Fiel Edvandro Domingos Muhongo

201406033

Data Structure and Algorithms

Assignment 1 codes

public class IndexRecord

{// Variables declaration:

private String key;

private int where;

// Declaring empty constructor of IndexRecord Class:

public IndexRecord(){

key = new String();

where = 0;

}

// Parameterized IndexRecord Constractor:

public IndexRecord(String s, int w){

key = new String(s);

where = w;

}

// Mtehod compareTo to compare indexRecords

public int compareTo(IndexRecord otherRecord){

return (key.compareTo(otherRecord.key));

}

// Method to display the IndexRecord

public String toString(){

return (key + " " + where);

}

//Method to get where out of this class:

public int getWhere(){

return where;

}

//Method to get where out of this class:

public String getKey(){

return key;

}

}

public class DatabaseRecord

{

//Declaring Variables

private String ID;

private String firstName;

private String lastName;

// DatabaseRecord Empty costructor

public DatabaseRecord(){

ID= new String();

firstName = new String();

lastName = new String();

}

//Parameterized DatabaseRecord constructor

public DatabaseRecord(String a,String b, String c){

ID = new String(a);

firstName = new String(b);

lastName = new String(c);

}

// Methods to get ID,LastName and FirstName out of this class:

public String getID(){

return ID;

}

public String getfirstName(){

return firstName;

}

public String getlastName(){

return lastName;

}

//Methods toString to display the data formated:

public String toString(){

return String.format("%-12s%-14s%-14s",ID,firstName,lastName);

}

}

public class IndexArray

{ //Variables Declaration:

IndexRecord[] Array ;

int next;

// Index array empty constructor

public IndexArray(){

Array =new IndexRecord[100];

next= 0;

}

// Index array parameterized constructor

public IndexArray(int size){

Array =new IndexRecord[size];

next= 0;

}

// Insert method of the ordered Array

public void Insert(IndexRecord val){

int j;

for (j = next-1 ; j>= 0 ; j--){

if (Array[j].compareTo(val) < 0) break;

Array[j+1]= Array[j];

}

Array[j+1]= val;

next ++;

}

// Search Method of the ordered array

public int Search(IndexRecord key){

int hi, lo, mid,num ;

hi = next-1;

lo = 0;

num = -1;

while (lo<=hi){

mid = (lo + hi)/2;

if (Array[mid].compareTo(key)== 0)

num = mid ;

if (Array[mid].compareTo(key)>0)

hi = mid - 1;

else

lo =mid +1;

}

return num;

}

//Delete method of the ordered Array

public int Delete(IndexRecord key){

int where = Search(key);

if (where != -1){

for (int j = where ; j<next-1; j++){

Array[j] = Array[j+1];

}

next--;

}

return(where);

}

}

/\*\*

\* Write a description of class Stack here.

\*

\* @author (Fiel Edvandro Domingos Muhongo)

\* @Student (201406033)

\* @version (Assignment1)

\*///Class Stack implemented in Class

public class Stack

{ //Variables declaration

private int[] theStack;

private int sp;

private int maxS;

// Stack Empty constructor

public Stack(){

sp = 0;

maxS = 10;

theStack = new int [10];

}

// Stack Parameterized constructor

public Stack(int sz){

maxS = sz;

sp=0;

theStack = new int [sz];

}

// Method to check if the Stack is Full

public boolean isFull(){

return (sp==maxS);

}

// Method to check if the Stack is Empty

public boolean isEmpty(){

return (sp == 0);

}

//Method to insert an element into the Array

public void Push(int val ){

theStack[sp++]= val;

}

//Method to delete an element in the Array

public int Pop(){

int p =0;

if(isEmpty()==true)

System.out.println("Error");

else

{

p = theStack[sp-1];

sp--;

}

return p;

}

//Method to get the last element in the Array

public int Peek(){

if (isEmpty() == true)

System.out.print("Stack is Empty");

return (theStack[sp-1]);

}

}

import java.util.Scanner; // importing Scanner

public class DataBase

{ // Variables Definition:

