**ASSIGNMENT BRIEF**

Programme: Games Development

Module code/title: 2CB101

Assessment mode: Portfolio

Assessor(s): Andrew Guest

|  |  |
| --- | --- |
| **DEADLINE** | Noon Fri 16th April 2021 |
| **HOW SUBMITTED** | Electronic through replit |

|  |  |
| --- | --- |
| **Overview**  *Please ensure you read all of this document. Page 3 gives a breakdown of the marking for this module. Page 4 gives a detailed breakdown on how the university grades different concepts.*  **The document should be submitted through moodle. It should contain a link to your repl on the front page. Any repls that have been modified after the submission date will be treated as late submissions.** | |
| **Storing Data In A Binary Search Tree**  Fork a copy of the repl at <https://repl.it/@andyguest/cResitAssessment>  For this assessment you must write code to read the data from the arrays in data.h and store it in a **binary search tree** (https://ysjprog03.netlify.app/docs/cstruct-10/) indexed using the module number. Each node in the binary tree should use the **moduleNode** structure provided.  **Requirements**   1. You should implement an **insertNode** function that creates a new node, populates it with the correct data for a module and inserts it in to the binary search tree in the correct position. 2. You should implement a **displayTree** function that outputs a diagram of the tree. It should traverse the tree **In Order** and for each node output the address of the node and the module number in the node. It should also output the address and module number of the nodes to the left and the right of the current node (if they exist).   The output for each node should look something like  [addr] 2103 : Left [addr] 1101, Right [addr] 2107  Where addr is the address of the node.   1. You should implement a **displayModules** function that outputs the full details for each module, sorted by the module number. This data **must** come from the binary search tree. 2. You should implement a **displaySemester** function that takes an integer as a parameter and outputs the modules that run in the semester that matches the integer. This display should be sorted by module number. This data **must** come from the binary search tree. You should validate the parameter value before searching the tree. 3. You should implement a **searchModule** function that takes a string (char\*) as a parameter and searches for any module names that **contain** that string. For each module that matches that string the full details should be output, sorted by module number. e.g. “Comp” should match with Computer Fundamentals and Human Computer Interaction. If no modules match the string then an appropriate message should be displayed. This data **must** come from the binary search tree. 4. Write a document describing how your code works. You should pay special attention to what is stored where in memory and when. The lectures give examples of how this can be described. (1000 words maximum)   **Assessment Data**  This repl contains a file data.c containing four arrays of data.  int moduleNumber[18] = {2103, 3109, 1101, 2107, 1103, 3101, 1105, 2108, 1106, 2102, 3105, 3107, 3108, 1102, 1107, 2101, 2109, 3104};  char moduleName[18][30] = {"Networking", "Advanced Web Development", "Programming 01", "Mobile Programming", "Computer Fundamentals", "Major Project", "Programming 02", "Professional Project", "Software Engineering", "Databases", "The Internet Of Things", "Artificial Intelligence", "Cybercrime Security", "Mathematics",  "Technology In Context", "Programmming 03", "Philosophies of Technology", "Human Computer Interaction" };  int moduleYear[18] = {2,3,1,2,1,3,1,2,1,2,3,3,3,1,1,2,2,3};  int moduleSemester[18] = {1,2,1,2,1,1,2,2,2,1,1,2,2,1,2,1,2,1};  This data refers to 18 modules. Each module has a module number, title, degree year (1, 2 or 3) and semester (1 or 2). Each module is stored in the same index in each array. i.e. in index 0 is moduleNumber[0]=2103, moduleName[0]=Networking, moduleYear[0]=2, moduleSemester[0]=1. Index 0 gives all the details for module 2103 Networking, a second year module taught in semester 1.  **moduleNode**  The code also defines a structure called *moduleNode* as follows  typedef struct moduleNode {  int number;  char name[30];  int year;  int semester;  struct moduleNode \*left;  struct moduleNode \*right;  } moduleNode; | |
|  | |
| **FEEDBACK (how & when?)** | Feedback will be provided through Moodle by Friday 7th May 2021 |

**Marking Guide**

|  |  |  |
| --- | --- | --- |
| **Component** | **Overall** | **Content** |
| Data stored correctly in binary search tree (insertNode function) | 20% | 1. Appropriate functions for creating a binary search tree and inserting nodes correctly |
| Node Diagram Produced  (displayTree function) | 10% | 1. Diagram produced according to requirements |
| Module list produced  (displayModules function) | 10% | 1. Module list produced according to requirements |
| Semester Module list produced  (displaySemester function) | 10% | 1. Semester module list produced according to requirements |
| Search Module list produced  (searchModule function) | 20% | 1. Module search function produces list to requirements |
| Coding Style | 5% | 1. Code structure, comments, etc. |
| Document | 25% | Document   1. Explains the functionality of the code 2. Explains where in memory the binary search tree and variables used in code are and what that memory is/how it is used. 3. Structured, well written, etc. |

University Generic Assessment Description