Project Report

Project 3A

Files/Folders

|  |
| --- |
| Explorer.h |
| -AVL\_Tree<Folders> folderTree  -std::list<File> fileList  +void addFolder(std::string sName)  +void addFolder(std::string sPath, std::string sName)  +void addFile(std::string filePath, std::string fName, int fSize)  +void deleteFolder(std::string dName)  +void deleteFolder(std::string dPath, std::string dName)  +void deleteFile(std::string filePath, std::string fName)  +void folderSizewrapper(std::string path)  +int folderSize(AVL\_Tree<Folders>\* localroot, std::string fPath, int filesize)  +File getFile(std::string path, std::string fName)  +std::list<File> getFilesWrapper(std::string sQuery)  +std::list<File> getFiles(AVL\_Tree<Folders>\* localroot, std::string query)  +AVL\_Tree<Folders> get\_folderTree() |

UML Diagram

|  |
| --- |
| Main.cpp |
| Int main(); |

|  |
| --- |
| Folders.h |
| -std::string folderName  -AVL\_Tree<Folders> folderTree;  -std::map <std::string, File> filesInFolder  +Folders()  +Folders(std::string name)  +bool addFileToFolder(std::string fName, int fSize)  +bool removeFileFromFolder(std::string fName)  +friend bool operator<(const Folders& f1, const Folders& f2)  +friend std::ostream& operator<<(std::ostream& os, const Folders& folder)  +const std::string get\_folderName()  +const std::string get\_folderPath()  +std::map < std::string, File > get\_filesInFolder() |

|  |
| --- |
| AVL\_Node.h |
| struct AVLNode  int height()  int balance()  virtual ~AVLNode()  void update\_right\_height()  void update\_left\_height()  virtual std::string to\_string() const |

|  |
| --- |
| AVL\_Tree.h |
| +AVL\_Tree()  +AVL\_Tree<Item\_Type> get\_left\_subtree()  +AVL\_Tree<Item\_Type> get\_right\_subtree()  +const Item\_Type& AVL\_Tree<Item\_Type>::get\_data()  +bool is null()  +bool is\_leaf()  +virtual bool insert(const Item\_Type& item)  +bool erase(const Item\_Type& item)  +Item\_Type& find(const Item\_Type& item) const  AVLNode<Item\_Type>\* root;  AVL\_Tree(AVLNode<Item\_Type>\* new\_root) : root(new\_root)  Item\_Type& find(AVLNode<Item\_Type>\* local\_root, const Item\_Type& target) const  bool erase(AVLNode<Item\_Type>\*& local\_root, const Item\_Type& item)  void rebalance(AVLNode<Item\_Type>\*& local\_root)  void adjust\_balance(AVLNode<Item\_Type>\*& node)  void replace\_parent(AVLNode<Item\_Type>\*& old\_root, AVLNode<Item\_Type>\*& local\_root)  bool insert(AVLNode<Item\_Type>\*& local\_root, const Item\_Type& item)  void rebalance\_left(AVLNode<Item\_Type>\*& local\_root)  void rebalance\_right(AVLNode<Item\_Type>\*& local\_root)  void rotate\_right(AVLNode<Item\_Type>\*& local\_root)  void rotate\_left(AVLNode<Item\_Type>\*& local\_root) |

|  |
| --- |
| File.h |
| -std::string fileName;  -int fileSize  +File(std::string name, int size)  +std::string get\_fileName()  +int get\_fileSize()  +void set\_fileSize(int new\_size)  +void set\_fileName(std::string new\_name) |

**Relationships:**

**Main & Explorer (1:N)**

* User may choose to add a folder
* User may choose to delete a folder
* User may choose to add a file
* User may choose to delete a file
* User may choose to access one/multiple file(s)

**Explorer & Folders (1:1)**

* Access and message notification to create file
* Access and message notification to delete file
* Access and message notification to delete folder
* Access and message notification to create folder

**Folders & File (1:1)**

* Creation of files
* Deletion of files

**Folders & AVL Tree (1:1)**

* File deleted will rebalance the tree
* File added will rebalance the tree
* Folder deleted will rebalance the tree
* Folder added will rebalance the tree

**AVL\_Tree & AVL\_Node (1:1)**

* Rebalancing and updating the height of the tree for left
* Rebalancing and updating the height of the tree for right

**Big O Notation:**

**File.h**

File(std::string name, int size) – O(n)

**Folders.h**

Folders(std::string path, std::string name) – O(n)

bool addFileToFolder(std::string fName, int fSize) – O(n)

bool removeFileFromFolder(std::string fName) – O(n)

**Explorer.h**

void addFolder(std::string sName) – O(n)

void addFolder(std::string sPath, std::string sName) – O(n)

void addFile(std::string filePath, std::string fName, int fSize) – O(n)

void deleteFolder(std::string dName) – O(n)

void deleteFolder(std::string dPath, std::string dName) – O(n)

void deleteFile(std::string filePath, std::string fName) – O(n)

int folderSize(AVL\_Tree<Folders>\* localroot, std::string fPath, int filesize) - O(log n)

std::list<File> getFiles(AVL\_Tree<Folders>\* localroot, std::string query) - O(log n)

**AVL\_Tree.h**

Item\_Type& find(AVLNode<Item\_Type>\* local\_root, const Item\_Type& target) const - O(log n)

bool erase(AVLNode<Item\_Type>\*& local\_root, const Item\_Type& item) - O(log n)

bool insert(AVLNode<Item\_Type>\*& local\_root, const Item\_Type& item) - O(log n)

**Assumptions:**

The user will be given options to either add a folder, delete a folder, add a file, delete a file, and to retrieve one or more file(s). Depending on the option that the user chooses there will be a search on the specific file/folder through the AVL tree until the target has been found. After the option of add or deleting the tree will rebalance itself. The user can continue to add and delete files/folders as much as they choose to quit the program itself.