

## Faculty of Computing, Online Examinations 2021

STUDENT NAME	R. B. Gihini Nimthara Suraweera		
INDEX NUMBER (NSBM)	21403	YEAR OF STUDY AND SEMESTER	Year-01 sem-02
MODULE NAME (As per the paper)	Object Oriented Programming with C#		
MODULE CODE	CS107.3		
MODULE LECTURER	Mr.Pramudya Thilakarathne	DATE SUBMITTED	13.09.2021

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: 13.09.2021

\*\*E- Signature:



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

### **Question-01**

01) Access modifications are keywords that define the accessibility of a program member, class, or data type. These are mainly used to limit the handling of unwanted data by external programs or classes.

02)

<b>Private</b>	<b>Protected</b>
Private members are declared with the keyword <code>colon(:)</code> .	Protected members are expressed by the keyword <code>protected</code> by the <code>colon(:)</code> characters.
Private members can also access through the friendship process.	Protected members may not be able to access the activity of friends.
Private members may enter the same class in which they are published.	Protected members can access the same class and the derivative/ sub/child class.

03) To access a private variable from another class, encapsulation must be used, and within the class we create two classes that include the private variable. To store the value of the variable from the class to which the private variable should be accessed and retrieve the value of the class to which access is required.

```
private int MyMarks;  
  
public void setMarks(int MyMarksOfUser)  
  
//to equalize the value  
  
{  
  
    MyMarks = MyMarksOfUser;  
  
}  
  
Public int getMarks;  
  
{  
  
    return MyMarks;  
  
}
```

04)

Class person

```
{  
  
    private int age;  
  
    private string name;  
  
    private double height;  
  
  
    public void SetValues(int ageofUser, string nameofUser, double heightofUser)  
    {  
  
        age = ageofUser;  
  
        name = nameofUser;  
  
        height = heightofUser;  
  
    }  
    public int getAge()  
    {  
  
        return age;  
  
    }  
    public string getName()  
    {  
  
        return name;  
  
    }  
    Public double getHeight()  
    {  
  
        return height;  
  
    }  
}
```

## Question-02

01)

Exception	Logical error
Often the only clue to the existence of logic errors is that static analyzers can sometimes identify them but produce erroneous solution.	A logical error is an error that causes a program to run incorrectly but doesn't end abnormally.
The programmer can handle the error.	The programmer can't handle the error.

02) When handling an exception (when caught), not only is there a specific error in the handling mechanism (as in conventional procedural programming) the exception can be captured. At OOP, it is recommended to use throws to manage errors or unforeseen occurrences during program execution.

03) Try-The attempt statement allows you to define a piece of code that should be checked for errors during execution.

Catch- A program gets an exception with the exception handler of the program you want to troubleshoot. Catch keyword indicates capture of an exception.

Finally- Final clause is used to activate a given set of statements, with or without an exception.

04) class program

```
{  
  
    static void main(string [] args)  
    {  
  
        Int[] MyArray = new int[5];  
  
        try  
        {  
            for(int x=0;x<=5;x++)  
            {  
                MyArray[x] = x;  
            }  
            for (int x=0;x<5;x++)
```

```

        {
            Console.WriteLine(MyArray[x]);
        }
    }

    Catch (Exception e)
    {
        Console.WriteLine(e);
    }

    Finally
    {
        This.close();
    }

    Console.ReadKey();
}
}

```

### **Question-03**

01) Inheritance is the acquisition of a new class based on the transformation of an existing class into a parent or basic super class and the new class into a child or a derivative class. And there may be several derivative classes for all mother class members and one elementary class who may have access to the children's class.

02) The secure access feature allows a member to access the class, friends, and derivative class they belong to. However, protected members cannot access from outside the classroom.

03)

class Person

```

{
    private string name;

    private int id;

```

```
public void setName(string UserName)

{

    name = UserName;

}

public string getName()

{

    return name;

}


public void setID(int UserID)

{

    id = UserID;

}

public int getID()

{

    return id;

}

}
```

class Lecturer : Person

```
{

    private string programme;


    public void setProg(string UserProgramme)

    {

        programme = UserProgramme;

    }

    public string getProg()

    {

        return programme;

    }

}
```

```
}
```

```
class Student : Person
{
    private string course;

    public void setCourse(string UserCourse)
    {
        course = UserCourse;
    }

    public string getCourse()
    {
        return course;
    }
}
```

#### **Question-04**

##### **Class DataSet**

```
{
SqlConnection connect = new SqlConnection(@"Data source =
DataProvider=.\\SQLEXPRESS;DataSource=E:\NSBM\School.mdf);

private void con()
{
    Connect.Open();
}

Private void discon()
{
    Connect.close()
}

Public DataView Gridview(string query)
{
```

```

Con();

sqlDataAdapter ad = new sqlDataAdapter(query, connct);

DataView objdataview = new DataView();

ad.Fill(objdataview);

discon();

return dataview;

}

```

### Public class student

```

{

Dataset objDataBase = new DataSet();

Private void button_click(object sender, EventArgs e)

{

    String query = "SELECT * FROM Student_details";

    Data_student.columns[0].HeaderText = "Name";

    Data_student.columns[1].HeaderText = "Age";

    Data_student.columns[2].HeaderText = "Degree";

}

}

```

### Question-05

01)

Constructors	Methods
A constructor is a block of code that initializes a newly created object.	A method is a collection of statement that provide value in execution.
The name of a constructor Should be the same as the name of the class.	The name of the method can be anything.
There is no return type for a constructor.	There must be a return type to a method.
A class can have many constructors but should not have the same parameters.	A class can have many methods but should not have the same parameters.
Subclasses cannot be inherited by a constructor.	Subclasses can inherit a method.



02)

```
class Employee
```

```
{
```

```
    public string name, age;
```

```
    //parameterized constructor
```

```
    Public Employee(string a, string b)
```

```
    {
```

```
        name = a;
```

```
        age = b;
```

```
    }
```

```
}
```

```
//main method
```

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

```
    {
```

```
        //the constructor will be called automatically once the instance of the class created
```

```
        Console.WriteLine("Enter the Employer Name");
```

```
        String name=Console.ReadLine();
```

```
        Console.WriteLine("Enter the Employer Age");
```

```
        int age = int.parse(Console.ReadLine());
```

```
        Employee employee = new Employee(name,age);
```

```
        Console.ReadLine();
```

```
    }
```

03)

A constructor is a special mode of class that is automatically called when a class opportunity is created. As well as methods, a constructor also has a collection of instructions that work when creating an object. It is used to assign basic values to data members of the same class.

04) Class students

```
{  
  
    public string name;  
  
    public int age;  
  
    public char gender;  
  
    public student(string name, int age, char gender)  
  
    {  
  
        Name = Name;  
  
        Age = Age;  
  
        Gender = Gender;  
  
    }  
  
}  
  
Class school  
  
{  
  
    Student student1 = student(18,"John",'M');  
  
    Student student2 = student(22,"Rose",'F');  
  
}
```

05)

namespace Navigation

```
{  
  
    public partial class Form1 : Form  
  
    {  
  
        public Form1()  
  
        {  
  
            InitializeComponent();  
  
        }  
  
  
  
  
        private void button1_Click(object sender, EventArgs e)  
  
        {  
  
            Horm ObjHome = new Horm();  
  
            ObjHome.Show();  
  
        }  
  
    }  
  
}
```

this.Hide();

}

}

}