Assignment 8 –CS246

Yu Sheng

// FILE: a8.c

// Purpose: To define a List

// Enables the reading of a database of airport codes

// from a file code.txt

// and allows user to search for airports by their code

// Usage: ./airports code.txt

// Written By: Yu Sheng

// Date: April 17, 2018

// FILE: a8.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "List.h"

#include "Airport.h"

void readData(char \*file, List \*l, int \*n);

int comp(void\* item1, void\* item2);

int more();

int main(int argc, char \*argv[]) {

List l=newList(); // A list of all airports

int nA = 0; // How many airports known

// Load database...

if (argc != 2) {

printf("Cannot proceed: Missing file name.\n");

exit(EXIT\_FAILURE);

}

readData(argv[1],&l, &nA);

// Begin user interaction

char code[5];

int result;

do {

printf("Enter the three-letter code for an airport: ");

scanf("%s", code);

printf("Searching: %s\n", code);

Airport \*result=contains(&l,code,comp);

if (result!=NULL) {

printf("Success: ");

show(\*result);

}

else

printf("Unsuccessful: %s is not a known airport.\n", code);

} while (more());

printf("This was a databse of %d airports.\n",nA);

printf("Goodbye!\n");

return(0);

} // main()

int comp(void\* item1,void\* item2){

const Airport \*a= item1;

return (strcmp(a->code,item2)==0);

}

int more() {

char resp[5];

while (1) {

printf("Would you like to search again (Y/N)? ");

scanf("%s", resp);

if ((strcmp(resp, "Y") == 0) || (strcmp(resp, "y") == 0))

return 1;

else if ((strcmp(resp, "N") == 0) || (strcmp(resp, "n") == 0))

return 0;

else

printf("Please enter a Y or a N.\n");

}

} // more()

void readData(char \*file, List \*l, int \*n) {

// Load airport database into list[] from file, set n

FILE \*fp;

int i = 0;

printf("Opening file: %s\n", file);

fp = fopen(file, "r");

if (fp == NULL) {

printf("Unable to open file %s.\n", file);

exit(EXIT\_FAILURE);

}

printf("Opened file %s.\n", file);

printf("Reading...");

char line[80];

fgets(line, sizeof(line), fp); // ignore first line

while ( (fgets(line, sizeof(line), fp)) != NULL) {

int L = strlen(line);

line[L-1] = '\0';

add(l,parseAirport(line));

i++;

}

\*n = i;

fclose(fp);

printf("...done. [%d]\n", i);

} // readData()

//List.h

#include "node.h"

typedef struct{

int size;

struct node \*head;

struct node \*tail;

}List;

List newList();

int size(const List \*l);

int empty(const List \*l);

void clear(List \*l);

void add(List \*l,void\* item);

void\* get(const List \*l,int index);

void\* contains(const List \*l,void\* item,int (\*comp)(void \*,void \*));

void print(const List \*l);

void\* \*queryAirport(const List \*l,char \*code);

//List.c

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include "List.h"

List newList(){

List \*L = malloc(sizeof(List));

L->head = NULL;

L->tail = NULL;

L->size = 0;

return \*L;

}

int size(const List \*l){

return l->size;

}

int empty(const List \*l){

return l->size == 0;

}

void add(List \*l,void\* item){

struct node \*n=newNode(item);

if(l->size==0)

l->head=l->tail=n;

else{

l->tail->next=n;

l->tail=n;

}

l->size++;

}

void print(const List \*l){

struct node \*n=l->head;

printf("L(%d) = ",l->size);

if(l->size==0){

printf("Null\n");

}else{

while(n !=NULL){

printf("%d ",n->data);

n=n->next;

}

printf("\n");

}

}

void\* get(const List \*l,int index){

if(index<0||index>=l->size){

printf("Error: List index out of bounds %d. Exiting!\n",index);

exit(EXIT\_FAILURE);

}

struct node \*n=l->head;

for(int i=0;i<index;i++)

n=n->next;

return n->data;

}

void clear(List \*l){

struct node \*n=l->head;

struct node \*nxt;

while(n!=NULL){

nxt=n->next;

free(n);

n=nxt;

}

l->head=l->tail=NULL;

l->size=0;

}

void \*contains(const List \*l,void\* item,int (\*comp)(void \*,void \*)){

struct node \*n=l->head;

int i=0;

while(n !=NULL){

if(comp(n->data,item)==1)

return n->data;

n=n->next;

}

return NULL;

}

//node.h

struct node{

void\* data;

struct node \*next;

};

struct node \*newNode(void\* item);

//node.c

#include "node.h"

#include <stdlib.h>

struct node \*newNode(void\* item){

struct node \*n=malloc(sizeof(struct node));

n->data=item;

n->next=NULL;

}

//Airport.h

// Define the Airport data structure

typedef struct {

char \*code; // code

char \*name; // name of airport

char \*city; // city

char \*state; // state

char \*country; // duh

} Airport;

