| NAME | RAJ SANJAY JADHAV |
| --- | --- |
| CLASS | BE-3-CSE |
| BATCH | B |
| ROLL NO | 512024 |
| PRN | 2019033800129402 |

**PRACTICAL-4**

**GIT REPOSITORY :** [**PRACTICAL-4**](https://github.com/Rajjadhav1710/.NET_Practical/tree/main/Practical-4)

**PROPERTIES**

**CODE:**

**using System;**

**class TimePeriod**

**{**

**private double \_seconds;**

**public double Hours**

**{**

**get { return \_seconds / 3600; }**

**set {**

**if (value < 0 || value > 24)**

**throw new ArgumentOutOfRangeException(**

**$"{nameof(value)} must be between 0 and 24.");**

**\_seconds = value \* 3600;**

**}**

**}**

**}**

**class Program**

**{**

**static void Main()**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**TimePeriod t = new TimePeriod();**

**// The property assignment causes the 'set' accessor to be called.**

**t.Hours = 24;**

**// Retrieving the property causes the 'get' accessor to be called.**

**Console.WriteLine($"Time in hours: {t.Hours}");**

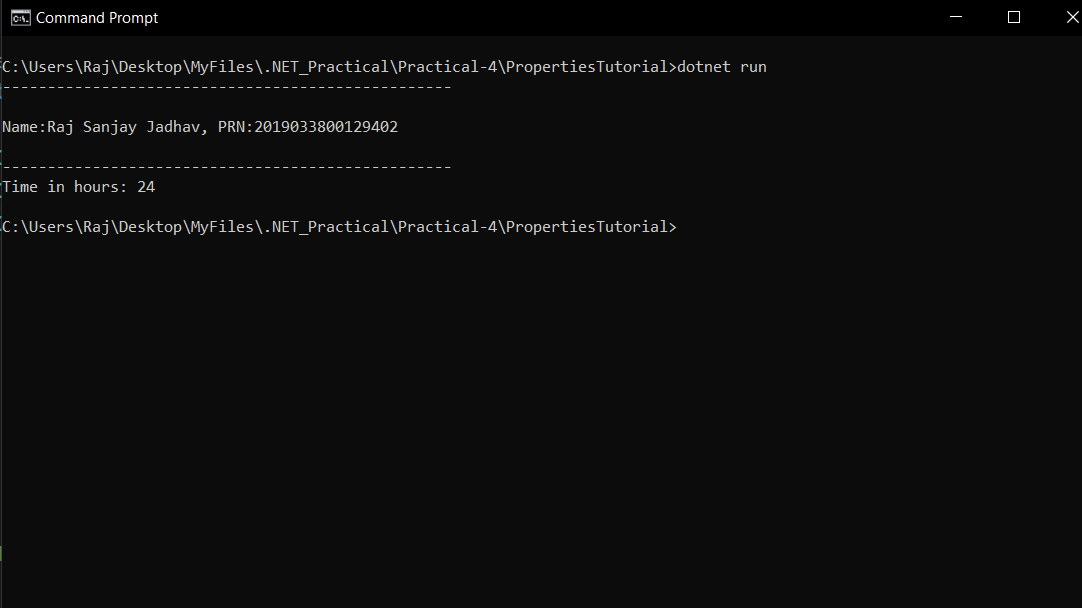
**}**

**}**

**// The example displays the following output:**

**// Time in hours: 24**

**OUTPUT:**

****

**CODE:**

**using System;**

**public class Person**

**{**

**private string \_firstName;**

**private string \_lastName;**

**public Person(string first, string last)**

**{**

**\_firstName = first;**

**\_lastName = last;**

**}**

**public string Name => $"{\_firstName} {\_lastName}";**

**}**

**public class Example**

**{**

**public static void Main()**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

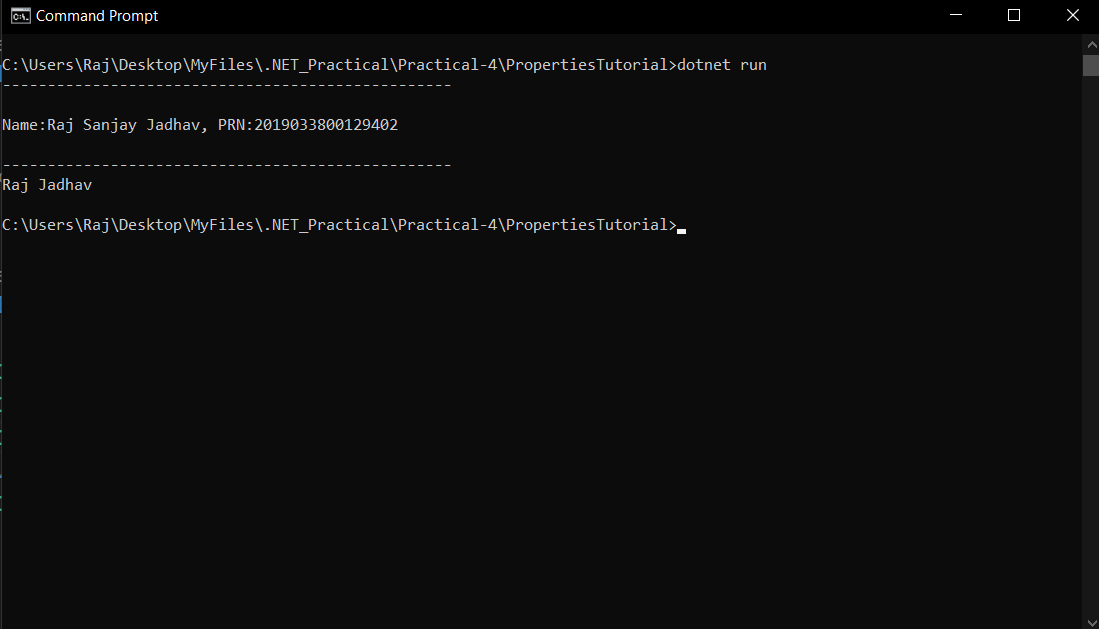
**var person = new Person("Raj", "Jadhav");**

