Ernie Argel

2/7/2019

CECS 326

[ernie.argel@student.csulb.edu](mailto:ernie.argel@student.csulb.edu)

Initializes a fixed size of library with 1024 lines and each is set to 2 billion to 3 billion bytes. Also, initializes another list (recent\_list) with a set of 128 lines and set to 2 billion to 3 billion bytes. At the start of the program it initializes each byte and each line to a random letter. Then, a user is prompted to look for a word in the given list and that word is searched in each line of the “recent\_list”. If the line is not found in the “recent\_list” it is ejected and reinitialized and copied into the library. The first line in the library is then copied to the “recent\_list.” The user is prompted is until they quit.

#include <iostream>

#include <string>

#include <ctime>

using namespace std;

void displayUi();

char \*getDocs();

int main() {

//initializes each doc/line in the library with null

char \*library\_docs[1024];

for (int i = 0; i < 1024; i++) {

library\_docs[i] = NULL;

}

//initializes each doc/line in the recent list with null

char \*list\_docs[128];

for (int i = 0; i < 128; i++) {

list\_docs[i] = NULL;

}

cout << "Starting Program 1 \n";

/\*

initializes each doc/line in the library with random letters

\*/

cout << "Initializing library w/ random letters \n";

for (int i = 0; i < 1024; i++) {

char \*doc = getDocs();

library\_docs[i] = doc;

}

/\*

initializes each doc/line in the recent list with random letters

\*/

cout << "Initializing recent\_list w/ random letters \n";

for (int i = 0; i < 128; i++) {

char \*doc = getDocs();

list\_docs[i] = doc;

}

while (true) {

displayUi();

int input;

int ejected = 0;

cin >> input;

if (input > 0 && input < 16) {

for (int i = 0; i < 128; i++) {

size\_t found;

string s = list\_docs[i];

switch (input) { //inserts input and finds the corresponding word in the recent list

case 1:

found = s.find("FIRST");

break;

case 2:

found = s.find("CPP");

break;

case 3:

found = s.find("REVIEW");

break;

case 4:

found = s.find("PROGRAM");

break;

case 5:

found = s.find("ASSIGNMENT");

break;

case 6:

found = s.find("CECS");

break;

case 7:

found = s.find("BEACH");

break;

case 8:

found = s.find("ECS");

break;

case 9:

found = s.find("FALL");

break;

case 10:

found = s.find("SPRING");

break;

case 11:

found = s.find("OS");

break;

case 12:

found = s.find("MAC");

break;

case 13:

found = s.find("LINUX");

break;

case 14:

found = s.find("WINDOWS");

break;

case 15:

found = s.find("LAB");

break;

}

if (found == string::npos) {

//cout << "NOT FOUND \n";

for (int j = i; j < 128; j++) { //shifts each doc/line in the recent list up one

list\_docs[j] = list\_docs[j + 1];

}

char \*doc = getDocs(); //returns a another set of reinitialized docs/lines

library\_docs[1023] = doc; //sets end of the library with the new set of doc

list\_docs[127] = library\_docs[0]; //sets end of recent list with the first doc of library

for (int j = i; j < 1024; j++) { //shifts each doc/line in the library up one

library\_docs[j] = library\_docs[j + 1];

}

ejected++; //increments if

}

else {

//cout << "FOUND \n";

}

}

}

cout << "#" << ejected << "ejected \n";

}

getchar();

}

//displays UI

void displayUi() {

cout << "SELECT A WORD, 0 TO QUIT \n";

cout << "1. FIRST \n"

<< "2. CPP \n"

<< "3. REVIEW \n"

<< "4. PROGRAM \n"

<< "5. ASSIGNMENT \n"

<< "6. CECS \n"

<< "7. BEACH \n"

<< "8. ECS \n"

<< "9. FALL \n"

<< "10. SPRING \n"

<< "11. OS \n"

<< "12. MAC \n"

<< "13. LINUX \n"

<< "14. WINDOWS \n"

<< "15. LAB \n";

}

/\*

initializes each doc in with a random letter and returns it

\*/

char \*getDocs() {

char alphabets[26] = { 'A','B','C','D','E','F','G','H','I','J','K','L','M','N',

'O','P','Q','R','S','T','U','V','W','X','Y','Z'};

int randomSize = rand() % (3000000000 - 2000000000 + 1) + 2000000000;

char \*docs = new char[randomSize];

for (int i = 0; i < randomSize - 1; i++) {

docs[i] = alphabets[rand() % 26];

}

return docs;

}