##### 

### UNIVERSITYOF HERTFORDSHIRE

**School of Engineering and Computer Science**

|  |  |
| --- | --- |
| COURSEWORK ASSIGNMENT | |
| **Module Title:** Programming | **Module Code:** 4COM1037 |
| **Assignment Title:** Supermarket Cashier System in C++ | **Individual Assignment** |
| **Tutor**: Dr Barry Ip | **Internal Moderator**: Dr Peter Lane |

|  |  |
| --- | --- |
| Student ID Number **ONLY**: | Year Code: |
|  |  |

|  |  |
| --- | --- |
| Marks Awarded %: | Marks Awarded after Lateness Penalty applied %: |
| Penalties for Late Submissions   * Late submission of any item of coursework for each day or part thereof (or for hard copy submission only, working day or part thereof) for up to five days after the published deadline, coursework relating to modules at Levels 0, 4, 5, 6 submitted late (including deferred coursework, but with the exception of referred coursework), will have the numeric grade reduced by 10 grade points until or unless the numeric grade reaches or is 40. Where the numeric grade awarded for the assessment is less than 40, no lateness penalty will be applied. * Late submission of referred coursework will automatically be awarded a grade of zero (0). * Coursework (including deferred coursework) submitted later than five days (five working days in the case of hard copy submission) after the published deadline will be awarded a grade of zero (0). * Where genuine serious adverse circumstances apply, you may apply for an extension to the hand-in date, provided the extension is requested a reasonable period in advance of the deadline. | |
| Please refer to your student handbook for details about the grading schemes used by the School when assessing your work. Guidance on assessment will also be given in the Module Guide. | |
| Guidance on avoiding academic assessment offences such as plagiarism and collusion is given at this URL: <http://www.studynet.herts.ac.uk/ptl/common/LIS.nsf/lis/citing_menu> | |