**Console.WriteLine(person.Name);**

**}**

**}**

**OUTPUT:**

****

**CODE:**

**using System;**

**public class SaleItem**

**{**

**string \_name;**

**decimal \_cost;**

**public SaleItem(string name, decimal cost)**

**{**

**\_name = name;**

**\_cost = cost;**

**}**

**public string Name**

**{**

**get => \_name;**

**set => \_name = value;**

**}**

**public decimal Price**

**{**

**get => \_cost;**

**set => \_cost = value;**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

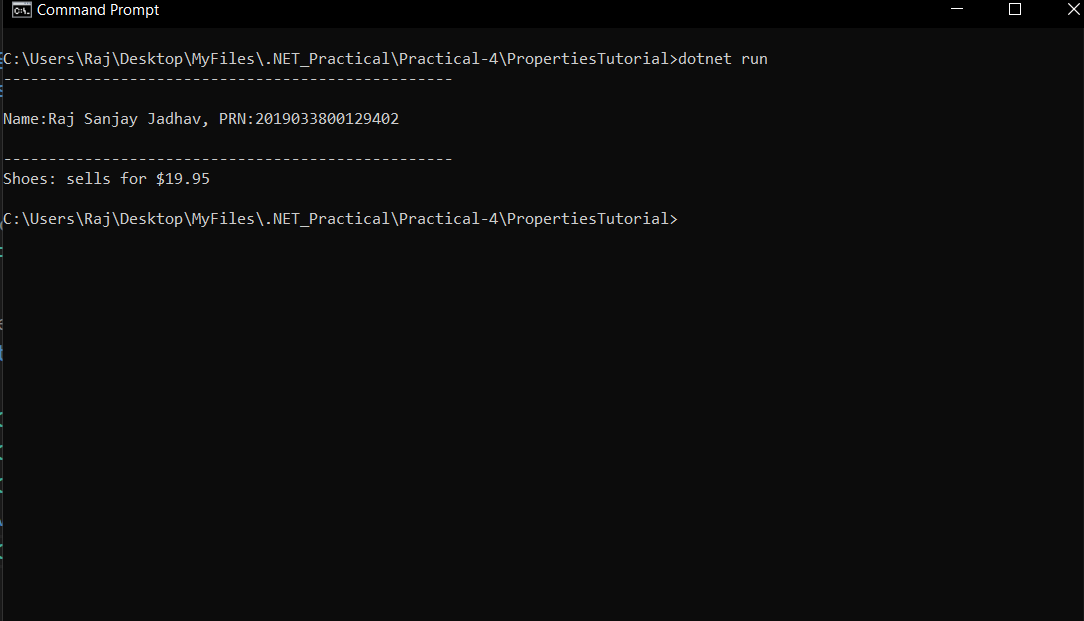
**var item = new SaleItem("Shoes", 19.95m);**

**Console.WriteLine($"{item.Name}: sells for ${item.Price}");**

**}**

**}**

**OUTPUT:**

****

**CODE:**

**using System;**

**public class SaleItem**

**{**

**public string Name**

**{ get; set; }**

**public decimal Price**

**{ get; set; }**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

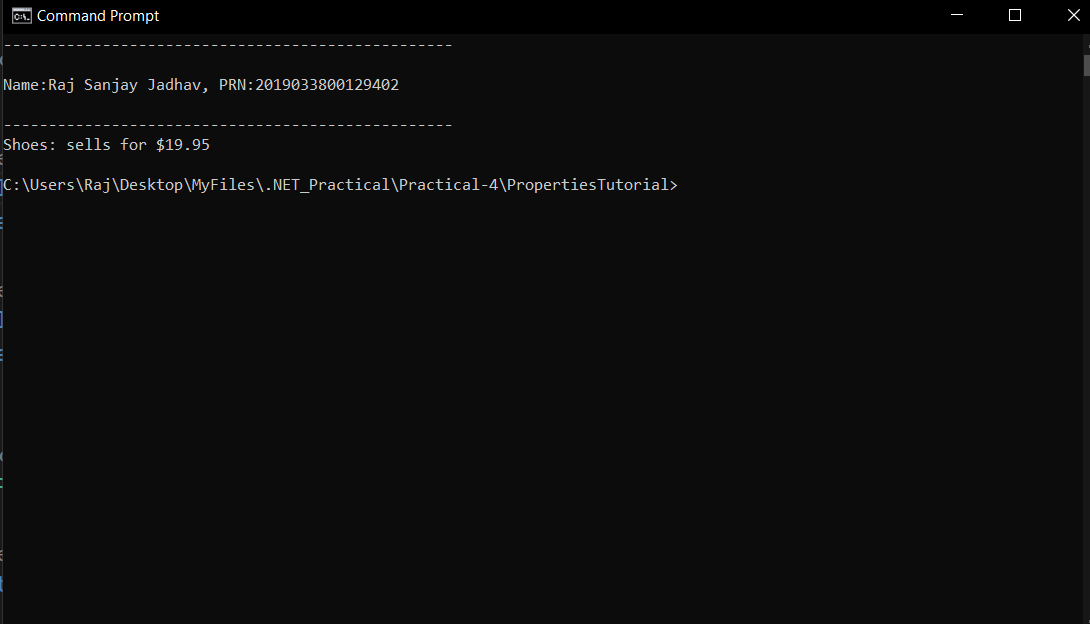
**var item = new SaleItem{ Name = "Shoes", Price = 19.95m };**

