Faculty of Computing, Online Examinations 2021

STUDENT NAME	E.M. Ruchira Amantha Edirisinghe		
INDEX NUMBER (NSBM)	21487	YEAR OF STUDY AND SEMESTER	Year 1 Semester 2
MODULE NAME (As per the paper)	Object Oriented Programming with C#		
MODULE CODE	CS107.3		
MODULE LECTURER	Mr. Pramudya Thilakarathne	DATE SUBMITTED	13 – Sep - 2021

For office purpose only:

GRADE/MARK	
COMMENTS	

Declaration

PLEASE TICK TO INDICATE THAT YOU HAVE SATISFIED THESE REQUIREMENTS

- ✓ I have carefully read the instructions provided by the Faculty
- ✓ I understand what plagiarism is and I am aware of the University's policy in this regard.
- ✓ I declare that the work hereby submitted is my own original work. Other people's work has been used (either from a printed source, Internet or any other source), has been properly acknowledged and referenced in accordance with the NSBM's requirements.
- ✓ I have not used work previously produced by another student(s) or any other person to hand in as my own.
- ✓ I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.
- ✓ I hereby certify that the individual detail information given (name, index number and module details) in the cover page are thoroughly checked and are true and accurate.

I hereby certify that the statements I have attested to above have been made in good faith and are true and correct. I also certify that this is my own work and I have not plagiarized the work of others and not participated in collusion.

Date: 13 – Sep - 2021 **E- Signature: [illegible]

**Please attach a photo/image of your signature in the space provided.

Question 1

1. C# access modifiers are used to determine the range of availability of a class member or class type itself. For example, the public class is accessible to everyone without restrictions, while the internal class is accessible only to the assembly.

   Access specifiers are an important part of an object-oriented program. Access specifiers are used to implement OOP encapsulation. Access specifiers allow you to define who is doing it or who can access certain features.

   There are 6 types of Access Modifiers,
   - Public
   - Private
   - Protected
   - Internal
   - Internal Protected
   - Private Protected

   Public Specific Modifiers have no limits on access to public members. Access to private access modifiers is limited within class definition. This is a type of automatic access converter if there is no official reference.
   In Protected Access modifiers, Access is limited within the definition of a class and any category that inherits from the class.
   For Internal Access Modifiers, their access is limited only to classes defined within the current project organization.
   For Internal Protected Modifiers, their access is limited to the current assembly and the categories based on the containing class. All members of the current project and all members of the derived class can have flexible access. Private Protected Access Modifiers' access is limited to the containing class or types based on the containing class within the current assembly.

2. The variable ‘Number’ that can be accessed by a code in the same class, has a private access modifier. As the ‘Number’ variable can access this variable through encapsulation, the variable has a private class.

   The variable ‘Name’ has a protected access modifier that grants access only to the members or types in the same class or the class that has been derived from that class. Inheritance is used in accessing these protected methods.

3.
```
class EncappedClass
    {
        private int MyMarks;

        public void setMarks(int marks)
        {
            MyMarks = marks;
        }

        public int setMarks()
        {
            return MyMarks;
        }
    }
```

4.
```
class Person
 {
  private int age;
  private string name;
  private double height;

      public void SetValues(int a, string b, double c)
      {
    age = a;
    name = b;
    height = c;
      }

public int getage()
      {
          return age;
      }
public string getname()
      {
          return name;
      }
public double getheight()
      {
          return height;
      }

 }
```

Question 2

1. Error in C# is an indication of an unexpected condition that occurs due to the lack of system resource whereas an exception is an issue in a program preventing the normal flow of the program.

   Exceptions in C# are recoverable, but errors in C# are not recoverable.

   Exceptions has the ability to manage an exception in a program using keywords whereas there's no way to handle an error using the program in an error in C#.

2. Exception handling is a method of controlling runtime programming error in a structured and control manner. It is used in defining a try block Exception handling is so much important because it kindly treats the unwanted event, which is different so that the program code can still make sense to the user.

3. TRY:- A try block indicates a part of code for which particular exceptions are lifted.

   CATCH:- An exception handler in a program captures an exception at the point in the program where you expects to handle the problem. The catch keyword indicates an exception that has been caught.

