrintDetails()
    {
        Console.WriteLine("\n--- Student Details ---");
        Console.WriteLine("Name: " + name);
        Console.WriteLine("Age: " + age);
        Console.WriteLine("Address: " + address);
        Console.WriteLine("Mobile Number: " + mobileNumber);
    }
}

class Mark : Student
{
    public int mark1, mark2, mark3;

    public override void GetDetails()
    {
        base.GetDetails();

        Console.Write("Enter Mark 1: ");
        mark1 = int.Parse(Console.ReadLine());

        Console.Write("Enter Mark 2: ");
        mark2 = int.Parse(Console.ReadLine());

        Console.Write("Enter Mark 3: ");
        mark3 = int.Parse(Console.ReadLine());
    }

    public override void PrintDetails()
    {
        base.PrintDetails();

        Console.WriteLine("Mark 1: " + mark1);
        Console.WriteLine("Mark 2: " + mark2);
        Console.WriteLine("Mark 3: " + mark3);
    }
}

class Program
{
    static void Main(string[] args)
    {
        Mark student = new Mark();
        student.GetDetails();
        student.PrintDetails();
    }
}
```

Input

Enter Name: Yoga
Enter Age: 18
Enter Address: 123 Main St
Enter Mobile: 9876543210
Enter Mark 1: 96
Enter Mark 2: 99
Enter Mark 3: 98

Output

Name: Yoga,
Age: 18,
Address: 123 Main St,
Mobile: 9876543210
Marks: 96, 99, 98

Result

Thus the program executed successfully.

8)

Aim

Design a C# program with a class Employee to compute the salary of an employee using basic salary for two job categories. Use a constructor to initialize basic salary, HRA, DA, loan, and PF. Include methods to calculate and display the salary.

Algorithm:

- ✓ Start.
- ✓ Define an Employee class with data members for basic, HRA, DA, loan, and PF.
- ✓ Use a constructor to initialize the values based on job category (1 or 2).
- ✓ Create a method calculateSalary() to compute salary using the formula: salary=basic+HRA+DA−(loan+PF)\text{salary} = \text{basic} + \text{HRA} + \text{DA} - (\text{loan} + \text{PF})salary=basic+HRA+DA−(loan+PF)
- ✓ Create a method display() to print the salary.
- ✓ In the Main method, create an employee object, calculate, and display the salary.
- ✓ End.

Program

```
using System;

class Employee
{
   private double basic, hra, da, loan, pf;

   public Employee(int jobCategory)
   {
      if (jobCategory == 1)
      {
         basic = 8000;
         hra = 0.11 * basic;
         da = 0.20 * basic;
         loan = 300;
         pf = 500;
      }
      else if (jobCategory == 2)
      {
         basic = 15000;
         hra = 0.20 * basic;
         da = 0.30 * basic;
         loan = 600;
         pf = 1000;
      }
```

```
        else
        {
            Console.WriteLine("Invalid job category.");
        }
    }


    public double calculateSalary()
    {
        return basic + hra + da - (loan + pf);
    }


    public void display()
    {
        Console.WriteLine("\n--- Salary Details ---");
        Console.WriteLine($"Basic: {basic}");
        Console.WriteLine($"HRA: {hra}");
        Console.WriteLine($"DA: {da}");
        Console.WriteLine($"Loan Deduction: {loan}");
        Console.WriteLine($"PF Deduction: {pf}");
        Console.WriteLine($"Net Salary: {calculateSalary()}");
    }
}

class Program
{
    static void Main(string[] args)
    {

        Console.Write("Enter Job Category (1 or 2): ");
        int jobCategory = int.Parse(Console.ReadLine());

        Employee employee = new Employee(jobCategory);

        employee.display();
    }
}
```

Input

Enter Job Category (1 or 2): 1

Output

Basic: 8000

HRA: 880

DA: 1600

Loan Deduction: 300

PF Deduction: 500

Net Salary: 9680

Result

Thus the program executed successfully.