**Console.WriteLine($"{item.Name}: sells for ${item.Price}");**

**}**

**}**

**OUTPUT:**

****

**USING INDEXERS**

**CODE:**

**using System;**

**public class TempRecord**

**{**

**// Array of temperature values**

**float[] temps = new float[10]**

**{**

**56.2F, 56.7F, 56.5F, 56.9F, 58.8F,**

**61.3F, 65.9F, 62.1F, 59.2F, 57.5F**

**};**

**// To enable client code to validate input**

**// when accessing your indexer.**

**public int Length => temps.Length;**

**// Indexer declaration.**

**// If index is out of range, the temps array will throw the exception.**

**public float this[int index]**

**{**

**get => temps[index];**

**set => temps[index] = value;**

**}**

**}**

**class Program**

**{**

**static void Main()**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**var tempRecord = new TempRecord();**

**// Use the indexer's set accessor**

**tempRecord[3] = 58.3F;**

**tempRecord[5] = 60.1F;**

**// Use the indexer's get accessor**

**for (int i = 0; i < 10; i++)**

**{**

**Console.WriteLine($"Element #{i} = {tempRecord[i]}");**

**}**

**// Keep the console window open in debug mode.**

**Console.WriteLine("Press any key to exit.");**

**Console.ReadKey();**

**}**

**/\* Output:**

**Element #0 = 56.2**

**Element #1 = 56.7**

**Element #2 = 56.5**

**Element #3 = 58.3**

**Element #4 = 58.8**

**Element #5 = 60.1**

**Element #6 = 65.9**

**Element #7 = 62.1**

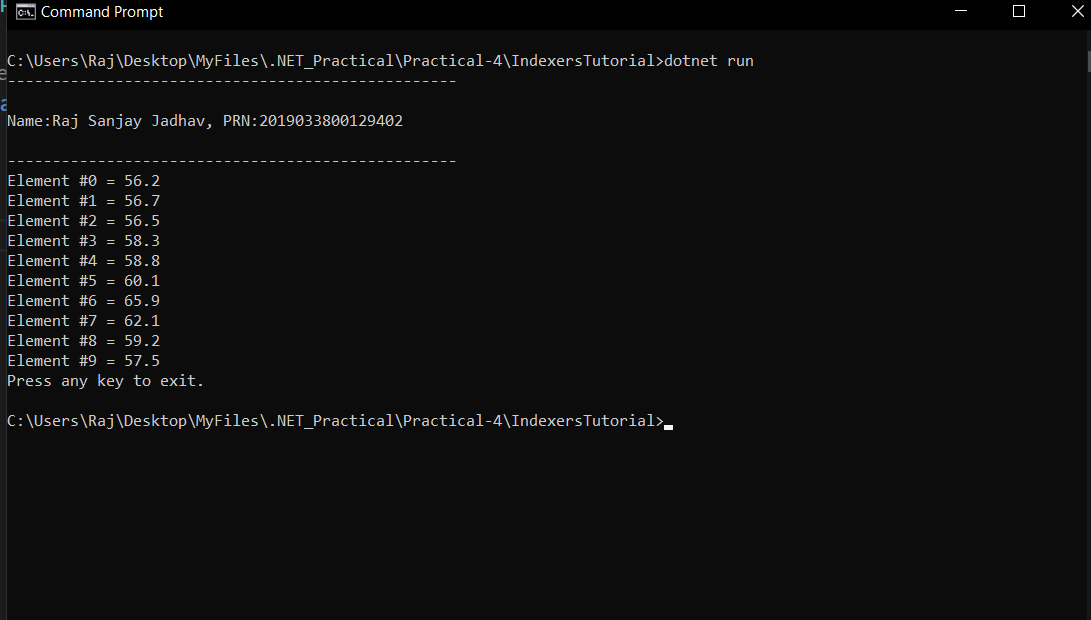
**Element #8 = 59.2**

**Element #9 = 57.5**

**\*/**

**}**

**OUTPUT:**

****

**CODE:**

**using System;**

**// Using a string as an indexer value**

**class DayCollection**

**{**

**string[] days = { "Sun", "Mon", "Tues", "Wed", "Thurs", "Fri", "Sat" };**

**// Indexer with only a get accessor with the expression-bodied definition:**

**public int this[string day] => FindDayIndex(day);**

**private int FindDayIndex(string day)**

**{**

**for (int j = 0; j < days.Length; j++)**

**{**

**if (days[j] == day)**

**{**

**return j;**

**}**

**}**

**throw new ArgumentOutOfRangeException(**

**nameof(day),**

**$"Day {day} is not supported.