Year 11 Assessment Task Notification

Course: Software Design and Development \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***Due on 28/8/2020**

Mini Major Project 2020

**Date Due:**

1st individual consultation date:

2nd individual consultation date:

3rd individual consultation date:

**Weighting:** 30%

**Task Type:** Software Development Project

**Outcomes:**

P1.2 describes and uses appropriate data types

P4.2 investigates a structured approach in the design and implementation of a software solution

P5.1 uses and justifies the need for appropriate project management techniques

P5.2 uses and develops documentation to communicate software solutions to others

P6.2 communicates with appropriate personnel throughout the software development process

P6.3 designs and constructs software solutions with appropriate interfaces.

|  |
| --- |
| **Task Outline:**  **Software Development Project**  You are to develop a piece of software to meet a specified need. You will have a scheduled meeting with your teacher to discuss your project ideas and confirm your final choice. You will also be required to develop complete supporting documentation, as outlined in this assessment booklet.  *Scenario:* You need to develop a memory game. Players will test their memory by attempting to correctly type a set of 6 random characters (that is, not real words) that are displayed to them for only a brief amount of time. If the player successfully types the same 6 characters, display an appropriate message indicating to the player they were correct. If the player does not type the matching 6 characters, display an appropriate message. After showing the player their success/failure message, clear the screen ready for another game.  Players can make the game harder by reducing the number of seconds the characters are displayed to them, or easier by increasing the number of seconds the characters are displayed. The default for the game is displaying the set of characters for 10 seconds. Minimum display time is 2 seconds, maximum is 20 seconds.  Players will need to know how to install and operate your game on their computer. Players need access to a help screen that explains how to play your game. Developers will need to know how your game works, so they can make any necessary fixes or enhancements.    *Program Features:*   * The 6 random characters can be alphabetic, numeric or special characters. An example of six random characters could be “8e3m&7” * You can come up with any method you see fit to generate the 6 random characters * You must use at least one sub-program and at least one function in your program. * Your game should be visually appealing * You should aim to design a game that will be intuitive for the user * Your code must be documented internally * You must include a help screen for the user explaining how to play the game   **Project Requirements**   1. **Design Brief** – listing the aims and objectives of the project. 2. **Detailed Design** – including: 3. Sketches of all the screens that will appear in the program 4. Program Specifications – what specifically will the program do, including any limitations of the program 5. The following diagrams: IPO, dataflow, context diagram, system flowchart and structure chart 6. **Algorithm Design**   A full set of algorithms for the program are to be completed as pseudocode or flowcharts. These are to be produced for each major module of your program. They are to show program logic and data handling.   1. **Program Testing**   You are to prepare a suitable plan demonstrating how your program will be tested for logic and run-time errors. This plan should include (where appropriate) suitable test data that will be used to test the program.  Should you have a program where traditional test data is not suitable you will need to detail how your program can be tested to ensure that “all paths” through the program operate correctly.   1. **Coded Program (as both source code and object code).**   You will be required to submit all files used in your project as Python source code. You will also supply a compiled version of your program (this will be the last stage and completed with your teacher). You will be saving your project on an external hard drive dedicated to your Project and a digital copy to our Google Classroom. Make sure you regularly back up your project to the school drive and to your google drive!  The following features should be present in your program:   * Suitable control structures, as described in class and your texts. For example: FOR and WHILE loops, IF statements, Case statements, use of arrays, use of records, use of files. * The code should be clear, concise, consistent and appropriate. * The program must make use of suitable data structures and identifiers e.g. btnQuit. * All input must be consistent and have some error trapping applied to stop incorrect data being entered (in the form of data validation) * Output must be consistent and relevant. * The user interface must be consistent with the guidelines discussed in class and your texts. * The program should include commenting to all the code explaining the purpose and function of sections of code. * The program should provide on-screen prompts and help for users.   **Note:** The majority of program coding MUST take place at school.   1. **Documentation**   You are to produce the following documentation:   1. A user manual which includes: 2. An installation guide 3. Instructions on how to use the program 4. A troubleshooting guide (solving simple problems the user may encounter) 5. A technical manual that includes: 6. A data dictionary for all variables used in your program 7. A listing of all procedure names with a description of what they are designed to do 8. A copy of all program code copy and pasted from Python (this is to safeguard against any potential issues) |

**Submission Requirements:** You will be saving your project on an external hard drive dedicated to your Project and a digital copy to our Google Classroom. All work should be submitted on via google classroom, with files clearly organised in labelled folders.

