**UNIVERSITY CENTRE SOMERSET**

**Computing and Digital Technologies**  
Assignment Coversheet and Grading Criteria  
2020 / 2021

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Qualification** | | | **Module Code and Title** | |
| BSc (Hons) Computing and Digital Technologies  FdSc Computing and Digital Technologies | | | SCDT41 Programming and Software Fundamentals | |
| **Student Name and Number** | | | **Module Tutor** | |
|  | | | James Shaun | |
| **Date Issued** | | **Submission Date** | | **Return Date** |
| 07/01/2021 09:00 | | 04/02/2021 15:00 | | 08/03/2021 12:00 |
|  | | |  | |
| **Assignment Number** | 1 of 2. This assignment is worth 50% of the overall module. | | | |
| **Assignment Title** | Coursework One – Programming Portfolio | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Module Learning Outcomes**  *To achieve the outcomes the evidence must show that the learner is able to:* | |  | **Task no.** |
| C1 | Produce programming solutions to solve problems, using high quality code and industry standard practices. |  | 1 |
| D1 | Apply logical thinking and a creative approach to problem solving. |  | 1 |

|  |  |
| --- | --- |
| **Word Count of Submission** | Equivalent to 2000 |
| **Student Declaration** | |
| Through submitting this assignment through Turnitin you agree that the work was prepared entirely by yourself in accordance with Open University’s Prevention of Academic Dishonesty Code of Practice. | |

|  |
| --- |
| **Assignment Feedback** |
| All feedback for this assignment will be provided through Turnitin in accordance with the grading criteria below on the given return date. |

|  |  |
| --- | --- |
| **Assignment Task(s)** | |
| **Task no.** | **Task details** |
| 1 | In this assignment, you are required to assemble a portfolio of completed programming solutions to given problems. For each solution you should aim to demonstrate two methods of solving the problem and include an analysis of **which solution is the most efficient and/or effective at solving the given problem and why.**  For each problem evidenced in your portfolio, you should:   * Decompose each problem into an algorithm using numbered step-by-step instructions * Provide two possible solutions for each problem, with one avoiding the use of predefined functions. * Compare between both solutions evaluating the efficiency, performance and resource usage. * Demonstrate high quality programming standards, annotations and structure.  1. You are provided with an integer array containing unique values, and a single integer value, *T*. What maximum number of different integers that can be combined without exceeding the value of *T?*    1. *A* = [3, 5, 2, 1, 7, 4]    2. *T* = 14 2. Write a program that takes string *P* as a password, this password must be a length of at least 7 characters, contain an uppercase letter, lower case letter, a number, a special character and not have more than three of the same characters next to each other.    1. The program should take in an input as a string    2. It should return True or False to the console depending on if the password is valid. If false it should identify which requirement(s) it did not meet. 3. Write a program that receives an unsorted integer array of length *N,* which thensorts in order of largest to smallest, and removes repeat values.    1. Sample array: [9, 88, 1, 9, 88, 87, 35, 12, 50, 23, 12, 1, 4, 9] 4. You are given a string *S* of length *N*. Each character of the string is either an ‘X’ or ‘Y’. Find the substring which has the largest number of consecutive characters of the same type. Output the highest number of consecutive characters and the substring.    1. Example *S*: “XXXYXYXXYYYXYYYYXX”    2. Output: [4, “YYYY”] 5. You are provided the 3 character string, *S,* which contains the value “ABC”. Create a solution which rearranges all possible character combinations of *S.*   All evidence generated must be presented in a single portfolio, ensuring use of adequate report presentation such as headings, contents pages, page numbers and formatted syntax. |

|  |
| --- |
| **Sources of Information** |
| Clarke, N. (2017) *C#: A Detailed Approach to Practical Coding*, Create Space Independent  Greene, J. and Stellman, A. (2013) *Head First C#,* O’Reilly Media  Haunts, S. (2017) *A Gentle Introduction to Agile Software Development,* Stephen Haunts Ltd.  McGrath, M. (2017) *C# Programming in Easy Steps*, In Easy Steps Ltd. |

