**GAME 2001**

**Assignment #2 (To be done in groups of 3)**

**Due Date: 12th January 2018 midnight**

**Date Given: 6th December 2017**

The attached file “**projectBase.cpp**” contains basic working code for adding weapons to an in game collection. The weapons are presently stored using a simple hash table.

You are required to modify the C++ “projectBase.cpp“ file to implement the storage of the weapons using a Binary Search Tree.

Part 1) (worth 80% of the marks)

You must implement the ability to perform the following operations

* Insertion
* Search
* In-order Traversal (for printing)

Part 2) (worth 20% of the marks)

You must allow the ability to Delete Weapons from the shop which requires the following operation

* Deletion

**NOTE: The Binary tree is to be ordered by Weapon Name.**

Submission Requirements:

**The Student ID number for each member of the group must be commented at the top of the file.**

**Choose one member of the group to make submissions on behalf of the group.**

**The chosen member can make as many submissions (versions) as they like. Only the last submission will be marked.**

**You must submit ONLY the C++ (cpp) file.**

**The marking scheme is given on the next page.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Trait** | **Excellent (90-100)** | **Good (75-85)** | **Satisfactory(60-75)** | **Unsatisfactory (< 60)** |
| **Delivery**  **(10 marks)** | * Submitted on time and in the correct format. * Completed 90 - 100 percent of the program requirements | * Submitted on time and in the correct format. * Completed 75 -90 percent of the program requirements | * Submitted on time and in the correct format. * Completed 70 -80 percent of the program requirements | * Submitted late or in the wrong format. * Completed less than 70% of the program requirements |
| **Coding Standards and Documentation (10 marks)** | * Includes name, date and assignment number. * Excellent variable names used (no global variables, or vague naming). * Useful documentation descriptions. * All functions commented. * Indented to standard. | * Includes name, date and assignment number. * Appropriate variable names used (little use of global variables, or vague naming). * Useful documentation descriptions. * Most functions commented. Indented to standard | * Includes name, date and assignment number. * Appropriate variable names used (a few use of global variables, or vague naming). * Basic documentation descriptions including purpose for functions. * Mostly indented well | * No name, program description included * Poor or misleading variable names used. Little or no indentation. * Regular use of global variables |
| **Specification and Runtime (25 marks)** | * The program meets all of the specifications required and works. * No errors in output. Output is formatted excellently. * All requirements met | * No errors in output. Output is formatted. * All requirements met. * It also meets most of the other specifications. | * No errors in output. * Output has basic formatting or meets core specifications only. | * Does not run due to errors, data read incorrectly. * Little or no requirement met. * Output is poorly formatted or does not follow specifications. |
| **Efficiency**  **(10 marks)** | * Algorithm is easy to understand and efficient. * Can be maintained or modified with minimal changes | * Algorithm is easy to understand and efficient | * Algorithm is easy to understand and but inefficient (excessive use of variables, loops or conditionals) | * Algorithm is hard to understand and very inefficient (excessive use of variables, loops or conditionals) |