# Description: _LITfinalLOGO

# SUMMER EXAMINATIONS 2014

**Friday, 9th May 2014, 14.30 p.m. – 16.30 p.m.**

**KSDEV\_8\_Y2**

**Course:** Bachelor of Science (Hons) in Software Development

**Year:** One

**Subject:** Object Oriented & GUI Programming

**Time Allowed:** 2 Hours

**Instructions: 1.** You **MUST** answer **Q.1**

Answer **ANY OTHER TWO** questions.

**2.** Marks for **Q.1** are **50 marks**.

All other questions are **25 marks**.

**3.** Start each question on a new page.

**4.** Write the question number at the top of each page.

**5.** Circle the numbers of the questions you answer at the front of your answer book.

**Additional Attachments Exam Materials to accompany this paper:**

### Code for Q.3

1. **Code for Q.4**

**Internal Examiners: External Examiners:**

Tom Costello Mr Brian Gillespie

**Q.1**  Answer any FIVE of the following **(10 marks per part)**

1. A Windows API program contains a function **WinMain**. Describe the role of the code that will be inserted in this function when a Win32 Project is created in Visual Studio
2. With reference to the .NET framework, write a brief note on each of the following
   1. Assemblies
   2. Garbage Collection
3. In a Windows Forms Application, Visual Studio will add a class derived from the **class System.Windows.Forms.Form** to the project. To that class it will add a method

**private void InitializeComponent( )**

Describe the role of this function.

1. Describe the role of the following method from a **class MyForm:Form**

**private void buttonExtend\_Click(object sender, EventArgs e)**

**{**

**this.listBoxNames = new System.Windows.Forms.ListBox();**

**this. listBoxNames.FormattingEnabled = true;**

**this. listBoxNames.Location = new System.Drawing.Point(20,30);**

**this. listBoxNames.Name = "listBox2";**

**this. listBoxNames.Size = new System.Drawing.Size(60, 40);**

**this. listBoxNames.TabIndex = 3;**

**this.Controls.Add(this. listBoxNames);**

**}**

1. Describe the role of the highlighted code in the following code fragment from a Windows Forms program in C#

public partial class formContainer : Form

{

public formContainer()

{

**Application.ApplicationExit**

**+= new EventHandler(this.OnApplicationExit);**

InitializeComponent();

}

**private void OnApplicationExit(object sender, EventArgs e)**

**{**

**BinaryFormatter bformatter = new BinaryFormatter();**

**FileStream studentFile = new FileStream("Students.bin",**

**FileMode.Create, FileAccess.Write);**

**bformatter.Serialize(studentFile, studentList);**

**studentFile.Close();**

**}**

}

1. “WPF applications use a markup plus code-behind model” . Describe what is meant by this statement and discuss the advantages of using such a model.
2. The following extract from a XAML file contains examples of each of Object Element Syntax, Attribute Syntax, Property Element Syntax and Content Syntax.

Write out two examples of each type, explaining its role.

**<Grid>**

**<Grid.Background>**

**<LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">**

**<GradientStop Color="Black" Offset="0"/>**

**<GradientStop Color="White" Offset="1"/>**

**</LinearGradientBrush>**

**</Grid.Background>**

**<Button HorizontalAlignment="Left" Margin="318,180,0,0"**

**VerticalAlignment="Top" Width="75" Click="Button\_Click">**

**<Button.Background>**

**<LinearGradientBrush EndPoint="0,1" StartPoint="0,0">**

**<GradientStop Color="#FFF3F3F3" Offset="0"/>**

**<GradientStop Color="#FF1BD0E2" Offset="1"/>**

**</LinearGradientBrush>**

**</Button.Background>**

**Go**

**</Button>**

**<ListBox HorizontalAlignment="Left" Height="150" Margin="166,78,0,0"**

**VerticalAlignment="Top" Width="109" Opacity="0.5">**

**<ListBox.Background>**

**<LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">**

**<GradientStop Color="Black" Offset="0.043"/>**

**<GradientStop Color="#FFD9ECEE" Offset="0.168"/>**

**</LinearGradientBrush>**

**</ListBox.Background>**

**</ListBox>**

**</Grid>**

1. With respect to control layout in WPF briefly describe each of the following layout containers
   * 1. Canvas
     2. DockPanel
     3. Grid
     4. StackPanel
     5. WrapPanel