DatabaseRecord[] DatabaseArray ;

private int nextDatabaseRecord;

private int counter;

IndexArray ID;

IndexArray firstName;

IndexArray lastName;

Stack myStack = new Stack(); // intatiation of a Stack

public DataBase(int size){// Parametarazed Database constructor

DatabaseArray = new DatabaseRecord[size];

ID = new IndexArray(size);

lastName = new IndexArray(size);

firstName = new IndexArray(size);

nextDatabaseRecord = 0;

counter = 0;

}

public void addIt(){// Method Addit to Add a new record from the keyBoard

int IndexInStack;// variable to store temporary Index in a Stack

String id;

String ln;

String fn;

if (myStack.isEmpty())

IndexInStack = nextDatabaseRecord;

else

IndexInStack= myStack.Pop();

// Prompting the user to insert data of the record from LastName, FirstName an ID

Scanner input = new Scanner(System.in);

System.out.println("Enter the new record by:");

System.out.println("Last Name:");

ln = input.nextLine();

System.out.println("First Name:");

fn = input.nextLine();

System.out.println("ID:");

id = input.nextLine();

IndexRecord key = new IndexRecord(id,0);

int found = ID.Search(key);

if (found >0)

System.out.println("This record can not be added, cause the ID exists in the database");

else {

DatabaseRecord temp = new DatabaseRecord(id,ln,fn);// creating a variable of the type DatabaseRecord

DatabaseArray[IndexInStack]= temp; // Adding the variable to DatabaseArray

//Adding Values into the three index arrays(ID,lastName,firstName)

ID.Insert(new IndexRecord(id,IndexInStack));

lastName.Insert(new IndexRecord(ln,IndexInStack));

firstName.Insert(new IndexRecord(fn ,IndexInStack));

nextDatabaseRecord++;// increasing Database array counter

counter++;

}

}

// Method to Add data into the database from the Textfile

public void addIt(String id, String ln, String fn){// Add method to read from the Text file

IndexRecord key = new IndexRecord(id,0);// Creating an instance of IndexRecord with a given Id

int found = ID.Search(key); //Searching of the Id in Indexarray Id an returning its position

if (found >0){// Check if id exists show the message if does not Add the Record

System.out.println("This record can not be added, cause the ID exists in the database");

System.out.println( "######################################################");

System.out.println( "");

}else {

DatabaseRecord temp = new DatabaseRecord(id,fn,ln);// instance of DatabaseRecord

DatabaseArray[nextDatabaseRecord]= temp;// Adding a databaseRecord into a Database Array

// Inserting the Records into the three index Arrays

ID.Insert(new IndexRecord(id,nextDatabaseRecord));

firstName.Insert(new IndexRecord(fn,nextDatabaseRecord));

lastName.Insert(new IndexRecord(ln ,nextDatabaseRecord));

nextDatabaseRecord++;

}

}

// Method to find An Id into Index IdArray

public void findIt(){

Scanner input = new Scanner(System.in); // creating an instance of Scanner

System.out.println("Enter the Id:");// prompting the user to insert the Id to be deleted

String id = input.nextLine();

IndexRecord key = new IndexRecord(id,0);//Creating an instace of IndexRecord with a given Id

int found = ID.Search(key);//Searching of the Id in Indexarray Id an returning its position

if (found ==-1 ){// if Statement to check if id exists, if not display a message or if does Display

System.out.println( "Sorry, but Id entered was not found");

System.out.println( "####################################");

} else {

System.out.println(DatabaseArray[ID.Array[found].getWhere()]);

System.out.println( "################################");

}

}

// Method to delete from the Index Array

public int deleteIt(){

Scanner input = new Scanner(System.in); // creating an instance of Scanner

System.out.println("Enter the ID:");// prompting the user to insert the Id to be deleted

String Id = input.nextLine();

IndexRecord key = new IndexRecord(Id,0);//Creating an instace of IndexRecord with a given Id

int pos = ID.Search(key); //Searching of the Id in Indexarray Id an returning its position

if (pos ==-1){

System.out.println("Record not found");