##### School of Engineering & Computer of Science

|  |  |  |
| --- | --- | --- |
| **ASSIGNMENT BRIEF**  ***Students, you should delete this section before submitting your work.*** | | |
| **This Assignment assesses the following module Learning Outcomes:**   * LO 1: Have knowledge and understanding of sufficient features of a high level programming language to develop solutions to simple programming problems; * LO 2: Have knowledge and understanding of the concepts of data declaration and operations, control flow (sequence, selection, iteration, subroutine call) and modularisation; * LO 3: Have knowledge and understanding of the terminology used in describing programs and programming. * LO 4: Design and implement solutions to simple programming problems in a given programming language; * LO 5: Execute, test and de-bug programs; * LO 6: Document programs to an agreed standard. | | |
| **Assignment Brief:**  **Supermarket Cashier System in C++**  **Requirements:**  Your task is to design a cashier system for a student supermarket. The price list of products in the store is shown below:   |  |  |  | | --- | --- | --- | | **Item** | **Barcode** | **Price** | | Milk | 0120001 | 10.50 | | Bread | 0120002 | 5.50 | | Chocolate | 0120003 | 8.00 | | Towel | 0120004 | 12.10 | | Toothpaste | 0120005 | 6.75 | | Soap | 0120006 | 5.20 | | Pen | 0120007 | 2.00 | | Biscuits | 0120008 | 4.45 | | Lamp | 0120009 | 20.50 | | Battery | 0120010 | 10.00 |   Upon starting the program, display a welcome screen. Here is a sample output of what it can look it:    The system should then prompt the user to enter barcodes, in the form of one entry per line. For each barcode entry, the system will search the database and locate the item’s name and price. If no match can be found, the program should display an error message. The barcode entry process should repeat until the user enters ‘F’ to finish, and then the total price should be calculated and displayed. Sample output may be:    The system will then ask for the required payment and calculate any resulting change. If an insufficient amount of payment has been received, the system should prompt for more payment. Sample output:    Upon completion, prompt for the next customer and repeat the same process as above. Sample output:    **Demo Video:**  A showcase demonstration video can be found on Canvas, which can be used as guidance for the development in addition to the design strategy outlined below.  **Design Strategy:**  There may be many approaches for implementing this program. Here is one possible pathway for the development of essential components – you may find it useful as a way to guide the development of core elements:   1. Write a simple main file. Display the welcome information in a user-friendly format.   (Refer to Unit 1 and 2 for basic C++ program structure, and refer to Unit 3 for formatted output.)   1. Store the information of the price list as arrays.   (Refer to Unit 8 for the use of arrays.)   1. Use loop to repeatedly ask the user to input a barcode.   (Refer to Unit 6 for the use of loops, and refer to Unit 3 for user input.)   1. Search the array based on the user input and give error information if a match is not found.   (Refer to Unit 8 for the use of arrays.)   1. Calculate the total price and ask for cash. Ask for more cash if necessary. Give change if applicable.   (Refer to Unit 4 for arithmetic operations, and refer to Unit 3 for formatted output.)   1. Add another loop to the program so the above process can be repeated for the next customer.   (Refer to Unit 6 for the use of loops.)   1. Add error tolerance to the system. For example, if the user inputs either ‘Y’ or ‘y’, it should be accepted as a “yes”, and an error message should be presented for the wrong input.   (Refer to Unit 5 for comparison statements.)   1. Improve the organisation of the program by separating certain code into suitable functions.   (Refer to Unit 7 for the use of functions.)   1. Enhance the documentation of the program and justification of key decisions.   (Refer to Unit 2 for the techniques of adding comments to the code.)  **Stretch Abilities:**  If you wish to extend this project further to gain extra credit, you may consider adding additional features: For example, a receipt file can be created for each transaction containing all the information of purchased items as well as for the total price. The receipt can be printed on the screen at the end of the transaction (refer to Unit 9 for file operations). The demo video that showcases this feature can be viewed via the assignment page on Canvas. | | |
| **Submission Requirements:**  **This assignment is to be submitted and marked anonymously. Students should ONLY use their student ID number to identify themselves on their work. Work submitted via Canvas for anonymous marking will automatically have an anonymity number allocated to it.**  **All submissions should be made via Canvas, including:**   * One zipped file for all your program files (the expectation is that you will likely upload a single cpp program file). Ensure all submitted files are present and operational – if necessary, submit a readme file to explain any nuanced aspects of the submission. Leave plenty of time before the deadline to ensure they are uploaded correctly.   **Please Note:** Work that is submitted to Canvas is subject strictly to the School policy on late submission – *even work submitted a second late will be subjected to the lateness penalty*, of which tutors have no control. You are strongly advised to submit your work one hour before the submission deadline, to give time to resolve difficulties. | | |
| This assignment is worth 40**%** of the overall assessment for this module.  **Marks awarded for:**   |  |  | | --- | --- | | **Components** | **Marks** | | Programming Technique   |  |  | | --- | --- | | *Appropriate data types and declarations* | *10%* | | *Appropriate program control techniques* | *10%* | | *Modular programming* | *10%* | | *Attempt of stretch features, if any* | *10%* | | *Appropriate comments throughout the program* | *10%* | | 50% | | Program Execution   |  |  | | --- | --- | | *Effective input and output* | *5%* | | *Fluent iterations* | *5%* | | *Error-checking and tolerance for user input* | *10%* | | *Accurate calculation of results* | *5%* | | *Suitably formatted output* | *10%* | | *Successful execution of stretch features, if any* | *10%* | | *User-friendly interface and smooth overall operation* | *5%* | | 50% | | **Total** | **100%** |   A note to students:   1. For undergraduate modules, a score above 40% represent a pass performance at honours level. 2. For postgraduate modules, a score of 50% or above represents a pass mark. 3. Modules may have several components of assessment and may require a pass in all elements. For further details, please consult the relevant Module Guide or ask the Module Leader. | | |
| Typical (hours) required by the student(s) to complete the assignment:  **70** hours | | |
| **Date Work handed out:**  Week commencing 4th November 2019 | **Date Work to be handed in:**  12pm **GMT**, Thursday 19th December 2019 | **Target Date for the return of the marked assignment:**  Week commencing 20th January 2020 |
| **Type of Feedback to be given for this assignment:**  Each individual student will receive written feedback as well as marks awarded based on the criteria set in the next section. | | |

