Benjamin Leanna

Project One – Page Three

CS300 02-09-2023

//global declarations

CONST unsigned INT DEFAULT\_SIZE = # // can input any random number wanted for default size

DOUBLE strToDouble(string str, char ch) //forward declaration

STRUCT course //Define a structure to hold course information

STRING courseNumber //use string incase course’s numbers have letters in them

STRING courseTitle

STRING prerequisites

Course function

INITIALIZE csv parser with path

TRY

IF line has at least 2 elements

FOR loop to read through file

course.courseNumber = to csv location

course.courseTitle = to csv location

course.prerequisites = to csv location to end of line

CATCH

ERROR if files wont open or read

DEFINE quicksort function

SET middle to zero

IF beginning number is greater or equal to end number

RETURN

SET middle to partition function

RUN quicksort function

CLASS HashTable // create hash table class

PRIVATE

STRUCT Node // define structures to hold courses

Course course

INT key

Node \*nextpointer

Node function // default constructor

key = UINT\_MAX

nextpointer = nullptr

Node(Course aCourse) : Node fuction // initialize with a course

course = aCourse

Node(Course aCourse, INT aKey) : Node(aCourse) // initialize course/key

key = aKey

vector<Node> nodes

Unsigned INT tableSize = DEFAULT\_SIZE //set default hashtable size

Unsigned INT hash(int key) function call

PUBLIC

HashTable fuction

HashTable(Unsigned INT size)

VIRTUAL ~HashTable // tilde makes it a bitwise NOT operator

VOID Insert(Course course) function

VOID PrintCourse function

VOID Remove(string coursed) function

Course Search(string coursed) function

HashTable::HashTable function // default constructor

Initialize node structure by resizing tablesize

HashTable::HashTable(size) function // use to improve efficiency of hashing algorithm

Invoke local tablsize to size with this ->

Resize nodes size

HashTable::~HashTable function //Destructor

Erase nodes beginning

Unsigned INT HashTable::hash(key) function // calculate the hash value of a given key

Return key tablesize

DEFINE loadCourseCSV function

INITIALIZE csv parser with csv path

TRY

IF line has at least 2 elements

FOR loop to read through file

course.courseNumber = to csv location

course.courseTitle = to csv location

course.prerequisites = to csv location to end of line

APPEND courseVector with courseNumber, courseTitle, prerequisites

CATCH

ERROR

DEF Course HashTable::PrintCourse(string courseID) function

Course course //create node for course

CREATE key for the given course

IF entry found for the key

PRINT courseID, courseTitle, prerequisites

RETURN course node

IF no entry found for the key

RETURN course

WHILE node not equal to nullptr

IF the current node matches

RETURN course

Node is now equal to next node

RETURN the course

DEF HashTable::PrintAll() function //loops through everything to print

FOR node begin to end iterate

IF key not equal to UINT\_MAX

OUTPUT key, courseID, courseTitle, prerequisites

Node is now equal to next iteration

WHILE node not equal to nullptr

OUTPUT key, courseID, courseTitle, prerequisites

Node is now equal to next iteration