# Dublin Business School

# Assessment Brief

# Assessment Details

|  |  |
| --- | --- |
| Unit Title: | **Object Oriented Programming** |
| Unit Code: | B8IT117 |
| Unit Lecturers: | Claire Caulfield |
| Level: | 8 |
| Assessment Title: | Practical Programming |
| Assessment Number: | 1 |
| Assessment Type: | Individual Submission |
| Restrictions on Time/Length : | N/A |
| Individual/Group: | Individual |
| Assessment Weighting: | 50% (5% UML & 30% Code and 15% Technical Merit) |
| Issue Date: | 2 Oct 2018 |
| Hand In Date: | 31 Nov 2018 |
| Mode of Submission: | On-line **ONLY** |

# Assessment

The following is the assignment as set out for the module of Object-Oriented Programming (OOP).

The assignment is an individual assignment.

Code must be written in C# and supported by UML documentation.

The following table illustrates the percentage allocation for each individual part of the assignment.

|  |  |  |
| --- | --- | --- |
| Task | Part | Breakdown of Marks (100%) |
| Task 1 Documentation | UML | 10% |
| Task 2 | C# Code | 60% |
| Task 3 | Technical Merit (Add code and something outside of the school) | 30% |

**Assessment Specification**

Dublin Business School has decided to implement a new management software package; your task is to present a solution in a console application.

Write a Console application in C# to implement the details given in the assignment specification below.

Your programme should allow for services such as add student and add teacher. (use list or collection)

Create an inheritance hierarchy for this application, below are the minimum amount of classes to use in your programme.

Bear Min to Get 60% of the mark

1. **Person Class**

2. **Student Class** (should inherit the functionality of base class Person)

3. **Employee Class** (should inherit the functionality of base class Person)

4. **Teacher Class** (should inherit the functionality of base class Employee)

* A class Person with attributes Name, Phone and Email.
* A class Student which Inherits from Person with attributes Status and StudentId. A student’s status can either be a cPostgrad or an Undergrad.
* Create a constructor for Student such that all attributes belonging to a Student can be assigned via the constructor. (overload the constructor)
* Create a class Employee which inherits from Person with an attribute of Salary
* Create a class Teacher which inherits from Employee with an attribute of Subject Taught
* Create a constructor for Teacher such that all attributes belonging to a Teacher can be assigned via the constructor. (overload the constructor)
* Provide at least one implementation of an override ToString method (You have to override – Bear min)

# UML Documentation 10%

Create UML documentation which describes the details given above. You will need to determine for yourself any details which are not given in the specification. You should submit the following:

1. Use Case Diagram
2. Class Diagram

# Technical Merit

You can add additional functionality to your application; you can choose any additional Classes you wish whether it was something we covered in class or something you researched yourself.

This must be included in the Use Case Diagram and Class Diagram.

Include a paragraph in your documentation on the technical merit you applied to your programme.

* It basically explaining anything extra you do e.g. The ability to remove all student

(Any thing you have included extra include it in Your use Case Diagram and Class Diagram…There is no database to)

Create a folder with your name and student number e.g. Peter Coker 1725266

What to put in the folder that is

Visual Code

Word Document into one Folder

**Note: Do not include database functionality.**

**Features for the application**

* The ability to remove all student
* **Unique Student Id number**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criteria/**  **Mark** | **< 40** | **40 - 49** | **50 - 59** | **60 – 69** | **70 +** |
| **UML** | Difficult to read; not related to code | Messy, needs tidying; partially related to code | Some errors and mistakes; a lot related to code | Few errors and mistakes; fully related to code | Presented excellently with no errors |
| **C# & Technical Merit** | Insufficient or incomplete code structure | Some but insufficient and poorly structured code which doesn’t solve the problem | Sufficient solves problem but lack of attention to coding practices | Well-structured code that solves the problem | Excellent solution to problem |

**General Requirements for Students:**

1. Please read carefully the following problems. You are expected to submit both the UML diagrams and C# code as part of your submission. Save all your files in a folder and give it name: yourName. Compress the folder and upload it on Moodle.
2. It is your responsibility to ensure your files are uploaded correctly.
3. Students are required to retain a copy of their assignment.
4. When an assignment is submitted, it is the student’s responsibility to ensure that the file is in the correct format and opens correctly.
5. Students should refer to the assessment regulations in their Course Guide.
6. DBS penalises students who engage in academic impropriety (i.e. plagiarism, collusion and / or copying). Please refer to the referencing guidelines on Moodle for information on correct referencing.
7. Extensions to assignment submission deadlines will be granted in exceptional circumstances only. The appropriate “Application for Extension” form must be used and supporting documentation (e.g. medical certificate) must be attached. Applications for extensions should be made directly to the Head of Year or Programme Leader in advance of the deadline date.
8. Late submissions will be penalised 2 marks per day.