University of Nevada, Reno

CS 302 — Data Structures

Assignment #5

Students:
Joshua Gleason
Josiah Humphrey

Instructor:
Dr. George Bebis

May 3, 2010

Contents

1 Introduction

2 Use of Code

3 Functions

3.1 binaryTree.h

BINARYTREE

```
binaryTree();
```

Purpose

Input

Output

Assumptions

BINARYTREE

```
~binaryTree();
```

Purpose

Input

Output

Assumptions

OPERATOR=

```
void operator=(const binaryTree<iType>&);
```

Purpose

Input

Output

Assumptions

MAKEEMPTY

```
void makeEmpty();
```

Purpose

Input

Output

Assumptions

isEmpty

```
bool isEmpty() const;
```

Purpose

Input

Output

Assumptions

isFull

```
bool isFull() const;
```

Purpose

Input

Output

Assumptions

numberOfNodes

```
int numberOfNodes() const;
```

Purpose

Input

Output

Assumptions

retrieveItem

```
bool retrieveItem(iType&);
```

Purpose

Input

Output

Assumptions

insertItem

```
                    void insertItem(iType);
```

Purpose

Input

Output

Assumptions

deleteItem

```
                    void deleteItem(iType);
```

Purpose

Input

Output

Assumptions

resetTree

```
                    void resetTree(oType);
```

Purpose

Input

Output

Assumptions

getNextItem

```
                    bool getNextItem(iType&, oType);
```

Purpose

Input

Output

Assumptions

printTree

```
                    void printTree(ostream&) const;
```

Purpose

Input

Output

Assumptions

countNodes

```
int countNodes(treeNode<iType>*);
```

Purpose

Input

Output

Assumptions

retrieve

```
bool retrieve(treeNode<iType>*, iType&);
```

Purpose

Input

Output

Assumptions

insert

```
void insert(treeNode<iType>*&, iType);
```

Purpose

Input

Output

Assumptions

deleteOut

```
void deleteOut(treeNode<iType>*&, iType);
```

Purpose

Input

Output

Assumptions

deleteNode

```
                    void deleteNode(treeNode<iType>*&);
```

Purpose

Input

Output

Assumptions

getPredecessor

```
                    void getPredecessor(treeNode<iType>*, iType&);
```

Purpose

Input

Output

Assumptions

print

```
                    void print(treeNode<iType>*, ostream&);
```

Purpose

Input

Output

Assumptions

destroy

```
                    void destroy(treeNode<iType>*&);
```

Purpose

Input

Output

Assumptions

copyTree

```
                    void copyTree(treeNode<iType>*&, treeNode<iType>
```

Purpose

Input

Output

Assumptions

countNodes

```
void countNodes(treeNode<iType>*&);
```

Purpose

Input

Output

Assumptions

preOrder

```
void preOrder(treeNode<iType>*&, queue<iType>&);
```

Purpose

Input

Output

Assumptions

inOrder

```
void inOrder(treeNode<iType>*&, queue<iType>&);
```

Purpose

Input

Output

Assumptions

postOrder

```
void postOrder(treeNode<iType>*&, queue<iType>&);
```

Purpose

Input

Output

Assumptions

3.2 user.h

getName

```
string getName() const
```

Purpose

Input

Output

Assumptions

getPass

```
string getPass() const
```

Purpose

Input

Output

Assumptions

setName

```
void setName( string& rhs )
```

Purpose

Input

Output

Assumptions

setPass

```
void setPass( string& rhs )
```

Purpose

Input

Output

Assumptions

operator>

```
bool operator>( const user& rhs )
```

Purpose

Input

Output

Assumptions

OPERATOR<

```
bool operator<( const user& rhs )
```

Purpose

Input

Output

Assumptions

OPERATOR>=

```
bool operator>=( const user& rhs )
```

Purpose

Input

Output

Assumptions

OPERATOR<=

```
bool operator<=( const user& rhs )
```

Purpose

Input

Output

Assumptions

OPERATOR==

```
bool operator==( const user& rhs )
```

Purpose

Input

Output

Assumptions

3.3 part1.cpp

readFile

```
bool readFile( string fileName, binaryTree<user>&
```

Purpose

Input

Output

Assumptions

storeTree

```
void storeTree( binaryTree<user>& tree, oType ord
```

Purpose

Input

Output

Assumptions

promptForMenu

```
menuChoice promptForMenu();
```

Purpose

Input

Output

Assumptions

3.4 heap.h

reheapDown

```
void reheapDown(int root, int bottom);
```

Purpose

Input

Output

Assumptions

reheapUp

```
void reheapUp(int root, int bottom);
```

Purpose

Input

Output

Assumptions

SWAP

```
void swap(ItemType &a, ItemType &b);
```

Purpose

Input

Output

Assumptions

3.5 pqueue.h

MAKEEMPTY

```
void makeEmpty();
```

Purpose

Input

Output

Assumptions

ISEMPTY

```
bool isEmpty() const;
```

Purpose

Input

Output

Assumptions

ISFULL

```
bool isFull() const;
```

Purpose

Input

Output

Assumptions

ENQUEUE

```
void enqueue(ItemType newItem);
```

Purpose

Input

Output

Assumptions

DEQUEUE

```
void dequeue(ItemType& item);
```

Purpose

Input

Output

Assumptions

3.6 U$_P$*QType.h*

U$_P$*QType*

Purpose

Input

Output

Assumptions

REMOVE

```
void Remove(ItemType);
```

Purpose

Input

Output

Assumptions

UPDATE

```
void Update(ItemType, ItemType);
```

Purpose

Input

Output

Assumptions

PRINTTREE

```
void printTree(std::ostream&);
```

Purpose

Input

Output

Assumptions

3.7 part2.cpp

READFILE

```
bool readFile( string fileName, U_PQType<int>* &t
```

Purpose

Input

Output

Assumptions

PROMPTFORMENU

```
menuChoice promptForMenu();
```

Purpose

Input

Output

Assumptions

4 Bugs and Errors

5 What was Learned

6 Division of Labor

7 Extra Credit