Written material should be in the following sections:

* Design Brief
* Detailed Design
* Algorithms
* User Manual
* Technical Manual

**Policy Requirements**: *If an assessment task is submitted late or a student does not attend an assessment task, and there is no successful illness / misadventure appeal, students will receive a zero for that task. Students should still submit the task to gain necessary feedback. All work presented in assessment tasks and examinations (including submitted works and practical examinations), must be your own. Forms of malpractice or misconduct in an assessment task by a student may result in a zero for that task.*

**Marking Criteria:** Total marks: 100

**Assessment of the Project – Marking Criteria**

|  |  |
| --- | --- |
| **Design Brief (10)** | |
| **4** | * A comprehensive design brief that explains all aims and objectives of the   program in full detail. |
| **3** | * A thorough design brief that outlines all aims and objectives of the   program in detail. |
| **2** | * A satisfactory design brief that outlines all aims and objectives of the   program in detail. |
| **1** | * A very basic design brief that lists in general terms the aims and objectives   of the program. |
| **0** | * Design brief is not submitted. |

|  |  |
| --- | --- |
| **Detailed Design (20%)** | |
| **20 - 24** | * Comprehensive sketches showings the screen design for all   components/modules of the program.   * Sketches are precise, professional and neat. * Screen design follows sound design principles with appropriate use of   fonts, colours and placement of items in all sections of the program.   * Complete and extensive specifications of what the program will do   including all limitations is provided |
| **16 - 19** | * Sketches showing the screen design for all components/modules of the   program have been provided.   * Screen design follows sound design principles with appropriate use of   fonts, colours and placement of items in most sections of the program.   * Detail specifications of what the program will do including mention of   major limitations is provided |
| **11 - 15** | * Some sketches showings the screen design for some   components/modules of the program.   * Screen design follows sound design principles with appropriate use of   fonts, colours and placement of items in some sections of the program.   * Generalized specifications of the main aspect of the program are   provided with some reference to program limitations. |
| **7 - 9** | * Sketches showings the screen design for some components/modules of   the program in limited details   * Screen design is poor or inconsistent. Screen design does not follow sound   design principles and inhibits users from using program to its maximum  potential.   * Basic details of program specifications and some limitations are provided |
| **1 - 6** | * Very basic screen design sketches or screen captures only. * Very poor or inconsistent screen design which makes the program hard to   use.   * Limited details of what the program will do with no mention of any   Limitations |
| **Algorithms and Test Data (20)** | |
| **18 - 22** | * Detailed algorithms for all significant sections of the project are submitted.   These follow correct structure and are free of logic errors.   * Comprehensive test data and testing methods are provided that will test   all aspects of program operation. |
| **14 - 18** | * Detailed algorithms for all significant sections of the project are submitted.   These follow correct structure but may contain logic errors.   * Comprehensive test data and testing methods are provided that will test   most aspects of program operation, some aspects do not get tested. The  number of aspects not tested may vary. |
| **9 - 13** | * Basic algorithms for all significant sections of the project are submitted.   These follow correct structure and are free of logic errors.   * Basic test data and testing methods are provided that will test major   aspects of program operation. Some aspects of program operation are  omitted from testing. |
| **5 - 8** | * Basic algorithms for some significant sections of the project are submitted.   These follow correct structure but may contain logic errors.   * Limited test data, but appropriate methods used. The testing method   provided does not fully test all aspects of the program. |
| **1 - 4** | * Basic algorithms for some sections of the project are submitted. These   algorithms do not follow correct structure or contain logic errors.   * Limited or inappropriate test data or testing methods. |

