Assignment 5: Small Database using a Binary Search Tree

Jake Gendreau

April 12, 2024

Contents

1 Program Design

1.1 Time Estimate

I estimate that this assignment will take me 4 - 6 hours to complete. I already have a decent idea how it will all work, but the knowing - doing gap is large.

1.2 Data Structures

1.2.1 Binary Search Tree

The BST will be implemented in a separate file through the use of `BSTree.h` and its corresponding implementation, `BSTree.cpp`. It will be sorted based on the name of the show, with the less than case going on the left of the current node, and the greater than or equal to case being on the right of the node. Each node will contain a struct called "show".

1.2.2 Show Struct

- `string name`
- `int startDate`
- `int endDate`
- `string genre`
- `string link`
- `nodeptr actorHead`

The show struct will hold all of the data in all of the listings, even though not all of the data is necessary. The actors / actresses will be stored in a linked list.

1.2.3 BST Functions

- `insertNode(BSTreeNodePtr &head, NodePtr newNode)` - Inserts newNode into the BST sorted by the name stored in newNode.
- `printShows(BSTreeNodePtr head)` - Prints the show names of the BST in order.
- `printActorsInShow(BSTreeNodePtr head, string showName)` - Prints all actors in the show ‘showName‘.
- `printShowsWithActor(BSTreeNodePtr head, string actorName)` - Prints all shows with actor ‘actorName‘.
- `printShowsReleasedBetween(BSTreeNodePtr head, int start, int end)` - Prints all shows that aired in years $\geq$ start and $\leq$ end.
- `deleteTree(BSTreeNodePtr &head)` - Deletes the tree.

1.3 Program

1.3.1 Read infile

1. Get path to the infile through the command line.
2. Check that the infile at the specified path exists, error out otherwise.
3. Repeat the following process:
   (a) Create a new show struct
   (b) Read in top line, adding the name of the show, the start date, and the end date to the struct.
   (c) Read in next line, adding genre to the struct.
   (d) Read in next line, adding link to the struct.
   (e) Read in following lines, adding all of the actors / actresses to the linked list in the struct.
   (f) Skip all of the following empty lines.
4. Use `insertNode()` to insert the new node into the BST.

1.3.2 Display all shows in the tree

1. Use `printShows()` to print all of the shows in the BST.

1.3.3 Display all actors of a given show

1. Use `printActorsInShow()` with (Required) Perry Mason, The Office, The Prisoner, and (My choice) Gilligan's Island, and M*A*S*H.

1.3.4 Display all shows with a given actor

1. Use `printShowsWithActor()` with (Required) Raymond Burr, Bill Mumy, Bob Newhart, and (My choice) Jerry Seinfeld, and Bob Denver.

1.3.5 Display all shows released between two dates

1. Use `printShowsReleasedBetween()` with (Required) 1965 - 1985, and (My choice) 1995 - 2005.

2 Program Log

2.1 Time Requirements

This program took me about 5 hours to complete. I was getting a bit confused on the dynamic arrays, but once I figured those out, it was pretty simple. Implementing the Greedy BeFS algorithm was also way easier than I thought it would be. I only had to rework the enqueue function and add a heuristic function to the program. It was essentially a drop-in replacement.

2.2 Things I encountered

- I forgot to properly place the bounds when searching for available unvisited cells, so I ran into some memory errors there that I found using GDB.
- I'm very happy that we were taught to use DATA_TYPE in header files instead of defined data types, because it made converting the queue from characters to cells.
- With dynamic arrays, my initial implementation was using C syntax because that's what I understood. The C++ version is much easier to read though. The C version does teach it better, I feel.
- The data set was really scuffed. Duplicate entries, whitespace, and inconsistencies in titles and names made the data set much harder to parse. Overall though, not too bad.
- I used function overloading to make it much prettier when calling functions on the show tree.

