Untitled-1**SCHOOL OF ENGINEERING AND COMPUTER SCIENCE  
Computer Science**

**AssessmenT DESCRIPTION 2019/20**

**(Exam tests WORTH ≤15% and Coursework)**

**MODULE DETAILS:**

|  |  |  |  |
| --- | --- | --- | --- |
| Module Number: | 700103\_A19 | Trimester: | 1 |
| Module Title: | Object Oriented Design and Development using C# | | |
| Lecturer: | Dr B Wang | | |

**COURSEWORK DETAILS:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Assessment Number: | 1 | of | | | 1 | | | |
| Title of Assessment: | Student Administration Design | | | | | | | |
| Format: | Report | program | | | | |  | |
| Method of Working: | Individual | | | | | | | |
| Workload Guidance: | Typically, you should expect to spend between | 30 | | and | | 40 | | hours on this assessment |
| Length of Submission: | This assessment should be **no** more than:  *(over length submissions* ***will be*** *penalised as per University policy)* | | Report, 4 pages in total,  Word document, single column, text in font size 11. No images or screen shots. Small code fragments may appear in Consolas font 9 or above. | | | | | |

**PUBLICATION:**

|  |  |
| --- | --- |
| Date of issue: | 21st Oct 2019 (5th teaching wk) |

**SUBMISSION:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ONE copy of this assessment should be handed in via: | Canvas | | If Other  (state method) |  |
| Time and date for submission: | **Time** | 2pm | **Date** | 3rd Dec 2019 |
| If **multiple hand–ins** please provide details*:* | Demonstration in time-tabled lab in Week 15 (12th teaching wk) | | | |
| Will submission be scanned via TurnitinUK? | No | If submission is via TurnitinUK, these should be one of the allowed types e.g. Word, RT, PDF, PPT, XLS etc.  Specify any particular requirements in the subumission details  Students MUST NOT submit ZIP or other archive formats. Students are reminded they can **ONLY** submit **ONE** file and must ensure they upload the correct file. | | |
|  | | | | |

The assessment must be submitted **no later** than the time and date shown above, unless an extension has been authorised on a *Request for an Extension for an Assessment* form:

search ‘student forms’ on <https://share.hull.ac.uk>.

Canvas allows multiple submissions: only the **last** assessment submitted will be marked and if submitted after the coursework deadline late penalties will be applied.

**MARKING:**

|  |  |
| --- | --- |
| Marking will be by: | Student name |

**ASSESSMENT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| The assessment is marked out of: | 100 | and is worth | 100 | % of the module marks |
| **N.B** If multiple hand-ins please indicate the marks and % apportioned to each stage above (i.e. Stage 1 – 50, Stage 2 – 50). It is these marks that will be presented to the exam board. | | | | |

**ASSESSMENT STRATEGY AND LEARNING OUTCOMES:**

The overall assessment strategy is designed to evaluate the student’s achievement of the module learning outcomes, and is subdivided as follows:

|  |  |  |
| --- | --- | --- |
| LO | Learning Outcome | Method of Assessment  *{e.g. report, demo}* |
| ***1***  ***2***  ***3*** | Intellectual Skills: Show evidence of a systematic and comprehensive understanding of object-oriented principles and design that meets given requirements and standards.  Practical Subject-Specific Skills: Implement an efficient, robust C# application based on an object-orientated design. Adapt approaches including some at the forefront of the discipline.  Practical Subject Specific Skills: Use appropriate development tools and processes to create, debug, test and optimize an object-oriented implementation. | report, program  program  demo, program |

|  |  |  |
| --- | --- | --- |
| Assessment Criteria | Contributes to Learning Outcome | Mark |
| Quality of design described in report. Architecture should conform to 3-layer model. Design should implement the requirements and respect state invariants. Object types should model real-world entities from the application domain. User interface should allow efficient interaction for the various tasks.  Extent to which program executable fulfills requirements. Object types should model real-world from application domain according to object oriented design principles. Code should implement reported design, make appropriate use of .NET classes, be well-structured and adhere to good coding guidelines.  Demonstration not assessed except that students must demonstrate fluency with IDE otherwise mark capped at 50%. Program functionality and code is inspected during demo and informs program assessment. Demonstration also provides an opportunity for the student to gain feedback. | 1,2,  1,2  2, 3 | Approx 50.  See more detailed scheme in assignment details.  Approx 50.  See more detailed scheme in assignment details. |

**FEEDBACK**

|  |  |  |  |
| --- | --- | --- | --- |
| Feedback will be given via: | Comments at demo | Feedback will be given via: | Demonstration |
| Exemption  (staff to explain why) |  | | |
| Feedback will be provided no later than 4 ‘teaching weeks’ after the submission date. | | | |

This assessment is set in the context of the learning outcomes for the module and does not by itself constitute a definitive specification of the assessment. If you are in any doubt as to the relationship between what you have been asked to do and the module content you should take this matter up with the member of staff who set the assessment as soon as possible.

You are advised to read the **NOTES** regarding late penalties, over-length assignments, unfair means and quality assurance in your student handbook, which is available on Canvas.