**(Total 50 Marks)**

**Q. 2**

Write suitable definitions in C# for the classes **Person, Student and Graduate** in the class hierarchy shown below, given the following conditions:

* The class Person should be an abstract base class and the Student and Graduate classes should be concrete classes **(3 marks)**
* The class Person should have
  + a string field name and an integer field id **(1 mark)**
  + a constructor that is passed a name and an id **(2 marks)**
  + get properties for the two fields **(2 marks)**
  + an abstract method DisplayDetails **(1 mark)**
* The class Student should have
  + a string field course and an integer field year **(1 mark)**
  + a constructor that is passed a name, id, course and year

**(3 marks)**

* + get and set properties for the fields **(3 marks)**
  + a defined method DisplayDetails **(1 mark)**
* The class Graduate should have
  + a field qualification of type string **(1 mark)**
  + a constructor that is passed a name, id, course, year and qualification **(3 marks)**
  + get and set properties for the field **(3 marks)**
  + a defined method DisplayDetails **(1 mark)**

##### class Person

##### class Student

##### class Graduate

**(Total 25 Marks)**

**Q.3**

1. Describe the role of interfaces in C# programming

**(5 marks)**

1. **Attachment A** shows the definitions for two classes **Sample** and **MyList** that implement the **ICloneable** interface. Explain in detail, commenting on the code used, the differences between the implementations of the **ICloneable** interface for the classes. **(20 marks)**

**(Total 25 Marks)**

**Q. 4** A student’s marks in a subject are composed of two elements, a continuous assessment mark and a final exam mark. Assume that the maximum continuous assessment mark allowed is 40 and the maximum exam mark is 60.

**Attachment B** contains the code for a **class Student** and also for a **class StudentException**. The second class may be used to support structured exception handling.

**Attachment B** also contains some sample test code in the method **Main** of the **class Program**.

Re-write the method **Main** with suitable structured exception handling code included. An exception should be thrown if a mark greater than the maximum allowed is entered.

Note also that a **System**.**FormatException** exception will be thrown if a non-numeric string is passed to **int.Parse( )**.

**(Total 25 Marks)**

**Attachment A**

**public class Sample : ICloneable**

**{**

**private int id;**

**private double weight;**

**public Sample(int id, double weight)**

**{**

**this.id = id;**

**this.weight = weight;**

**}**

**public int Id { get{return id;} }**

**public double Weight{ get{return weight;} set{weight = value;} }**

**public object Clone()**

**{**

**return MemberwiseClone();**

**}**

**}**

**public class MyList:ICloneable**

**{**

**private ArrayList theList;**

**string name;**

**public MyList()**

**{**

**theList = new ArrayList();**

**name = "Samples";**

**}**

**public ArrayList TheList{ get{return theList;} }**

**public string Name{ get{return name;} }**

**public object Clone()**

**{**

**MyList newList = new MyList();**

**newList.name = "Original Samples";**

**foreach (Sample sample in theList)**

**{**

**newList.TheList.Add(sample.Clone() as Sample);**

**}**

**return newList;**

**}**

**}**

**Attachment B**

**public class Student**

**{**

**private string name;**

**private int caMark;**

**private int examMark;**

**public Student(string name)**

**{**

**this.name = name;**

**caMark = 0;**

**examMark = 0;**

**}**

**public string Name{ get{return name;} }**

**public int CaMark{ get{return caMark;} set{caMark = value;} }**

**public int ExamMark{ get{return examMark;} set{examMark = value;} }**

**public int TotalMark()**

**{**

**return caMark + examMark;**

**}**

**}**

**public class StudentException**

**{**

**private string elementType;**

**private int mark;**

**public StudentException(string elementType, int mark)**

**{**

**this.elementType = elementType;**

**this.mark = mark;**

**}**

**public void ShowReason()**

**{**

**Console.WriteLine(“EXCEPTION THROWN: Maximum marks exceeded”);**

**Console.WriteLine("{0} marks to element {1} not allowed",**

**mark, elementType);**

**}**

**}**

**class Program**

**{**

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

**{**

**Student s = new Student("Fred");**

**Console.Write("Enter CA Mark :");**

**int CaMark = int.Parse(Console.ReadLine());**

**s.CaMark = caMark;**

**Console.Write("Enter Exam Mark :");**

**int ExamMark = int.Parse(Console.ReadLine());**

**s.ExamMark = examMark;**

**Console.WriteLine("{0} has total of {1}", s.Name, s.TotalMark());**

**}**

**}**