|  |
| --- |
| Day14 Morning Assignment  By  Anusha Bellala  10-9-2-2022 |

|  |
| --- |
| 1. Research and write what is the use of sealed class.  WACP to illustrate sealed class. |
| Sealed classes are used to restrict the users from inheriting the class. A class can be sealed by using the **sealed**keyword. The keyword tells the compiler that the class is sealed, and therefore, cannot be extended. No class can be derived from a sealed class.  The following is the syntax of a sealed class :  sealed class class name  {  // data members  // methods  .  . |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day14Project4  {  sealed class SealedClass  {  // Calling Function  public int Add(int a, int b)  {  return a + b;  }  }  internal class Program  {  static void Main(string[] args)  {  // Creating an object of Sealed Class  SealedClass slc = new SealedClass();  // Performing Addition operation  int total = slc.Add(6, 4);  Console.WriteLine("Total = " + total.ToString());  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 2. Research and write what is the difference between normal properties and auto-implemented properties. |
| **Normal Properties:**  Property in C# is a member of a class that provides a flexible mechanism for classes to expose private fields. Internally, C# properties are special methods called accessors. A C# property have two accessors, get property accessor and set property accessor. A get accessor returns a property value, and a set accessor assigns a new value. The value keyword represents the value of a property. |
| **Auto-Implemented Properties:**   * C# 3.0 includes a concept of auto-implemented properties that requires no code inside get and set methods of the class properties. It makes code concise and readable. * The C# compiler creates private fields correspond to the properties and are accessible using the get and set methods. |

|  |
| --- |
| 2.)a) WACP to illustrate normal properties |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace NormalVariables  {  class Employee  {  private int id;  private string name;  private string designation;  private int salary;  public int Id  {  get { return id; }  set { id = value; }  }  public string Name  {  get { return name; }  set { name = value; }  }  public string Designation  {  set { designation = value; }  }  public int Salary  {  get  {  salary = (designation == "M") ? 30000 : 100000;  return salary;  }  }  }  internal class Program  {  static void Main(string[] args)  {  Employee emp = new Employee();  emp.Designation = "HR";  Console.WriteLine(emp.Salary);  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 2.)b) WACP to illustrate auto-implemented properties |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace AutoImplemented  {  class Student  {  // Auto-implemented Properties  public int ID { get; set; }  public string Name { get; set; }  public string Email { get; set; }  }  internal class Program  {  static void Main(string[] args)  {  Student student = new Student();  // Setting properties  student.ID = 101;  student.Name = "Anusha Bellala";  student.Email = "Anu@example.com";  // Getting properties  Console.WriteLine(student.ID);  Console.WriteLine(student.Name);  Console.WriteLine(student.Email);  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 4. WACP to check if the number is prime or not using logic discussed in the class  HINT : use break; |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day14Project1  {  internal class Program  {  static void Main(string[] args)  {  int n = 11,j;  for (j = 2; j < n; j++)  {  if (n % j == 0)  break;  }  if(j==n)  {  Console.WriteLine("{0} is Prime Number", n);  }  else  {  Console.WriteLine("{0} is not a Prime Number",n);  }  Console.ReadLine();  }  }  } |
| Output: |

|  |
| --- |
| 5. print numbers from 1 to 30 and skip the numbers divisible by 3  HINT : use continue; |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day14Project2  {  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 is divisible by 97.  HINT : use for loop and break |
| Code: |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Day14Project3  {  internal class Program  {  static void Main(string[] args)  {  for(int i=1000;i<=1095;i++)  {  if (i % 97 == 0)  {  Console.WriteLine(i);  break;  }    }  Console.ReadLine();  }  }  } |
| Output: |