\nDay input must be in the form \"Sun\", \"Mon\", etc");**

**}**

**}**

**class Program**

**{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**var week = new DayCollection();**

**Console.WriteLine(week["Fri"]);**

**try**

**{**

**Console.WriteLine(week["Made-up day"]);**

**}**

**catch (ArgumentOutOfRangeException e)**

**{**

**Console.WriteLine($"Not supported input: {e.Message}");**

**}**

**}**

**// Output:**

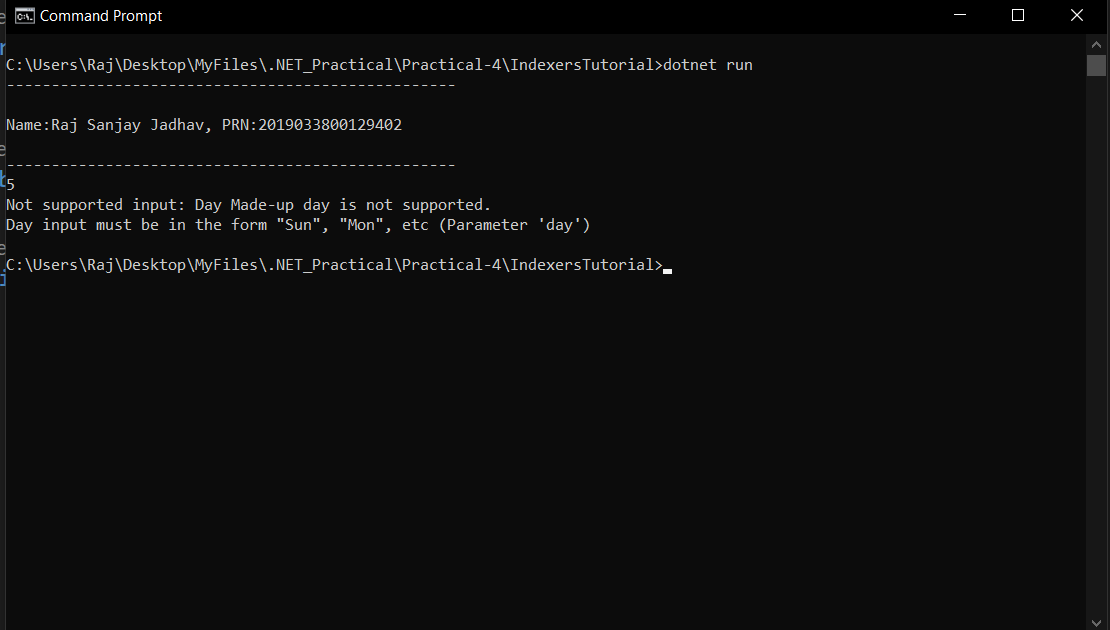
**// 5**

**// Not supported input: Day Made-up day is not supported.**

**// Day input must be in the form "Sun", "Mon", etc (Parameter 'day')**

**}**

**OUTPUT:**

****

**SIMPLE CLASS EXAMPLE**

**CODE:**

**using System;**

**using System.Reflection;**

**public class SimpleClassExample**

**{**

**public static void Main()**

**{**

**Type t = typeof(SimpleClassExample);**

**BindingFlags flags = BindingFlags.Instance | BindingFlags.Static | BindingFlags.Public |**

**BindingFlags.NonPublic | BindingFlags.FlattenHierarchy;**

**MemberInfo[] members = t.GetMembers(flags);**

**Console.WriteLine($"Type {t.Name} has {members.Length} members: ");**

**foreach (var member in members)**

**{**

**string access = "";**

**string stat = "";**

**var method = member as MethodBase;**

**if (method != null)**

**{**

**if (method.IsPublic)**

**access = " Public";**

**else if (method.IsPrivate)**

**access = " Private";**

**else if (method.IsFamily)**

**access = " Protected";**

**else if (method.IsAssembly)**

**access = " Internal";**

**else if (method.IsFamilyOrAssembly)**

**access = " Protected Internal ";**

**if (method.IsStatic)**

**stat = " Static";**

**}**

**var output = $"{member.Name} ({member.MemberType}): {access}{stat}, Declared by {member.DeclaringType}";**

**Console.WriteLine(output);**

**}**

**}**

**}**

**// The example displays the following output:**

**// Type SimpleClass has 9 members:**

**// ToString (Method): Public, Declared by System.Object**

**// Equals (Method): Public, Declared by System.Object**

**// Equals (Method): Public Static, Declared by System.Object**

**// ReferenceEquals (Method): Public Static, Declared by System.Object**

**// GetHashCode (Method): Public, Declared by System.Object**

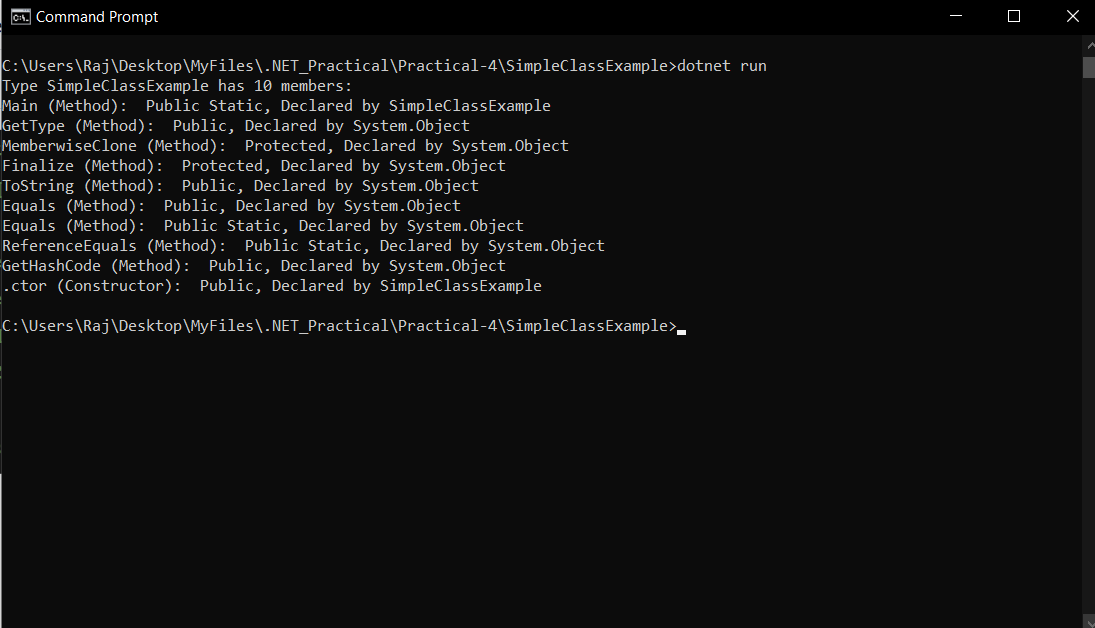
**// GetType (Method): Public, Declared by System.Object**