3 Source Code Files

3.1 showDB.cpp

Listing 1: showDB.cpp

```
/* showDB.cpp
 *
 * CS 121.Bolden...gcc 11.4.0...Jake Gendreau
 * 04/24/24 ...Core i9 13900H POP!_os 22.04 ...gend0188@vandals.uidaho.edu
 *
 * A program to store info about shows in a tree
 *---------------------------------------------------------
 */

#include <iostream>
#include <fstream>
#include "BSTree.cpp"

using namespace std;

BSTree readFile(string);

string getName(string);
string removeFollowingWhiteSpace(string);

int getStartDate(string);
int getEndDate(string);

void insertActor(NodePtr&, string);
void deleteActorList(NodePtr&);

void printActorsInShow(BSTree, string);
void printShowsWithActor(BSTree, string);

int main(int argc, char* argv[])
{
    if(argc != 2)
    {
        cout << "Usage: ./a.out <path_to_datafile>. Exiting program..." << endl;
        exit(-1);
    }

    //read in the tree from the data file
    BSTree showTree = readFile(argv[1]);

    cout << "All show titles in the tree:" << endl;
    showTree.printShows();

    //print actors in show
    printActorsInShow(showTree, "Perry Mason");
    printActorsInShow(showTree, "The Prisoner");
    printActorsInShow(showTree, "Gilligan's Island");
    printActorsInShow(showTree, "M*A*S*H");

    //print shows with actor
    printShowsWithActor(showTree, "Raymond Burr");
    printShowsWithActor(showTree, "Bill Mumy");
    printShowsWithActor(showTree, "Bob Newhart");
    printShowsWithActor(showTree, "Jerry Seinfeld");
    printShowsWithActor(showTree, "Bob Denver");

    showTree.deleteTree();
}

BSTree readFile(string filepath)
{
    BSTree showTree = BSTree();
    string curLine;

    //open file
    ifstream infile;
    infile.open(filepath);

    //ensure that file exists and is open
    if(!infile.is_open())
    {
        cout << "Error opening file at " << filepath << ". Exiting program..." << endl;
        exit(-1);
    }

    //While not at the end of the file
    while(getline(infile, curLine))
    {
        //make new show for each
        Show newShow = Show();

        //skip empty lines
        while(curLine == "")
        {
            if(!getline(infile, curLine)) {
                //exit loop if end of file is reached
                break;
            }
        }

        //check for end of file
        if(infile.eof())
        {
            break;
        }

        //get name and dates
        newShow.name = removeFollowingWhiteSpace(getName(curLine));
        newShow.startDate = getStartDate(curLine);
        newShow.endDate = getEndDate(curLine);

        //get genre
        getline(infile, curLine);
        newShow.genre = removeFollowingWhiteSpace(curLine);

        //get link
        getline(infile, curLine);
        newShow.link = removeFollowingWhiteSpace(curLine);

        //get actors / actresses
        while(curLine != "")
        {
            getline(infile, curLine);
            insertActor(newShow.actorHead, removeFollowingWhiteSpace(curLine));
        }

        //handle duplicates
        if(!showTree.isInTree(newShow.name))
        {
            showTree.insertNode(newShow);
        }

        else
        {
            deleteActorList(newShow.actorHead);
        }
    }

    return showTree;
}

string getName(string curLine)
{
    string name = "";

    //read through string, stopping at parenthesis
    //use curLine[i + 1] to account for the space between the title and dates
    for(int i = 0; curLine[i + 1] != '('; i++)
    {
        name += curLine[i];
    }

    return name;
}
```

```
string removeFollowingWhiteSpace(string curLine)
{
    string returnString = "";

    //get size of string
    int numChars = 0;
    for(int i = 0; curLine[i] != '\0'; i++)
    {
        numChars++;
    }

    //cout << "numChars in " << curLine << ": " << numChars << endl;

    //get whitespace
    int whiteSpace = 0;
    //check for tabs and spaces
    for(int i = numChars - 1; i >= 0 && (curLine[i] == ' ' || curLine[i] == '\t'); i--)
    {
        whiteSpace++;
    }

    for(int i = 0; i < numChars - whiteSpace; i++)
    {
        returnString += curLine[i];
    }

    return returnString;
}

int getStartDate(string curLine)
{
    string startDate = "";

    //navigate until first date is met, defined by open parens
    int i = 0;
    while(curLine[i - 1] != '(')
    {
        i++;
    }

    //add the following four digits to the return string
    for(int x = 0; x < 4; x++)
    {
        startDate += curLine[i + x];
    }

    //return the stoi conversion of startDate
    return stoi(startDate);
}
```

```

int getEndDate(string curLine)
{
    string endDate = "";

    //get size of string
    int numChars = 0;
    for(int i = 0; curLine[i] != ')'; i++)
    {
        numChars++;
    }

    //add the first four of the last 5 characters to the string
    for(int i = numChars - 4; i < numChars; i++)
    {
        endDate += curLine[i];
    }

    //return the stoi conversion of endDate
    return stoi(endDate);
}

void insertActor(NodePtr &head, string actorName)
{
    //handle empty name
    if(actorName == "")
        return;

    //insert new actor at the beginning
    NodePtr n = new node();

    n -> data = actorName;
    n -> next = head;

    head = n;
}

void deleteActorList(NodePtr &head)
{
    //delete linked list of actors
    NodePtr p = head;
    NodePtr q = p;

    while(p != NULL)
    {
        p = p -> next;
        q -> next = NULL;
        delete q;
        q = p;
    }
```

```cpp
}

void printActorsInShow(BSTree showTree, string showName)
{
    cout << "All actors in " << showName << endl;
    showTree.printActorsInShow(showName);
}

void printShowsWithActor(BSTree showTree, string actorName)
{
    cout << "All Shows with " << actorName << endl;
    showTree.printShowsWithActor(actorName);
}
```