|  |
| --- |
| **Submission Requirements** |
| 1. Check the grading criteria below to ensure your assignment document meets the demands of the above task(s). 2. Include screenshots of each solution with 3. Name the assignment document and any relevant ZIP file using the following format:  SCGT42 \_CW1\_*StudentNumber\_FirstName\_LastName*.docx/zip  (replace the *placeholders* with your student number, first and last name respectively) 4. Go to the Turnitin and use the upload facility to submit your assignment and any required ZIP file to the relevant module. There is no need to submit this assignment brief. |

|  |  |
| --- | --- |
| **IMPORTANT INFORMATION** | |
| * Please stay within the limits of the word count stated at the top of assignment brief. Any additional content over the word count limit (plus or minus 10%) will be disregarded and not be assessed at all.  All work should be submitted online via Turnitin.Please ensure that you submit your assignment on the right submission slot for each module.It is your responsibility to check that you can access Turnitin and Blackboard properly. If your college student account is locked, please contact IT on 01823 366 354 or email them to ITSHelpdesk@btc.ac.uk and request to have your account unlocked, but please ensure you allow plenty of time to do this, do not leave everything until the last day of your deadline.If there are circumstances where you need to submit your assignment other than online, please discuss your needs with the module tutor and alternative arrangements could be made so that you can submit your coursework within the set deadline.Regulations allow you to submit coursework up to 6 working days late. A penalty of deducting 10% will be applied for each day an assignment is late, with a maximum penalty of deducting 60% from your final mark for the late assignment. Any assignment submitted later than 6 days with be awarded a mark of zero. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Numeric Grade | Descriptor  (Class Band) |  | Undergraduate Grading Criteria | | | | |
| Problem Solving and Approach to Solutions | | Use of Logic and Data Structures | Code Quality and Industry Practices | Justification and Evaluation of Solutions | Overall Structure and Presentation |
| 80-100 | Outstanding  (Upper Distinction) | Solutions demonstrate industry standard application of problem solving, with effective decomposition of problems to find efficient algorithms to achieve the goals of all solutions efficiently. Overall, algorithms are precise, effective and are performance driven, providing an outstanding demonstration of problem solving. | | Industry standard application of logic and data structures is outstanding, with all solutions demonstrating the expected outcomes efficiently. Implementation of data structures demonstrates outstanding skills and knowledge of programming and software fundamentals. | Industry standard quality of code is consistently excellent and includes all elements of good coding practices such as being readable, efficient, well structured, reliable and could potentially be extendable. Evidence of expected industry practices such as, structure, organisation, naming conventions and informative comments in some solutions is present in all solutions. | Industry standard evidence of justification of methods or techniques, with excellent evaluations which consider the implemented methods for all solutions, providing outstanding justifications of one solution over another, which fully considers performance, readability, maintenance and efficiency. | Portfolio is of professional quality, with use of industry standard technical vocabulary which communicates solutions effectively. There is no spelling, grammatical or punctuation errors evident. Code is presented clearly in an easily readable format. |
| 70-79 | Excellent  (Lower Distinction) | Solutions demonstrate excellent application of problem solving, with effective decomposition of problems to find efficient algorithms to achieve the goals of all solutions efficiently. Overall, algorithms are precise and effective providing an excellent demonstration of problem solving. | | Use of logic and data structures is excellent, with all solutions demonstrating the expected outcomes. Implementation of data structures demonstrates excellent skills and knowledge of programming and software fundamentals. | Quality of code is generally excellent and includes most elements of good coding practices such as being readable, efficient, well structured, reliable and could potentially be extendable. Evidence of expected industry practices such as, structure, organisation, naming conventions and informative comments in some solutions is present in all solutions. | Excellent evidence of justification of methods or techniques, with excellent evaluations which consider the implemented methods for all solutions, providing excellent justification of one solution over another, which fully considers performance, readability, maintenance and efficiency. | Portfolio is of excellent quality, with frequent use of industry standard technical vocabulary which communicates solutions effectively. There are very minor spelling, grammatical or punctuation errors evident. Code is presented clearly in an easily readable format. |
