Kyle Knudson

CSCI 591

Cs301188

Section 1

04/18/18

Project 11

Project 11 - Design Document

Introduction:

A binary search tree (BST) is a binary tree for which each node is greater than all the keys in the left subtrees and less than all the ones in the right subtrees. This is often used to represent an ordered list and can allow for easy implementations of ordered data such as an inventory. This program manipulates an inventory that is held in an ordered list. The client program in this project allows the user to perform multiple operations on the ordered list such as searching to see if an item is present, inserting a new item and removing an item from the inventory. This program allows the user to continue to do the manipulations on the inventory until they enter the quit option in the menu. This program also uses files to read data from an existing inventory.

Data Structures:

This project utilizes the order list abstract data type by using the pointer-based Binary Search Tree implementation. This project also uses files to read in data about an inventory.

Functions:

BST(); - Constructor for a binary search tree.

~BST(); - Destructor for a binary search tree , deletes all the nodes in the list.

Empty(); - returns true If the BST is empty and false if there is any nodes in the tree.

Insert(); - public function that calls the help\_insert function.

Remove(); - removes an item from a tree and reconnects the tree so there is not any disjointed nodes.

Report(); - public function that calls the help\_report function

Value() – public function that calls the total\_value function.

Total\_value(); - returns the total value of the items in the inventory using recursion.

Length(); - returns the length of the BST.

Present(); - Returns true If an item is found in the list and false if the item is not in the tree.

Print(); - public function that calls the help\_print function.

Destroy(); - removes all of the nodes from the tree. (same function as the destructor).

Find\_length(); - private function that helps to find the length of the tree using recursion.

Help\_remove(); - private function that helps to remove an item from a tree using recursion.

Help\_report()-reports to the terminal all of the information about an item using recursion.

Remove\_node()- removes a node from the tree

Help\_print()- prints out the content of the tree using recursion.

Help\_insert()-inserts an item into the tree by searching for the correct location and attaching it to that point using recursion.

The Main program:

The main program begins by presenting the user with a menu of choices. The user can read an inventory from a file, Insert an item into the inventory, remove an item from the inventory, report if the inventory is empty, report an item’s information, report the inventory’s total value, and print the inventory to the terminal. If the user selects to read an inventory from a file, the main program opens this file and reads in the data from the file. The data is stored in different nodes in a BST class. After this is done, the user can choose to manipulate the inventory data until they exit out of the loop by choosing the quit option.