3.2 BSTree.cpp

Listing 2: BSTree.cpp

```
/*
BSTree.cpp
Implementation of a BSTree for assignment 5 using a Show struct
Jake Gendreau
April 12, 2024
*/

#include <iostream>
#include "BSTree.h"

using namespace std;

void BSTree::insertNode(BSTreeNodePtr &n, Show show)
{
    //check empty tree
    if(n == NULL)
    {
        BSTreeNodePtr newNode = new BSTreeNode();

        //error check
        if(newNode == NULL)
        {
            cout << "FAILED TO ALLOC NEW NODE IN insertNode(). EXITING PROGRAM..." << endl;
            exit(-1);
        }

        //set data members of newNode
        newNode -> data = show;
        newNode -> left = NULL;
        newNode -> right = NULL;

        //assign it to list
        n = newNode;

        return;
    }

    //handle less than case
    if(show.name < n -> data.name)
    {
        insertNode(n -> left, show);
    }

    //handle greater than or equal to case
    if(show.name >= n -> data.name)
    {
        insertNode(n -> right, show);
    }
}

void BSTree::insertNode(Show show)
{
    insertNode(head, show);
}

void BSTree::printShows(BSTreeNodePtr n)
{
    //print names in order
    if(n != NULL)
    {
        printShows(n -> left);
        cout << " | " << n -> data.name << endl;
        printShows(n -> right);
    }
}

void BSTree::printShows()
{
    printShows(head);
    cout << endl;
}

void BSTree::printActorsInShow(BSTreeNodePtr n, string showName)
{
    if(n == NULL)
    {
        return;
    }

    //if show found
    if(n -> data.name == showName)
    {
        NodePtr p = n -> data.actorHead;

        //print out the actors in the show
        while(p != NULL)
        {
            cout << " | " << p -> data << endl;
            p = p -> next;
        }
    }

    //if showname < current show
    if(showName < n -> data.name)
    {
        printActorsInShow(n -> left, showName);
    }

    //handle showName >= current show
    if(showName >= n -> data.name)
    {
        printActorsInShow(n -> right, showName);
    }
}

void BSTree::printActorsInShow(string showName)
{
    printActorsInShow(head, showName);
    cout << endl;
}

void BSTree::printShowsWithActor(BSTreeNodePtr n, string actorName)
{
    //error check
    if(n == NULL)
    {
        cout << "ERROR: printShowsWithActor() ON EMPTY TREE. EXITING PROGRAM..." << endl;
        exit(-1);
    }

    //traverse through the linked list
    NodePtr p = n -> data.actorHead;

    while(p != NULL)
    {
        //if actor is found, print the name of the show
        if(p -> data == actorName)
        {
            cout << " | " << n -> data.name << endl;
        }

        p = p -> next;
    }

    //go to child nodes
    if(n -> left != NULL)
        printShowsWithActor(n -> left, actorName);

    if(n -> right != NULL)
        printShowsWithActor(n -> right, actorName);
}

void BSTree::printShowsWithActor(string actorName)
{
    printShowsWithActor(head, actorName);
    cout << endl;
```

```
}

void BSTree::printShowsReleasedBetween(BSTreeNodePtr n, int start, int end)
{
    //error check
    if(n == NULL)
    {
        cout << "ERROR: printShowsReleasedBetween() ON EMPTY TREE. EXITING PROGRAM" << endl;
        exit(-1);
    }

    //handle started in given period
    if(n -> data.startDate >= start && n -> data.startDate <= end)
    {
        cout << " | " << n -> data.name << endl;
    }

    //handle ended in given period
    else if(n -> data.endDate >= start && n -> data.endDate <= end)
    {
        cout << " | " << n -> data.name << endl;
    }

    //go to child nodes
    if(n -> left != NULL)
        printShowsReleasedBetween(n -> left, start, end);

    if(n -> right != NULL)
        printShowsReleasedBetween(n -> right, start, end);
}

void BSTree::printShowsReleasedBetween(int start, int end)
{
    printShowsReleasedBetween(head, start, end);
    cout << endl;
}

void BSTree::deleteTree(BSTreeNodePtr n)
{
    //if NULL, return
    if(n == NULL)
        return;

    //handle linked list of actors
    NodePtr p = n -> data.actorHead;
    NodePtr q = p;

    while(p != NULL)
    {
        p = p -> next;
        q -> next = NULL;
        delete q;
        q = p;
    }

    //delete children
    deleteTree(n -> left);
    deleteTree(n -> right);

    //delete current node
    delete n;
}

void BSTree::deleteTree()
{
    deleteTree(head);
}

bool BSTree::isInTree(BSTreeNodePtr n, string showName)
{
    //null case
    if(n == NULL)
        return false;

    //match case
    if(showName == n -> data.name)
        return true;

    //general case
    return(isInTree(n -> left, showName) || isInTree(n -> right, showName));
}

bool BSTree::isInTree(string showName)
{
    return isInTree(head, showName);
}
```

