Marisa Kuyava

CS 300

5-3 Milestone

Binary Tree Pseudocode Milestone 5-3.

//Create Class for Course

```
class Course.{
        string variable  courseNumber
        string variable  courseName
        vector of prerequisites
}
```

//Tree node internal structure to hold courses

```
Struct Node{
        Course* course
        Node* left;
         Node* right;
}
```

// default constructor for Node

```
 Node() {
         Set left equal to nullptr;
        Set right equal to nullptr;
   }
```

// initialize with acourse

```
 Node() :
Parameter: Course course
        Node() {
        Set course equal to aCourse;
   }
```

//Create a class for Binary SearchTree

```
Class BinarySearchTree{
        Private:
                Node* root;
                addNode() – Parameters: Node* node, Course courseNumber
        Public:
                Insert () – Parameters: courseNumber
                Serach() – Parameters: courseNumber
}
```

//Used to validate data for formatting errors before course is inserted

```
lineParser(vector<string> line) {
        if line.size() is equal to 2 line can be added as it has required format{
                Create new course
                Set courseNumber equal to line 0
                Set courseName equal to line 1
                Return new course
        }
        Else if line size is greater than 2{
                Create new course
                Set courseNumber equal to line 0
                Set courseName equal to line 1
                for each additional line until the end of the vector {
                        pushback each line greater than 1 to prerequisite vector
                }
                Return new course
        }
```

Else if line size is less than 2{

PRINT There is an error in the file format. Every course must have a course number and course name

}

}

//Add a bid recursively to a node

addNode()

Parameters: Node* node, Bid bid

if node. bidId is larger than bid. bidId : The bid is added to the leftside of the BinarySearchTree

if the leftnode is equal to null

New node is created from ‘bid’ and this node becomes the leftnode

Else

addNode is called with node ->left and bid parameters

else (if the node. bidId is larger than the bid bidId : The bid is added to the rightside of the tree)

if the rightnode is equal to null

New node is created from ‘bid’ and this node becomes the rightnode

Else

addNode with node -> right and bid parameters

//Insert

void Insert()

Parameter: Bid bid

if the root is null

Make a newNode with bid and set equal to the node root

Else (if the root is not null)

Make a call to public function ‘addNode’ and as a parameter pass the root

//file loading

loadFile(file FileName){

Create BinaryTree

Create vector of strings to hold file data

String variable to hold each line

Open file with Ifstream

while get line finds a next line in the file {

stringstream stst (line)

while stst.good() is to true{

create variable to store substring of line

Use get line to break substring from string using comma delimitator

Push substring to temporary <string> vector

}

Insert temporary line vector to BinaryTree using Insert Function and lineParser function

Clear temporary vector

}

}

//Search

Search (string courseNumber){

Set the node variable ‘current’ equal to the root

While ‘current’ is not equal to null

If current.courseNumber matches courseNumber

return ‘current’ course

else if courseNumber is smaller than current.courseNumber

Traverse the leftside of the BinarySearchTree

else (if courseNumber is greater than current.courseNumber)

Travers the rightside of the BinarySearchTree

return course;

//Print number, name and prerequisites for course

Print (string courseNumer)

Create new course to hold course that will be returned

If course returned by search is empty {

Print ‘Course is not in the catalog’

Return

Else

Print course’s number and Name

For each prerequisite in courses’s prerequisite vector{

Print prerequisite