|  |  |
| --- | --- |
| **Documentation (20)** | |
| **17 - 20** | * Easy to follow installation guide has been produced with step by step   instructions that include visual screen shots.   * A comprehensive and extensive user manual had been produced which   is easy to follow and contains detailed information on how the program  operates.   * An exhaustive trouble shooting guide has been produced * A data dictionary for variable with variable name, data type and   description has been prepared. Data types are appropriate for the  information being stored.   * A listing of all procedure (module) names is provided including detailed   explanation of their role.   * Program listing has been provided in the specified format. |
| **13 - 16** | * Easy to follow installation guide has been produced with step by step   instructions. May or may not include screen shots   * A comprehensive user manual had been produced which is easy to   follow and contains detailed information on how the program operates.   * A trouble shooting guide has been produced which deals with major   problems users are likely to encounter.   * A data dictionary for variable with variable name, data type and   description has been prepared. Data types are generally appropriate for  the information being stored.   * A listing of all procedure (module) names is provided including   explanation of their role.   * Program listing has been provided in the specified format. |
| **9 - 12** | * A basic installation guide has been produced with step by step   instructions. May or may not include screen shots   * A significant user manual had been produced which is easy to follow and   contains information on how major program components operates.   * An trouble shooting guide has been produced which deals with some of   the problems users are likely to encounter.   * A basic data dictionary for variable with variable name, data type and   description has been prepared. Data types may not be appropriate.   * A listing of all procedure (module) names is provided including basic   explanation of their role.   * Program listing has been provided in the specified format. |
| **5 - 8** | * A basic user manual has been produced. It has some worthwhile   components but is very limited in detail.   * Simple instructions on how to install the program are provided in point   form.   * An attempt to produce a troubleshooting guide is evident. This will deal   with at least two possible problems in a very simple manner.   * Program listing for major modules has been provided. |
| **1 - 4** | * A very limited user manual has been created * User manual omits to include installation guide * Trouble shooting guide has been omitted or is very basic * A list of variables names is provided but not in the format specified * No program listing or only a partial program listing. |

|  |  |
| --- | --- |
| **Coded Program (30)** | |
| **30-24** | * Coding methods used are appropriate and complete the required task in   an efficient manner. The coded program utilises iteration or loops successfully (e.g. FOR or WHILE) and utilises IF statements as well as case statements.   * Appropriate naming conventions for variables, modules, and component   elements have been used at all times   * The use interface is consistent, user-friendly and allows the user to use the   program to maximum potential.   * Appropriate data structures are used consistently (e.g. records, files and arrays). * Appropriate error trapping methods are used to prevent run-time errors. * The program is free of logic errors. * The solution/idea chosen involves high complexity and contains a comprehensive range of functionality. |
| **23-17** | * Coding methods used are appropriate and complete the required task but may not be in the most efficient manner. The coded program utilises some iteration or loops successfully (e.g. FOR or WHILE) and utilises IF statements with little to no use of case statements. * Appropriate naming conventions for variables, modules, and component   elements have been used in most cases   * The use interface is consistent and user-friendly but could have been   made more efficiently with further refinement.   * Appropriate data structures are used consistently. * Appropriate error trapping methods are used to prevent run-time errors. * The program is free of logic errors. * The solution/idea chosen involves complexity and contains a thorough range of functionality. |
| **16-11** | * Coding methods generally complete the task as required, but are not the   most efficient methods available.   * The coded program does not utilise iteration or loops successfully (e.g. FOR or WHILE) and utilises a basic level of IF statements. Case statements are not used. * There may be some inconsistency in code and very minor parts that do not function. * There is some attempt to use appropriate naming conventions for   variables, modules, and component elements.   * The use interface has some appropriate aspects to its design. * Appropriate data structures are used for most aspects of the program. * Minimal error trapping methods are used to prevent run-time errors. * The program is generally free of logic errors. Any errors present are minor   in nature.   * The solution/idea chosen involves some complexity and contains a basic range of functionality. |
| **10-6** | * Some of the coding works to complete some tasks in the program. There is   an attempt to use appropriate methods.   * Some code may not function as required or may not be an appropriate or efficient method used. * Limited use of naming conventions for variables, modules, and   component elements.   * The use interface has some appropriate aspects to its design. * Attempts to use appropriate data structures in some parts of the program. * The program may contain run-time errors that could have been   prevented with appropriate error trapping.   * The program may contain logic errors in some sections. * The solution/idea chosen involves little complexity and contains a limited range of functionality. |
| **5-0** | * Limited sections of coding operate correctly but are not efficient in their   manner of operation.   * No use of naming conventions for variables, modules, and component   elements.   * The use interface has been poorly designed making the program hard to   use.   * Very limited use of appropriate data structures. * The program may contain run-time errors that could have been   prevented with appropriate error trapping.   * The program contains logic errors in significant sections. * The solution/idea chosen is simple and contains little functionality. |