3.3 BSTree.h

Listing 3: BSTree.h

```
/*
BSTree.h
A header file for implementing a binary search tree using the "show" struct
Jake Gendreau
April 12, 2024
*/

#ifndef BST_H
#define BST_H

#include <iostream>

using namespace std;

struct node
{
    string data;
    node* next;
};

typedef node* NodePtr;

struct Show
{
    string name;
    string genre;
    string link;

    int startDate;
    int endDate;

    NodePtr actorHead;
};

class BSTree
{
    private:
        struct BSTreeNode
        {
            Show data;
            BSTreeNode* left;
            BSTreeNode* right;
        };
        typedef BSTreeNode* BSTreeNodePtr;

        BSTreeNodePtr head;

    public:
        //constructor
        BSTree()
        {
            head = NULL;
        }

        //prototypes with overloads
        void insertNode(BSTreeNodePtr&, Show);
        void insertNode(Show);

        void printShows(BSTreeNodePtr);
        void printShows();

        void printActorsInShow(BSTreeNodePtr, string);
        void printActorsInShow(string);

        void printShowsWithActor(BSTreeNodePtr, string);
        void printShowsWithActor(string);

        void printShowsReleasedBetween(BSTreeNodePtr, int, int);
        void printShowsReleasedBetween(int, int);

        void deleteTree(BSTreeNodePtr);
        void deleteTree();

        bool isInTree(BSTreeNodePtr, string);
        bool isInTree(string);
};

#endif
```

4 Input Files

The input files used were large.dat, taken from the website, then small.dat, which was only the first five elements of large.dat.

