|  |
| --- |
| **1.Research and write what is the use of sealed class. WACP to illustrate sealed class.** |
| “SEALED CLASS can’t be used as base class to any other derived class”  **Uses:**   * Sealed class is used to stop a class to be inherited. * You can’t derive or extend any class from it. * Sealed method is implemented so that no other class can overthrow it and implement its own method. |
| **Code:-**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  // Purpose: program to illustrate sealed method  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_4  {  sealed class Police  {  public static int Helpline = 100;  public string Getsecret()  {  return "007";  }  }  internal class Program  {  static void Main(string[] args)  {  Police p = new Police();  Console.WriteLine(p.Getsecret());  Console.WriteLine(Police.Helpline);  Console.ReadLine();  }  }  } |
| **Output:-** |

|  |
| --- |
| **2. Research and write what is the difference between normal properties and auto- implemented properties.** |
| **NORMAL PROPERTY:**   * Its refer to other variables. * We can use both get and set. * A property with only get is readonly. * A property with only set is writeonly.   **AUTO-IMPLEMENTED PROPERTY:**   * When there is no additional logic in the property accessors. * We can use only get (its mandatory). * A property with only get is readonly. * Here set property is optional. |
| **WACP to illustrate normal properties.** |
| **CODE:**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  // Purpose: illustrate normal properties  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_6  {  class employee  {  private int Id;  private string Name;  private string Designation;  private int Salary;  public int id  {  get { return Id; }  set { Id = 101; }  }  public string name  {  get { return Name; }  set { Name = "JEEVITHA"; }  }  public string designation  {  get { return Designation; }  set { Designation = "manager"; }  }  public int salary  {  get { return Salary; }  set { Salary = 10000; }  }  }  internal class Program  {  static void Main(string[] args)  {  employee employee = new employee();  employee.id = 101;  employee.name = "JEEVITHA";  employee.designation = "manager";  employee.salary = 10000;  Console.WriteLine(employee.id);  Console.WriteLine(employee.name);  Console.WriteLine(employee.designation);  Console.WriteLine(employee.salary);  Console.ReadLine();  }  }  }  **Output:** |
| **WACP to illustrate auto implemented properties.** |
| **CODE:**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  // Purpose: illustrate auto implemented properties  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_5  {  internal class Program  {  class Employee  {  public int Id { get; set; }  public string Name { get; set; }  public string Designation { get; set; }  public int Salary { get; set; }  }  static void Main(string[] args)  {  Employee emp = new Employee();  emp.Id = 78;  Console.WriteLine($"id = {emp.Id}");  emp.Name = "JEEVITHA";  Console.WriteLine($"name = {emp.Name}");  emp.Designation = "developer";  Console.WriteLine($"designation = {emp.Designation}");  emp.Salary = 50000;  Console.WriteLine(emp.Salary);  Console.ReadLine();  }  }  }  **OUTPUT:** |

|  |
| --- |
| **4. WACP to check if the number is prime or not using logic discussed in the class. [ use break; ]** |
| **Code:-**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  // Purpose: to check the num is prime or not [ use break; ]  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_3  {  class Prime  {  public void Isprime()  {  int n=78 , i;  for (i=2;i<n;i++)  {  if (n % i == 0)  break;  }  if (i == n)  Console.WriteLine("IT IS A PRIME NUMBER");  else  Console.WriteLine("IT IS NOT A PRIME NUMBER");  }  }    internal class Program  {  static void Main(string[] args)  {  Prime prime = new Prime();  prime.Isprime();  Console.ReadLine();  }  }  } |
| **Output:-** |

|  |
| --- |
| **5. print numbers from 1 to 30 and skip the numbers divisible by 3. [ use continue; ]** |
| **Code:-**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  //Purpose: print nums from 1 to 30 & skip the nums divisible  // by 3 [ using continue; ]  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_1  {  internal class Program  {  static void Main(string[] args)  {  for (int i=1; i<=30; i++)  {  if (i % 3 == 0)  continue;  Console.WriteLine(i);  }  Console.ReadLine();  }  }  } |
| **Output:-** |

|  |
| --- |
| **6. find the first number after 1000 which id divisible by 97. [ use forloop and break; ]** |
| **Code:-**  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // Author: JEEVITHA  // Purpose: find the first num after 1000 which is divisible by 97.  // [ using for loop and break ]  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  namespace Project\_2  {  internal class Program  {  static void Main(string[] args)  {  for (int i=1000; i<1097; i++)  {  if (i%97 == 0)  {  Console.WriteLine(i);  break;  }    }  Console.ReadLine();  }  }  } |
| **Output:-** |