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

$$\tilde{}\,binaryTree();$$

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

$$\text{void countNodes (treeNode<iType>*\&);}$$

Purpose

Input

Output

Assumptions

PREORDER

$$\text{void preOrder (treeNode<iType>*\&, queue<iType>\&);}$$

Purpose

Input

Output

Assumptions

INORDER

$$\text{void inOrder (treeNode<iType>*\&, queue<iType>\&);}$$

Purpose

Input

Output

Assumptions

POSTORDER

$$\text{void postOrder (treeNode<iType>*\&, queue<iType>\&);}$$

Purpose

Input

Output

Assumptions

3.2   user.h

GETNAME

<div align="center">

string getName() const

</div>

    Purpose

    Input

    Output

    Assumptions

GETPASS

<div align="center">

string getPass() const

</div>

    Purpose

    Input

    Output

    Assumptions

SETNAME

<div align="center">

void setName(string& rhs)

</div>

    Purpose

    Input

    Output

    Assumptions

SETPASS

<div align="center">

void setPass(string& rhs)

</div>

    Purpose

    Input

    Output

    Assumptions

OPERATOR>

<div align="center">

bool operator >(const user& rhs)

</div>

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_PQType.h

U_PQTYPE

```
U\_PQType(int);
```

Purpose

Input

Output

Assumptions

REMOVE

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

Purpose

Input

    Output

    Assumptions

UPDATE

$$\text{void Update(ItemType, ItemType);}$$

    Purpose

    Input

    Output

    Assumptions

PRINTTREE

$$\text{void printTree(std::ostream\&);}$$

    Purpose

    Input

    Output

    Assumptions

3.7   part2.cpp

READFILE

$$\text{bool readFile(string fileName, U\\\_PQType<int>* \&}$$

    Purpose

    Input

    Output

    Assumptions

PROMPTFORMENU

$$\text{menuChoice promptForMenu();}$$

    Purpose

    Input

    Output

    Assumptions

4   Bugs and Errors

5   What was Learned

6   Division of Labor

7   Extra Credit