   FINALLY:- if an exception is thrown or not, to execute a collection of statements, the finally block is used. If you open a file, for example, you must close it whether or not an exception is triggered.

4.
```
class Program
  {
    static void Main(string[] args)
    {
      int[] MyArray = new int[5];
      try
   {
        for (int x = 0; x <= 5; x++)
        {
          MyArray[x] = x;
        }
      }
```

```
         catch (Exception ex)
      {
           Console.WriteLine(ex);
      }

      for (int x = 0; x < 5; x++)
         {
        Console.WriteLine(MyArray[x]);
        }
         Console.ReadKey();
      }
     }
```

Question 3

1. The process of obtaining all the properties of one class into another class is known as “inheritance in OOP”. It has the ability to create a class that inherits the behavior and attributes to the children class from the parent class. It has many advantages such as helps to reduce the redundancy of a code, provides reusability, improves the readability of a code, reduces the size of the code and easy to manage.

2. Protected Access Specifiers hide the variable of its members and its functions in other classes and objects. This type of variable or function can only be found in a child class. It is very important when using an asset. The protected modifier allows the derived class to inherit the variables and methods in the parent class. This type of variable or function can only be found in a child class. They have access to its methods and variables in the same class or to the inherited class from the main class.

3.

   -
     ```
     class Person
      {
        private string name;
        private int id;

        public void setName(string uname)
        {
          name = uname;
        }
     ```

```
    public string getName()
    {
        return name;
    }

    public void setID(int uid)
    {
        id = uid;
    }
    public int getID()
    {
        return id;
    }
}
```

- class Lecturer : Person

```
{
    private string programme;

    public void setProg(string uprog)
    {
        programme = uprog;
    }
    public string getProg()
    {
        return programme;
    }
}
```

- class Student : Person

```
{
    private string course;

    public void setCourse(string ucourse)
    {
        course = ucourse;
    }
    public string getCourse()
    {
        return course;
    }
}
```

Question 4

```
string conStr = @"DataProvider=.\SQLEXPRESS;Data
Source=C:\NSBM\2ndSemFinal\C#\School.mdf";
        string qry = "SELECT StudentID,Name,Age,DegreeProgram FROM
Student_Details";

        try
        {
          SqlDataAdapter objDataAdapt = new SqlDataAdapter(qry, conStr);
          DataSet objdataset = new DataSet();
          objDataAdapt.Fill(objdataset, "Student_Details");
          DataGridView1.DataSource = objdataset.Tables["Student_Details"];
        }
        catch(SqlException except)
        {
          MessageBox.Show("Error Occrured ! " + except.ToString());
        }
```

Question 5

1.

Constructor	**Method**
Has No return type.	Can return a value or not.
Induced implicitly.	Induced explicitly.
Has Access Specifiers. (Public)	Has Access Specifiers. (Public, Private and Protected)
Has the same name as the class name.	Can have any other names other than keywords.
Used to initialize an object.	Consists of code to be executed.

2.
   ```
   class employee
     {
       public string empname;

       public employee (string uname)
       {
         empname = uname;
       }
     }
   ```

3. Constructors execute every time a new class environment is created. The Constructor, as a method, contains a set of commands that are made when creating an object. Used to provide data for members of the same class initial values.

4.

   -
     ```
     class Student
     {
       public string name;
       public int age;
       public string gender;

       public Student(string studentName, int studentAge, string
     studentGender)
       {
         name = studentName;
         age = studentAge;
         gender = studentGender;
       }
     }
     ```

   -
     ```
     class School
      {
        public void passdetails()
        {
          Student firststudentinfo = new Student("John", 18, "M");
          Student secondstudentinfo = new Student("Rose", 22, "F");
        }
      }
     ```

5. .

Connecting Forms

```
Registrationform object = new Registrationform();

        this.Hide();
        object.Show();
```

Validating Forms

```
if (tname.Text == "" || tage.Text == "" || tgneder.Text == "")
        {
          MessageBox.Show("Error: Null Values has been entered!");
          return;
        }

try
        {
          string name = tname.Text;
          int age = Convert.ToInt32(tage.Text);
          string gender = tgender.Text;
        }

        catch (System.FormatException exept)
        {
          MessageBox.Show("Invalid Input in Age Box ! \n" + exept);
        }
```