**// Finalize (Method): Internal, Declared by System.Object**

**// MemberwiseClone (Method): Internal, Declared by System.Object**

**// .ctor (Constructor): Public, Declared by SimpleClass**

**OUTPUT:**

****

**CLASSES**

**CODE:**

**using System;**

**class Employee{**

**private string \_firstName;**

**private string \_lastName;**

**private double \_monthlySalary;**

**public Employee(string firstName,string lastName,double monthlySalary){**

**this.\_firstName=firstName;**

**this.\_lastName=lastName;**

**this.\_monthlySalary=monthlySalary;**

**}**

**public string firstName**

**{**

**get => \_firstName;**

**set =>\_firstName=value;**

**}**

**public string lastName**

**{**

**get => \_lastName;**

**set =>\_lastName=value;**

**}**

**public double monthlySalary**

**{**

**get => \_monthlySalary;**

**// set => \_monthlySalary=value;**

**set{**

**if(value<0){**

**\_monthlySalary=0.0;**

**}else{**

**\_monthlySalary=value;**

**}**

**}**

**}**

**public double yearlySalary**

**{**

**get{**

**return monthlySalary\*12;**

**}**

**}**

**public virtual void giveRaise(){**

**monthlySalary+=0.1\*monthlySalary;**

**}**

**public override string ToString()**

**{**

**return "\nFirst Name:"+firstName+" , Last Name:"+lastName+" , Yearly Salary:"+yearlySalary+"\n";**

**}**

**}**

**class EmployeeTest{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**//1**

**Employee emp1=new Employee("john","doe",10000);**

**Employee emp2=new Employee("raj","jadhav",20000);**

**Console.WriteLine(emp1);**

**Console.WriteLine(emp2);**

**emp1.giveRaise();**

**emp2.giveRaise();**

**Console.WriteLine("\nAfter Raise In Salary Employee Details:\n");**

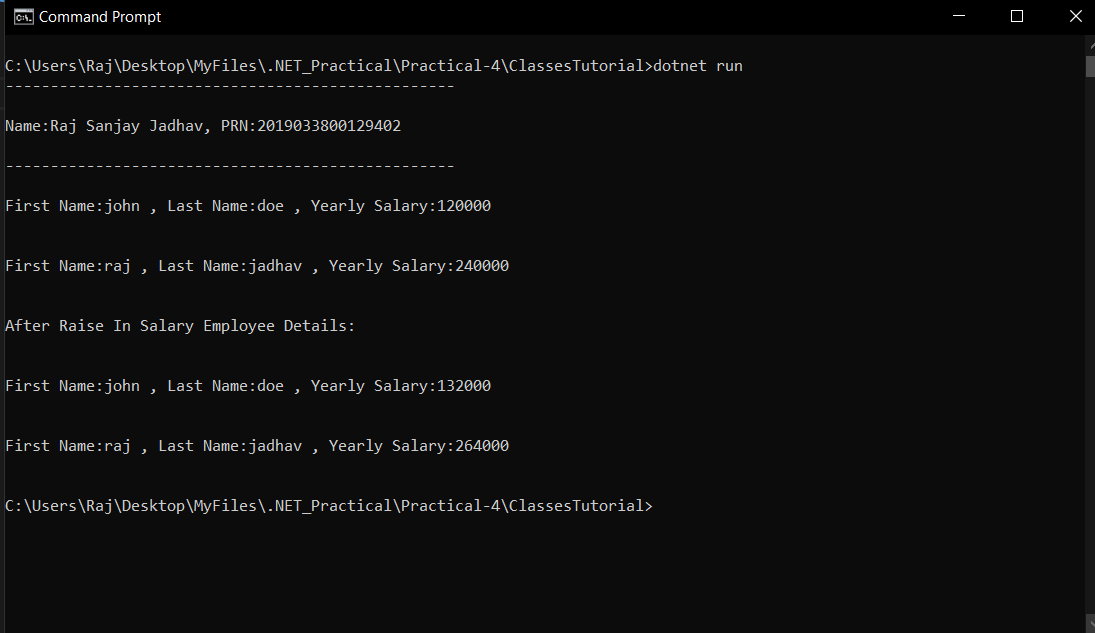
**Console.WriteLine(emp1);**

**Console.WriteLine(emp2);**

**}**

**}**

**OUTPUT:**

****

**INHERITANCE & OVERRIDING**

**CODE:**

**using System;**

**class Employee{**

**private string \_firstName;**

**private string \_lastName;**

**private double \_monthlySalary;**

**public Employee(string firstName,string lastName,double monthlySalary){**

**this.\_firstName=firstName;**

**this.\_lastName=lastName;**

**this.\_monthlySalary=monthlySalary;**

**}**

**public string firstName**

**{**

**get => \_firstName;**

**set =>\_firstName=value;**

**}**

**public string lastName**

**{**

**get => \_lastName;**

**set =>\_lastName=value;**

**}**

**public double monthlySalary**

**{**

**get => \_monthlySalary;**

**// set => \_monthlySalary=value;**

**set{**

**if(value<0){**

**\_monthlySalary=0.0;**