// Functions relating to Airport type

void show(Airport a); // prints out a given airport object

Airport \*parseAirport(char \* str); // Parses str into an Airport object

//Airport.c

#include "Airport.h"

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

void show(Airport a) { // print Airport object

printf("%s - %s, %s, %s (%s).\n",a.code,a.name,a.city,a.state,a.country);

} // show()

Airport \*parseAirport(char \* str) { // Parse srt into an Airport object

Airport \*result=malloc(sizeof(Airport));

char \*token;

token = strtok(str, ","); // code

result->code=malloc(sizeof(char)\*(strlen(token)+1));

strcpy(result->code, token);

token = strtok(NULL, ","); // name

result->name=malloc(sizeof(char)\*(strlen(token)+1));

strcpy(result->name, token);

token = strtok(NULL, ","); // city

result->city=malloc(sizeof(char)\*(strlen(token)+1));

strcpy(result->city, token);

token = strtok(NULL, ","); // state

result->state=malloc(sizeof(char)\*(strlen(token)+1));

strcpy(result->state, token);

token = strtok(NULL, ","); // country

result->country=malloc(sizeof(char)\*(strlen(token)+1));

strcpy(result->country, token);

return result;

} // parseAirport()

//Makefile

a8: a8.o List.o node.o Airport.o

gcc -o a8 a8.o List.o node.o Airport.o

a8.o: List.h Airport.h a8.c

gcc -c a8.c

List.o: List.h List.c

gcc -c List.c

node.o: node.h node.c

gcc -c node.c

Airport.o: Airport.h Airport.c

gcc -c Airport.c

Output

[jsheng@powerpuff a8+]$ make

gcc -c a8.c

gcc -c List.c

gcc -c node.c

gcc -c Airport.c

gcc -o a8 a8.o List.o node.o Airport.o

[jsheng@powerpuff a8+]$ ./a8 code.txt

Opening file: code.txt

Opened file code.txt.

Reading......done. [3376]

Enter the three-letter code for an airport: BMC

Searching: BMC

Success: BMC - Brigham City, Brigham City, UT (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: 172

Searching: 172

Unsuccessful: 172 is not a known airport.

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: 00M

Searching: 00M

Success: 00M - Thigpen, Bay Springs, MS (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: TCS

Searching: TCS

Success: TCS - Truth Or Consequences Municipal, Truth Or Consequences, NM (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: BUZ

Searching: BUZ

Unsuccessful: BUZ is not a known airport.

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: 5A8

Searching: 5A8

Success: 5A8 - Aleknagik, Aleknagik, AK (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: DAR

Searching: DAR

Unsuccessful: DAR is not a known airport.

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: ZZV

Searching: ZZV

Success: ZZV - Zanesville Municipal, Zanesville, OH (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: 22M

Searching: 22M

Success: 22M - Pontotoc County, Pontotoc, MS (USA).

Would you like to search again (Y/N)? Y

Enter the three-letter code for an airport: PHL

Searching: PHL

Success: PHL - Philadelphia Intl, Philadelphia, PA (USA).

Would you like to search again (Y/N)? N

This was a databse of 3376 airports.

Goodbye!

Reflection

I used generic type while implementing the list in this homework assignment and I implemented the queryAirport function using the contains(item1,item2,compare function) function professor mentioned in the class. I was stuck twice in this assignment.

First time is when I changed Airport data type to have no limits on the sizes of the strings(allocate memory dynamically). It printed out a “malloc.c:2399: sysmalloc” error, it turns out to be that we did not add 1 for the null terminator when we use malloc for allocating the memory.

Second time is because of we changed the parameter of add function from Airport to void\*(generic type). I found I got the segmentation fault after I search for an airport. And I print out the list l, found that all produced the same number, which is caused by adding the address of a local variable to the list. Thus I change the return type of parseAirport() to be a pointer variable of Airport, and use malloc to locate that pointer variable in the heap segment. Then the airport can be successfully added to the list.

Questions

1. How many airports in the database are not in the US? Give at least one example.

There are four airports in the database are not in the US.

[jsheng@mira ass8]$ fgrep USA code.txt > c2.txt

[jsheng@mira ass8]$ diff code.txt c2.txt

< ROP,Prachinburi,NA,NA,Thailand

< ROR,Babelthoup/Koror,NA,NA,Palau

< SPN,Tinian International Airport,NA,NA,N Mariana Islands

< YAP,Yap International,NA,NA,Federated States of Micronesia

2. Why did we not need a limit for sizes of strings? Or the size of the linked list?

Because we can now allocate the memory dynamically using malloc function (using pointer variable to store the string). All data is stored in the heap section, and the space used to store the data is determined by the size of data.

3. How would you determine the limit on the number of airports for your program?

We should use linux command, wc. Command “wc” will show how many lines are there in the code.txt, then we can approximately how many airports are there (number of lines -1=3376), and round up the numbers to be the limit on the number of airports for my program.

[jsheng@powerpuff a8+]$ wc code.txt

3377 7817 127473 code.txt