System.out.println( "################################");

}else{

int a = ID.Array[pos].getWhere();

// Get Elemrnts from the Each IndexArray to delete

IndexRecord ln = new IndexRecord(DatabaseArray[a].getlastName(),pos);

IndexRecord fn = new IndexRecord(DatabaseArray[a].getfirstName(),pos);

IndexRecord id= new IndexRecord(DatabaseArray[a].getID(),pos);

// Deleting Elements

lastName.Delete(ln);

firstName.Delete(fn);

ID.Delete(id);

myStack.Push(a);// Puching the deleted index into the Stack

System.out.println("Record Deleted");

System.out.println( "################################");

}

return pos;

}

//Method do display the three indexArray in a descending order

public void ListAscending(IndexArray theArray){

int pos;

for (int i=0;i<theArray.next;i++){

pos = theArray.Array[i].getWhere();

System.out.println(DatabaseArray[pos]);

}

}

//Method do display the three indexArray in a descending order

public void ListDescending(IndexArray theArray){

int pos;

for (int i = theArray.next -1;i>=0;i--){

pos = theArray.Array[i].getWhere();

System.out.println(DatabaseArray[pos]);

}

}

// Method to Print a Database

public String printit(){

return ID+" "+firstName+" "+lastName;

}

}

//Driver Program Copied from Prof Bill's Web

//I took Option 1 of Deletion

import java.util.\*;

public class COSC311Driver

{

public static void main(String[] args)

{

/\*The following declaration declares a data structure that will change from one assignment to the next. For example, you will need to implement

\* the following as a doubly linked list, as well as a tree.

\*/

DataBase d= new DataBase(100);

int response;

Scanner keyboard= new Scanner(System.in);

ReadFile r = new ReadFile();

r.openFile();

r.readFile(d);

r.closeFile();

/\* Read the data into the database from the external disk file here

\* IMPORTANT: duplicate ID numbers should not be added. Disregard

\* the entire record for duplicate IDs

\*/

do

{

System.out.println(" 1 Add a new student");

System.out.println(" 2 Delete a student");

System.out.println(" 3 Find a student by ID");

System.out.println(" 4 List students by ID increasing");

System.out.println(" 5 List students by first name increasing");

System.out.println(" 6 List students by last name increasing");

System.out.println(" 7 List students by ID decreasing");

System.out.println(" 8 List students by first name decreasing");

System.out.println(" 9 List students by last name decreasing");

System.out.println(" ");

System.out.println(" 0 End");

response=keyboard.nextInt();

switch (response)

{

case 1: d.addIt( ); //Note: if the user enters an ID already in use, issue a warning and return to the menu

break;

case 2: d.deleteIt(); //Note: output either "Deleted" or "ID not Found" and return to menu

break;

case 3: d.findIt(); //Note: output the entire record or the message "ID not Found" and return to menu

break;

case 4: d.ListAscending(d.ID);

break;

case 5: d.ListAscending(d.firstName);

break;

case 6: d.ListAscending(d.lastName);

break;

case 7: d.ListDescending(d.ID);

break;

case 8: d.ListDescending(d.firstName);

break;

case 9: d.ListDescending(d.lastName);

break;

default:

}

} while (response!=0);

}

}

/\*\*

import java.util.\*; // Importing all classes of java.util

import java.io.\*;//Importing all classes of java.io

public class ReadFile

{//declaring a variable of thr type Scanner

private Scanner in;

//Method to Open the file

public void openFile(){

try{

in = new Scanner(new File("C:/Users/Edvandro/Desktop/Data.txt"));

}

catch(Exception e){

System.out.println("File not found");

}

}

//Method to read the file into a database

public void readFile(DataBase a){

while (in.hasNext()){

String ln = in.next();

String fn = in.next();

String id = in.next();

a.addIt(id,ln,fn);

}

}

// a method to close the file

public void closeFile(){

in.close();

}

}