| 60-69 | Very Good (Commendation) | Solutions demonstrate very good application of problem solving, with effective decomposition of problems to find efficient algorithms to achieve the goals of all solutions. Overall, algorithms are generally effective and provide very good demonstrations of problem solving. | | Use of logic and data structures is very good, with all solutions demonstrating the expected outcomes. Implementation of data structures demonstrates very good skills and knowledge of programming and software fundamentals. | Quality of code is generally very good and includes many elements of good coding practices such as being readable, efficient, well structured, reliable and could potentially be extendable. Evidence of expected industry practices such as, structure, organisation, naming conventions and informative comments in some solutions is present in all solutions. | Very good evidence of justification of methods or techniques, with very good evaluations which consider the implemented methods for all solutions, providing very good justification of one solution over another in terms of performance, readability, maintenance or efficiency. | Portfolio is of very good quality, with some use of industry standard technical vocabulary which communicates solutions well There are some spelling, grammatical or punctuation errors evident. Code is presented clearly in an easily readable format. |
| 50-59 | Good/Satisfactory  (Upper Pass) | Solutions demonstrate generally good application of problem solving, utilising an appropriate algorithm to solve given problems. Outcomes from solutions are mostly correct. | | Use of logic and data structures is generally good, with most solutions demonstrating the expected outcomes. Implementation of data structures demonstrates good skills and knowledge of programming and software fundamentals. | Quality of code is generally good and may include some elements of good coding practices such as being readable, efficient, well structured, reliable and could potentially be extendable. Evidence of expected industry practices such as naming conventions and informative comments in some solutions is present in all solutions. | Good evidence of justification of methods or techniques, with generally good evaluations considering the implemented methods for some solutions, providing good justification of one solution over another which could consider performance, readability, maintenance or efficiency. | Portfolio is of good quality, with some use of industry standard technical vocabulary which communicates solutions correctly. There may be frequent spelling, grammatical or punctuation errors evident. Code is presented appropriately in readable format. |
| 40-49 | Marginal Pass / Satisfactory (Lower Pass) | Solutions demonstrate satisfactory skills and application of problem solving. Some solutions demonstrate appropriate outcomes. | | Use of logic and data structures is generally satisfactory and demonstrates some appropriate application of skills and knowledge in software and programming fundamentals. Some solutions demonstrate appropriate outcomes | Quality of code is generally satisfactory and demonstrates some application of industry practices in programming and software fundamentals. Evidence of appropriate naming conventions and informative comments in some solutions. | Satisfactory evidence of justification of methods or techniques, with generally appropriate evaluations considering the implemented methods for some solutions. | Portfolio is of satisfactory quality, with general use of some industry standard technical vocabulary which communicates solutions. There many spelling, grammatical or punctuation errors evident. Code is presented but may not be easy to read or inspect. |
| 20-39 | Clear Fail  (Fail) | Problem solving is inadequate and does not appropriately produce an accurate outcome. | | Use of logic or data structures demonstrates a lack of skill and knowledge in software and programming fundamentals. | Quality of code is unsatisfactory, demonstrating a lack of skills in the application of standard industry practices. | Unsatisfactory evidence of the justification of methods or techniques, with no evaluation of chosen implementation of techniques for solutions. | Portfolio is incomplete, with many missing features or content, not meeting the requirements of tasks, no additions of code or evidence. |
|  | Nothing of Merit  (Fail) | Problem solving is not attempted and no evidence or nothing of merit has been submitted. | | Use of logic or data structures are incomplete or not present, and nothing of merit has been submitted. | Quality of code is poor and/or nothing of merit has been submitted. | Submission includes no evidence of evaluation or justification, nothing of merit has been submitted. | Portfolio is incomplete and/or has very poor structure. |