**}else{**

**\_monthlySalary=value;**

**}**

**}**

**}**

**public double yearlySalary**

**{**

**get{**

**return monthlySalary\*12;**

**}**

**}**

**public virtual void giveRaise(){**

**monthlySalary+=0.1\*monthlySalary;**

**}**

**public override string ToString()**

**{**

**return "\nFirst Name:"+firstName+" , Last Name:"+lastName+" , Yearly Salary:"+yearlySalary+"\n";**

**}**

**}**

**class PermanentEmployee : Employee{**

**private double \_HRA;**

**private double \_DA;**

**private double \_providentFund;**

**private double \_totalMonthlySalary;**

**private DateOnly \_joiningDate;**

**private DateOnly \_expectedRetirementDate;**

**public PermanentEmployee(string firstName,string lastName,double monthlySalary,double HRA,double DA,double providentFund,DateOnly joiningDate,DateOnly expectedRetirementDate):base(firstName,lastName,monthlySalary)**

**{**

**this.\_HRA=HRA;**

**this.\_DA=DA;**

**this.\_providentFund=providentFund;**

**this.\_joiningDate=joiningDate;**

**this.\_expectedRetirementDate=expectedRetirementDate;**

**this.\_totalMonthlySalary=monthlySalary+HRA+DA+providentFund;**

**}**

**public double HRA => \_HRA;**

**public double DA => \_DA;**

**public double providentFund => \_providentFund;**

**public DateOnly joiningDate => \_joiningDate;**

**public DateOnly expectedRetirementDate => \_expectedRetirementDate;**

**public double totalYearlySalary**

**{**

**get{**

**return \_totalMonthlySalary\*12;**

**}**

**}**

**public override void giveRaise(){**

**\_totalMonthlySalary+=0.1\*\_totalMonthlySalary;**

**}**

**public override string ToString()**

**{**

**return "\nFirst Name:"+firstName+" , Last Name:"+lastName+" , Total Yearly Salary:"+totalYearlySalary+" , HRA:"+HRA+" , DA:"+DA+" , Provident Fund:"+providentFund+"\n";**

**}**

**public void displayJoiningDate(){**

**Console.WriteLine("\n Name Of Employee:"+firstName+","+lastName+" Joining Date:"+joiningDate+"\n");**

**}**

**public void displayExpectedRetirementDate(){**

**Console.WriteLine("\n Name Of Employee:"+firstName+","+lastName+" Expected Retirement Date:"+expectedRetirementDate+"\n");**

**}**

**}**

**class EmployeeTest{**

**static void Main(string[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**//2**

**PermanentEmployee emp1=new PermanentEmployee("john","doe",10000,1000,1000,1000,new DateOnly(2001,10,1),new DateOnly(2023,10,1));**

**PermanentEmployee emp2=new PermanentEmployee("raj","jadhav",20000,2000,2000,2000,new DateOnly(2001,10,17),new DateOnly(2023,10,17));**