5 Program output

```
./a.out
Usage: ./a.out <path_to_datafile>. Exiting program...

./a.out lists/large.dat
All show titles in the tree:
 | 3rd Rock from the Sun
 | Alfred Hitchcock Presents
 | All in the Family
 | American Dad!
 | Animaniacs
 | Babylon 5
 | Banacek
 | Batman
 | Battlestar Galactica
 | Benson
 | Bewitched
 | Burke's Law
 | CHiPs
 | Charlie's Angels
 | Chicago Hope
 | Coach
 | Criminal Minds
 | Dexter's Laboratory
 | ER
 | Evening Shade
 | F Troop
 | Family Ties
 | Futurama
 | Gidget
 | Gilligan's Island
 | Gomer Pyle: USMC
 | Happy Days
 | Hawaii Five-O
 | Herman's Head
 | Hogan's Heroes
 | I Dream of Jeannie
 | I Love Lucy
 | Ironside
| JAG
| Jake and the Fatman
| Kojak
| Kung Fu: The Legend Continues
| Lassie
| Law & Order
| Leave It to Beaver
| Little House on the Prairie
| Lost
| Lost in Space
| M*A*S*H
| MacGyver
| Make Room for Daddy
| Mannix
| Marcus Welby, M.D.
| Married with Children
| Mary Tyler Moore
| Matlock
| McCloud
| McHale's Navy
| Mister Ed
| Mod Squad
| Mork & Mindy
| Mr. Lucky
| Murder, She Wrote
| My Three Sons
| NCIS
| NCIS: Los Angeles
| NCIS: New Orleans
| Newhart
| Night Court
| Northern Exposure
| Perry Mason
| Quantum Leap
| Rawhide
| Riptide
| Room 222
| Scarecrow and Mrs. King
| Seinfeld
| St. Elsewhere
| Star Trek
| Star Trek: Deep Space Nine
| Star Trek: Enterprise
| Star Trek: The Next Generation
| Star Trek: Voyager
| Taxi
```

```
 | The A-Team
 | The Adventures of Ozzie & Harriet
 | The Andy Griffith Show
 | The Beverly Hillbillies
 | The Big Valley
 | The Bob Newhart Show
 | The Bullwinkle Show
 | The Carol Burnett Show
 | The Cosby Show
 | The Fall Guy
 | The Flying Nun
 | The Fugitive
 | The Honeymooners
 | The Invaders
 | The Jack Benny Program
 | The Jeffersons
 | The Lucy Show
 | The Man from U.N.C.L.E.
 | The Many Loves of Dobie Gillis
 | The Odd Couple
 | The Office
 | The Phil Silvers Show
 | The Prisoner
 | The Saint
 | The Simpsons
 | The Six Million Dollar Man
 | The Streets of San Francisco
 | The Twilight Zone
 | The Wild Wild West
 | The X-Files
 | Topper
 | Voyage to the Bottom of the Sea
 | WKRP in Cincinnati
 | Walker, Texas Ranger
 | Wonder Woman

All actors in Perry Mason
 | William Talman
 | Ray Collin
 | William Hopper
 | Barbara Hale
 | Raymond Burr

All actors in The Prisoner
 | Leo McKern
 | Peter Swanwick
 | Angelo Muscat
 | Patrick McGoohan

All actors in Gilligan's Island
 | Dawn Wells
 | Russell Johnson
 | Tina Louise
 | Natalie Schafer
 | Jim Backus
 | Alan Hale Jr.
 | Bob Denver

All actors in M*A*S*H
 | Larry Linville
 | David Ogden Stiers
 | Gary Burghoff
 | Kellye Nakahara
 | Mike Farrell
 | Harry Morgan
 | William Christopher
 | Jamie Farr
 | Loretta Swit
 | Alan Alda

All Shows with Raymond Burr
 | Ironside
 | Perry Mason

All Shows with Bill Mumy
 | Babylon 5
 | Lost in Space

All Shows with Bob Newhart
 | Newhart
 | The Bob Newhart Show

All Shows with Jerry Seinfeld
 | Seinfeld

All Shows with Bob Denver
 | Gilligan's Island
 | The Many Loves of Dobie Gillis
```