**Grading Criteria**

|  |  |  |
| --- | --- | --- |
| **Marks** | **Programming Technique (50%)** | **Program Execution (50%)** |
| 90 – 100 (Outstanding) | Solutions are novel & fully address task within context. Profound depth of engagement and incisive ideas development. Decision making is in perceptive; techniques are used in an outstanding manner. | Outstanding use of appropriate technology as applied to the problem domain. Consistently accurate and outstanding application of skills and techniques. Components are fully functional as required and evidence of stepping beyond expectations using sophisticated solutions. |
| 80 – 89 (Excellent) | Solutions are innovative and partially address the task within its context. Considerable depth of engagement and successful ideas development. Clear rationale for decisions; techniques are used in an excellent manner. | Excellent use of appropriate technology as applied to the problem domain. Minor errors in technique and/or application with little or no impact on deliverables. Required components are functional as required, with a generally excellent and highly accurate application of skills and techniques. |
| 70 – 79 (Very good) | Solutions relate directly to task and may step beyond conventions. Very good engagement with subject material and processes, evaluation of alternatives, solutions come from process. Very good use of techniques to derive solutions. | Very good use of appropriate technology as applied to the problem domain. Understanding is demonstrated. Small errors in technique and/or application with little impact on deliverables. Most required components are functional, with a high level and accurate application of skills and techniques. |
| 60 – 69  (Good) | Solutions are appropriate to task, work well within conventions. Good use of subject material and processes. Experimentation is in evidence to support implementation within conventions. Good use of techniques to derive solutions. | Good use of appropriate technology as applied to the problem domain. Good and reasonably accurate application of skills and techniques demonstrated. Some errors in technique and/or application with minor impact on deliverables; required components may not be fully functional but may be easily resolved. |
| 50 – 59  (Clear pass) | Solutions are appropriate to task and work adequately within conventions. Appropriate use of subject material and processes. Experimentation is evident but may be limited in scope. Appropriate use of techniques to derive solutions but may be expanded for depth. | Appropriate use of appropriate technology as applied to the problem domain. Reasonably accurate application of skills and techniques demonstrated. Some errors in technique and/or application with some direct impact on deliverables; required components may not be fully functional but may be resolved through additional effort. |
| 40 – 49 (Marginal pass) | Solutions limited to task and address conventions. Solutions found or adopted but completion is rushed. Some experiments but limited alternatives. Techniques are applied to derive solutions but steps are missed. | Satisfactory use of appropriate technology as applied to the problem domain. Satisfactory application of skills and techniques demonstrated but with minor inaccuracies. Errors in technique and/or application with some impact on deliverables; required components may suffer from notable errors that affect overall functionality, but which can be rectified by minor improvement. |
| 30 – 39 (Marginal fail) | Solutions frame task inappropriately and do not address conventions. Basic use of strategies, few alternatives, limited evaluation with limited experimentation. Limited use of techniques to derive solutions. | Limited use of appropriate technology as applied to the problem domain. Limited application of skills and techniques demonstrated. Many errors in technique and/or application with high impact on deliverables; required components may suffer from critical errors, and may require significant improvement. |
| 20 – 29  (Clear fail) | Lacking in appropriate solutions with very limited use of strategies, no evaluation and little evidence of ideas development. Little use of techniques to derive solutions. Certain elements of academic misconduct. | Very little use of appropriate technology as applied to the problem domain. Very little skill and application of techniques demonstrated. High number of errors with very high impact on deliverables; required components may not function in any discernable way and/or are entirely missing. Certain elements of academic misconduct. |
| 0 – 19  (Fail – nothing of merit) | No or completely inappropriate solution. No use of strategies, no planning and no experimentation. No application of techniques to derive solutions. Academic misconduct. | No use of appropriate technology as applied to the problem domain. No skill and application of technique demonstrated. Very high number of errors in deliverable or no deliverable submitted. Required components are under developed, missing, or inappropriate. Academic misconduct. |