**Console.WriteLine("Before Raising Salary:\n");**

**Console.WriteLine(emp1);**

**Console.WriteLine(emp2);**

**emp1.giveRaise();**

**emp2.giveRaise();**

**Console.WriteLine("After Raising Salary:\n");**

**Console.WriteLine(emp1);**

**Console.WriteLine(emp2);**

**emp1.displayJoiningDate();**

**emp1.displayExpectedRetirementDate();**

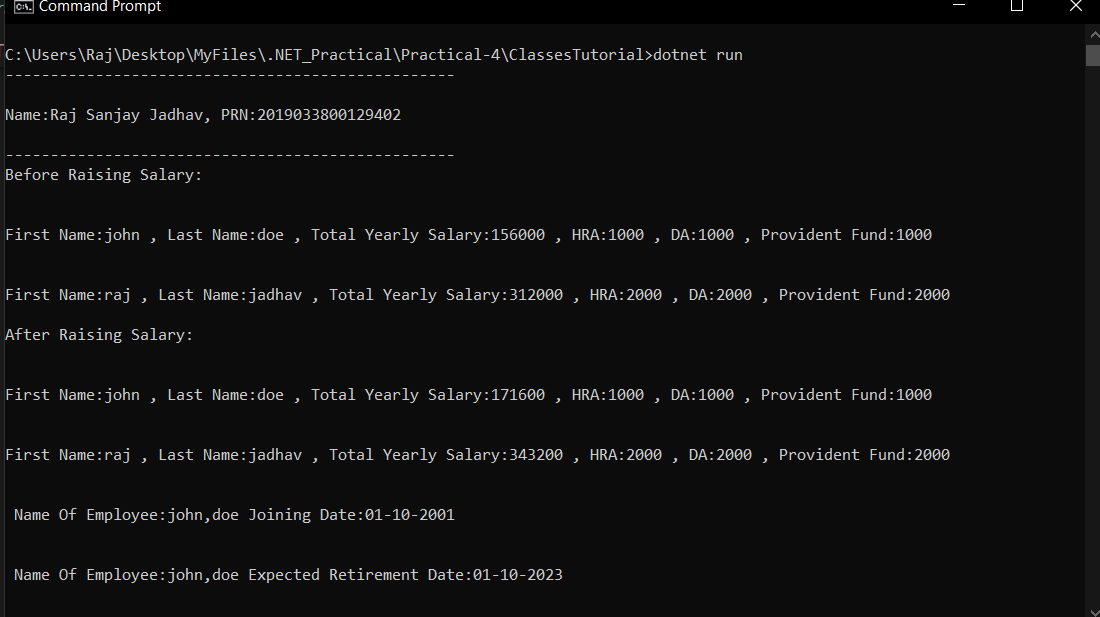
**emp2.displayJoiningDate();**

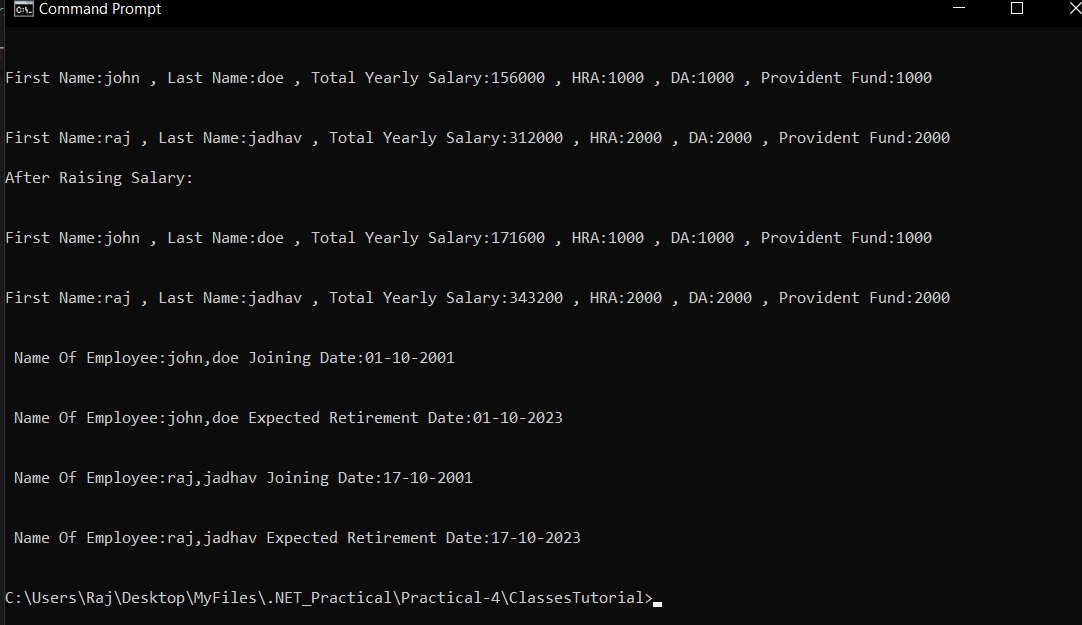
**emp2.displayExpectedRetirementDate();**