In particular, please be aware that:

* Up to and including 24 hours after the deadline, a penalty of 10%
* More than 24 hours and up to and including 7 days after the deadline; either a penalty of 10% or the mark awarded is reduced to the pass mark, **whichever results in the lower mark**
* More than 7 days after the deadline, a mark of zero is awarded.
* The overlength penalty applies to your written report (which includes bullet points, and lists of text. It does not include contents page, graphs, data tables and appendices). 10-20% over the word count incurs a penalty of 10%. Your mark will be awarded zero if you exceed the word count by more than 20%.

Please be reminded that you are responsible for reading the University Code of Practice on Academic Misconduct through the Assessment section of the Quality Handbook (via the SharePoint site). This govern all forms of illegitimate academic conduct which may be described as cheating, including plagiarism. The term ‘academic misconduct’ is used in the regulations to indicate that a very wide range of behaviour is punishable.

In case of any subsequent dispute, query, or appeal regarding your coursework, you are reminded that it is your responsibility to produce the assignment in question.

**Description of assessment task**

**Scenario**

A student administration system is required for use by a departmental administrator. A student is registered on a degree programme and becomes a member of a particular student cohort. A degree programme is either a one year programme or a two year programme. A one year programme contains six modules and a two year programme contains twelve modules. Each module contains one or more assessments. A student may gain a mark for each assessment of a module to provide an overall module mark and then a programme mark. The system should allow the operations listed below. Data must persist between invocations of the system.

**Details**

A student is registered on a degree programme and becomes a member of a particular student cohort. A student cohort is the set of students registered for a given degree programme in a given academic session (academic year). A student has an ID and a name, the ID begins with the year of the cohort followed by unique 6 digit string. A degree programme has a 6 digit programme identifier and a programme title. A degree programme is either a one, two or three year programme. A one year programme contains six modules, a two year programme therefore contains twelve modules.

A module has a 5-digit module code and a module title. Some of the modules of a programme are determined by that program and some are chosen as options by the student registered on that programme. No two degree programs may have the same set of modules.

Each module contains one or more assessments. Each assessment of a module has a maximum mark which is less than or equal to 100. The sum of the maximum marks for the assessments of any module is 100. An assessment may be shared across more than one module. A student may gain a mark for each assessment of a module, the sum of which is the overall module mark. The module result is Pass if the module mark is greater than or equal to 50. If the module mark is less than 50 but greater than or equal to 45 then the module result is PassCompensation. Otherwise the module result is Fail. A module result is undefined if there is no mark for any assessment in that module. The average mark across all the modules in a programme, as an integer percentage, is the programme mark. The programme mark is undefined if any module mark is undefined. The programme result is Pass if the programme mark is greater than or equal to 50, Distinction if the programme mark is greater than or equal to 70 and Fail if less than 50.

Note that, like real world requirements, the above requirements can be expected to be reasonably accurate, reasonably complete and reasonably consistent but minor problems may be present. Dealing with such problems is part of the assessment. The system must be implemented (using C# and Visual Studio) and execute on a laboratory machine.

**Application design**

The application should consist of 3 layers, 1 – business logic, 2 – data storage, 3 - user interface. There should be a well-defined separation between the layers so that, for example, should be no user interface code or data storage code in the business logic. The business logic should have an API and the user-interface should access the business logic only through this API.

**Assessment**

It is important to appreciate that the evidence for good design must be present both in the report and in the program. The time allowed for this assignment does not permit a complete implementation and nor is a complete implementation required for full marks. Students should concentrate on demonstrating a good design. Elements of the implementation may be mocked up providing the design is clear. You are required to demonstrate your program in the lab using a laboratory machine and also to answer questions about your program code.

Note that, like real world requirements, the above requirements can be expected to be reasonably accurate, reasonably complete and reasonably consistent but minor problems may be present. Solving such problems is part of the stage. Again, like real world requirements, clarification or minor modifications to the above requirements may be issued, via University email, after the publication of this document.

**Mark Scheme**

Report (50 marks) consisting of:

Logical description of application entities or objects and relationships between objects, business logic (approx. 20).

Statement of the state invariants (approx. 5)

Data storage, data design, translation from to and from data storage to business logic objects, user interaction design (approx. 25)

Program (50 marks) consisting of:

Extent to which program design and functionality captures requirements. (30 marks).

Object types should model real-world from application domain according to object oriented design principles. Code should implement design described in the associated report, make appropriate use of .NET classes, be well structured and adhere to good coding guidelines. (20 marks)

Demonstration not assessed except that students must demonstrate fluency with IDE otherwise mark capped at 50%. Program functionality and code is inspected during demo and informs program assessment.

Demonstration also provides an opportunity for the student to gain feedback.

**Submission**

1. Place the Word report in the Visual Studio solution folder.
2. Clean (Build menu >Clean) the solution and then Zip the solution (use 7-Zip) into a single compressed file.
3. Submit the compressed file as a single attachment to Canvas.
4. No resubmission allowed so please make sure the correct file is submitted.

Second marking by Umar Manzoor

**PLEASE publish 1 PDF document for each assessment (omitting reserved comments) on the module canvas site.**

**Include a copy of the FULL description and the completed FA sheet in a hidden (staff only) Examinations folder on the module Canvas site for the External Examiners. This should include guidance – e.g. a readme file - for external examiners on how to locate and review the relevent marks, feedback etc.**