|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **WMG Undergraduate Mark Sheet WM392 Assignment 1 (2021/22)** |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |
| **Student ID** 1921983 **Mark Sheet for CW and weighting of CW 40%** | **Mark** | 64 |  | **Marker** | | Dr Young Saeng Park | | | | |  |  | | |  |

| **Apprenticeship Standard KSB** | **Intended Learning Outcome or Evaluative Criteria** | **Comments** | **100%** | **94%** | **88%** | **82%** | **78%** | **74%** | **68%** | **65%** | **62%** | **58%** | **55%** | **52%** | **48%** | **45%** | **42%** | **38%** | **32%** | **25%** | **12%** | **0%** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Exceptional 1st | | High 1st | Upper Mid 1st | Lower Mid 1st | Low 1st | High 2:1 | Mid 2:1 | Low 2:1 | High 2:2 | Mid 2:2 | Low 2:2 | High 3rd | Mid 3rd | Low 3rd | Fail | | Low Fail | | Fail |
| ***Standard ST0027***  K3, K9  S2  B1, B7, B14 | **LO1:**  Describe the mechanisms of operating system to handle processes, threads,  scheduling and communication. | This is Part1 of the assignment to understand and handle threads and their resources. **(40%)** | Exceptional work of the highest quality, demonstrating excellent knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  The excellent programs for the tasks related to threads in the Part 1 are provided, satisfying the teaks’ requirement and also including scalability and usability beyond the tasks’ requirement.  All the explanations are complete and sufficient with well representation and their results, including the scalability and usability. | | Very high-quality work demonstrating excellent knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  The programs for the tasks related to threads in the Part 1 are provided, satisfying the teaks’ requirement.  All the explanations are complete and sufficient with well representation and their results. | | | | High quality work demonstrating good knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  All the tasks related to threads in the Part 1 are complete but there are potential errors hidden in the programs for the tasks.  The explanations given for the tasks are provided but are not sufficient to demonstrate and explain the results. | | | Competent work, demonstrating reasonable knowledge and understanding, some analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  All the tasks related to threads in the Part 1 are complete but there are minor errors in the developing programs for the tasks.  Some explanations given could be improved and have missing parts. | | | Work of limited quality, demonstrating some relevant knowledge and understanding.  All the tasks related to threads in the Part 1 are not solved or there are some major errors in developing programs for the tasks, providing wrong results.  The explanations given are not complete and have missing parts. | | | Work does not meet standards required for the Level 6 of an Honours degree.    There is evidence of study and some basic understanding of relevant concepts and techniques, but subject to significant omissions and errors (**38**)  Work is significantly below the standard required for Level 6 of an Honours degree. Some evidence of study and some knowledge and evidence of under-standing but subject to very serious omissions and errors (**32**) | | Poor quality work well below the standards required for the Level 6 of an Honours degree. | | Work of no merit OR Absent, work not submitted, penalty in some misconduct cases |
| ***Standard ST0027***  K3, K7, K9  S2  B1, B7, B9, B14 | **LO2:**  Know the structure and organization of the file system and analyse the  components for concurrency management | This is Part 2 of the assignment to understand, design and develop file systems.  **(60%)** | Exceptional work of the highest quality, demonstrating excellent knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  The excellent programs for the tasks related to file systems in the Part 2 are provided, satisfying the teaks’ requirement and also including extensibility beyond the tasks’ requirement.  All the explanations are complete and sufficient to explain and demonstrate the results, based on the requirement. Also, it is described in detail how to include the extensibility. | | Very high-quality work demonstrating excellent knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  The programs for the tasks related to file systems in the Part 2 are provided, satisfying the teaks’ requirement.  All the explanations are complete and sufficient to explain and demonstrate the results, based on the requirement. | | | | High quality work demonstrating good knowledge and understanding, analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  All the tasks related to file systems in the Part 2 are complete but there are potential errors hidden in the programs for the tasks.  The explanations given for the tasks are provided but are not sufficient to demonstrate and explain the results. | | | Competent work, demonstrating reasonable knowledge and under-standing, some analysis, organisation, accuracy, relevance, presentation, and appropriate skills.  All the tasks for file systems in the Part 2 are complete but there are minor errors in the developing programs for the tasks.  Some explanations given could be improved and have missing parts. | | | Work of limited quality, demonstrating some relevant knowledge and understanding.  All the tasks related to file systems in the Part 2 are not solved or there are some major errors in developing programs for the tasks, providing wrong results.  The explanations given are not complete and have missing parts. | | | *Same as in the cell above* | | *Same as in the cell above*. | | *Same as in the cell above*. |

|  |
| --- |
| **What went well What could be improved and how *Marker additional comments***  [Part1-Task1] It should mention “race condition” for the problem of task 1 clearly rather than “it is only incremented by one of the threads”.  [Part1-Task2 & Part1-Task3] There is a compile error in the pthread\_create() function because you cannot pass (int)i as a parameter but pass (void \*) type as the parameter. Thus, you should pass the value like (void \*)&i.  Still there is another problem that if you pass &i, the thread\_part variable in the sum\_part() function might have a different value than you expected because we do not know which thread will start first and also the value of i variable keep changing.  Also, you should write the pthread\_create() and pthread\_join() function in a separated for-loop statement.  [Part1-Task3] The Task 3 description says that there are 5 baskets and 1 tank but you define BUCKET\_TOTAL as 6, which is one more basket. Also, there is no need to wait for 1 second after calling the pthread\_cond\_wait() function.  [Part2-Task1] According to the SFS constraints, the SFS data block size is 256 bytes and the maximum file size is 10000 bytes. It means that the number of necessary for 1000 bytes will be 39.0625. (40 data blocks required). But, you define the MAX\_FILE\_BLOCKS as 39, which is one less.  [Part2-Task2] In the SFS\_save() and SFS\_read(), you may also consider fread() and fwrite() to read the SFS\_DATA file than fprintf() and fscanf() in a case that performance is a high priority. So, you can read and write data from/to a file as the size of SFS structure easily.  In the SFS\_init(), “superBlock.inode[i] = aFile” (Line 27) cannot guarantee to populate inodes because “inode\_t aFile” will be defined locally and will be disappeared after calling the SFS\_init() function. So, you must use malloc() to allocate a heap memory but in this case, the superBlock.inode must be a pointer and change other codes totally to handle this pointer variable. This comment also applies to the SFS\_read() function for “superBlock.inode[file\_pos] = aFile” (Line 111).  [Part2-Task3] It is said that all the information for the SFS is stored in the one data file, called SFS\_DATA. So, the SFS\_DATA file should contain not only the file information but also data blocks. Also, it is said that the number of data blocks is 100 and each data block is only 256 bytes. Therefore, when creating a file, it is necessary to create the file while taking into account the limitation of the data space.  In the SFS\_create() function, if the filename is more than one character, e.g., ‘aaa.c’ and the file size is 100, the actual file size will be 300 based on your logic. So, it will be 3 times more than the file size.  It is good to see your own Mini Shell for testing your SFS library. But it might be better if the Mini Shell is more user-friendly. |
|  |
|
| **Student reflections and forward planning *Space for students to write reflections on their work and note action plans*** |
|
|