jason carr

cs300

Project 1

Analysis and runtime.

|  |  |  |
| --- | --- | --- |
| Data type | Best case runtime | Worst Case Runtime |
| Binary Search Tree | 0(logN) | 0(n) |
| Array/Linked List | 0(n) | 0(n) |
| Hash Table | 0(1) | 0(1) |

Evaluation

The Binary Tree is an efficient but complicated way to implement a data search, but in the worst case is as slow as a linked list. A linked list is slow but easy to implement. A hash table is less efficient than a Binary Tree but it is good if we don’t know how many elements are going to be in the data set ahead of time. Since we’re loading from a pre-defined dataset I think a Binary Search tree would be the most efficient way to organize my data.

Binary Tree Route

Import datatree routine(filename){

string courseNum;

string courseTitle;

string coursePrereqs;

io filename;

BinarySearchTree courseTree;

string tempData;

bool found;

if filename != null;

open filename;

else{

error;

return;

}

tempData = filename;

while (!eof filename){

readline filename;

if courseNum && courseTitle != null;

{

while (!eof tempData && found!=true){

readline tempData;

if tempData.coursePrereqs = filename.coursePrereqs;

found=True;

}

if found = true;

courseTree->next = tempData;

else

error: prerequisite not found;

}

else

error: not enough variables;

}

}

Create routine(courseNum,coursePrereqs,courseTitle){

while(!EOF)

courseTree->next =TempData;

if TempData.courseNum = courseNum{

cout<<"Course already exists"<<endl;

return 0;

}

}

courseTree->next = courseNum,coursePrereqs,courseTitle;

}

print routine(courseNum){

while !eof courseTree

{

readline courseTree;

if (courseTree.courseNum=courseNum){

cout<<"Course Number: "<<courseNum<<"Course Title: "<<courseTitle<<"Prerequisites: "<<coursePrereqs;

return;

}

}

Error: course not found;

return;

}

Hash Table route

Import hashdata routine(filename){

string courseNum;

string courseTitle;

string coursePrereqs;

io filename;

HashTable courseTable;

string tempData;

bool found;

if filename != null;

open filename;

else{

error;

return;

}

tempData = filename;

while (!eof filename){

readline filename;

if courseNum && courseTitle != null;

{

while (!eof tempData && found!=true){

readline tempData;

if tempData.coursePrereqs = filename.coursePrereqs;

found=True;

}

if found = true;

courseTable->next = tempData;

else

error: prerequisite not found;

}

else

error: not enough variables;

}

}

Create routine(courseNum,coursePrereqs,courseTitle){

while(!EOF)

courseTable->next =TempData;

if TempData.courseNum = courseNum{

cout<<"Course already exists"<<endl;

return 0;

}

}

courseTable->next = courseNum,coursePrereqs,courseTitle;

}

print routine(courseNum){

while !eof courseTable

{

readline courseTable;

if (courseTable.courseNum=courseNum){

cout<<"Course Number: "<<courseNum<<"Course Title: "<<courseTitle<<"Prerequisites: "<<coursePrereqs;

return;

}

}

Error: course not found;

return;

}

Array Route

Import arraydata routine(filename){

string courseNum;

string courseTitle;

string coursePrereqs;

io filename;

array courseData;

array tempData;

bool found;

if filename != null;

open filename;

else{

error;

return;

}

tempData = filename;

while (!eof filename){

readline filename;

if courseNum && courseTitle != null;

{

while (!eof tempData && found!=true){

readline tempData;

if tempData.coursePrereqs = filename.coursePrereqs;

found=True;

}

if found = true;

courseData.addline()=filename.currentline;

else

error: prerequisite not found;

}

else

error: not enough variables;

}

}

Search Data routine(courseNum){

while !eof courseData

{

readline courseData;

if (courseData.courseNum=courseNum){

cout<<"Course Number: "<<courseNum<<"Course Title: "<<courseTitle<<"Prerequisites: "<<coursePrereqs;

return;

}

}

Error: course not found;

return;

}

print routine(courseNum){

while !eof courseData

{

readline courseData;

if (courseData.courseNum=courseNum){

cout<<"Course Number: "<<courseNum<<"Course Title: "<<courseTitle<<"Prerequisites: "<<coursePrereqs;

return;

}

}

Error: course not found;

return;

}

Menu Routine

Menu Routine{

Print << “Course Information Menu System”

<<”1) Load Course Data”

<<”2) Search for Course”

<<”3) Print all course”

<<”9) Exit”

Try 1{

Goto load data routine}

Try2{

Goto Search Routine}

Try3{

Goto Print all routine}

Try9{

Stop program}

}

Print ALL Routine

Print All{

While !eof data{//whichever routine used

readline courseData;

cout<<"Course Number: "<<courseNum<<"Course Title: "<<courseTitle<<"Prerequisites: "<<coursePrereqs;

loop

}