**}**

**}**

**OUTPUT:**

****

****

**METHOD HIDING**

**CODE:**

**using System;**

**public class My\_Family {**

**public void member()**

**{**

**Console.WriteLine("Total number of family members: 3");**

**}**

**}**

**// Derived Class**

**public class My\_Member : My\_Family {**

**public new void member()**

**{**

**Console.WriteLine("Name: Rakesh, Age: 40 \nName: Somya, "+"Age: 39 \nName: Rohan, Age: 20 ");**

**}**

**}**

**class MyClass {**

**static public void Main()**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

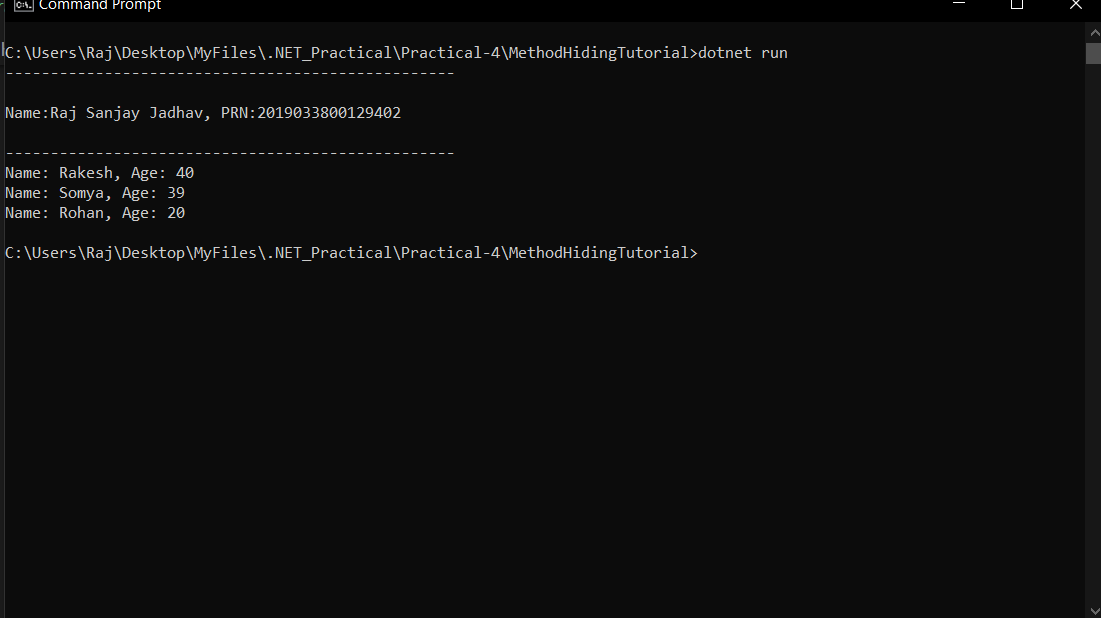
**My\_Member obj = new My\_Member();**

**obj.member();**

**}**

**}**

**OUTPUT:**

****

**METHOD OVERLOADING**

**CODE:**

**using System;**

**class MyClass {**

**public int multiply2(int a, int b)**

**{**

**return a\*b;**

**}**

**public int multiply3(int a, int b, int c)**

**{**

**return a\*b\*c;**

**}**

**public static void Main(String[] args)**

**{**

**Console.WriteLine("--------------------------------------------------");**

**Console.WriteLine("\nName:Raj Sanjay Jadhav, PRN:2019033800129402\n");**

**Console.WriteLine("--------------------------------------------------");**

**MyClass obj = new MyClass();**

**int mul1 = obj.multiply2(5, 2);**

**Console.WriteLine("Multiplication of the two "**

**+ "integer value : " + mul1);**

**int mul2 = obj.multiply3(5, 2, 3);**

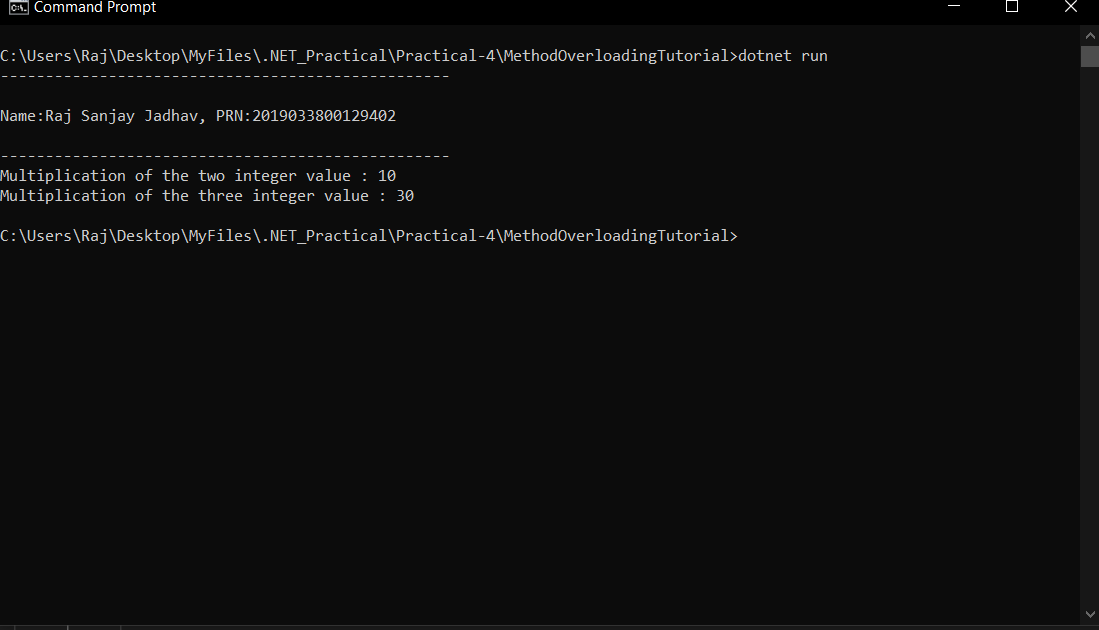
**Console.WriteLine("Multiplication of the three "**

**+ "integer value : " + mul2);**

**}**

**}**

**OUTPUT:**

****