****

**MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII**

**AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

**Calancea Cătălin**

**MI-222**

**Raport**

**pentru lucrarea de laborator Nr.8**

***la cursul de “Structuri de date şi algoritmi”***

Verificat:

**Guțu Maria,** *doctor, conf. univ.*

Departamentul Informatică şi IS,

Facultatea FCIM, UTM

**Chișinău – 2023**

**Scopul lucrării:** Obținerea deprinderilor practice de implementare și de utilizare a tipului abstract de date „Listă simplu înlănțuită” în limbajul C cu asigurarea operațiilor de prelucrare de bază ale listei.

**Nr. variantei:** 4

**Condiția problemei ( sarcinii de lucru ) : Să se scrie 3 fișiere-text în limbajul C pentru implementarea și utilizarea tipului abstract de date „Listă simplu înlănțuită” cu asigurarea operațiilor de prelucrare de bază ale listei:**

**1. Fișier antet cu extensia .h, care conține specificarea structurii de date a elementului listei simplu înlănțuite (conform variantelor) și prototipurile funcțiilor de prelucrare de bază ale listei.**

**2. Fișier cu extensia .c, care conține implementările (codurile) funcțiilor declarate în fișierul antet.**

**3. Fișier al utilizatorului, funcția mаin() pentru prelucrarea listei cu afișarea la ecran a următorului meniu de opțiuni de bază:**

**1. Crearea listei în memoria dinamică**

**2. Introducerea informației despre elementele listei de la tastatură. 3. Afișarea informației despre elementele listei la ecran.**

**4. Căutarea elementului în listă.**

**5. Modificarea câmpurilor unui element din listă.**

**6. Determinarea adresei ultimului element din listă.**

**7. Determinarea lungimii listei (numărul de elemente).**

**8. Interschimbarea a două elemente indicate în listă.**

**9. Sortarea listei.**

**10. Eliberarea memoriei alocate pentru listă.**

**0. Ieșire din program.**

**Structura Farmacie cu câmpurile: denumirea, adresa, telefonul, orele de lucru, volumul de medicamente.**

**Codul programului:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Farmacie {

char denumirea[100];

char adresa[100];

char telefonul[20];

char oreleDeLucru[100];

int volumulDeMedicamente;

struct Farmacie\* urmatorul;

};

struct Farmacie\* creareFarmacie(char denumirea[], char adresa[], char telefonul[], char oreleDeLucru[], int volumulDeMedicamente) {

struct Farmacie\* farmacie = (struct Farmacie\*)malloc(sizeof(struct Farmacie));

strcpy(farmacie->denumirea, denumirea);

strcpy(farmacie->adresa, adresa);

strcpy(farmacie->telefonul, telefonul);

strcpy(farmacie->oreleDeLucru, oreleDeLucru);

farmacie->volumulDeMedicamente = volumulDeMedicamente;

farmacie->urmatorul = NULL;

return farmacie;

}

void inserareLaInceput(struct Farmacie\*\* cap, struct Farmacie\* farmacie) {

farmacie->urmatorul = \*cap;

\*cap = farmacie;

}

void inserareLaSfarsit(struct Farmacie\*\* cap, struct Farmacie\* farmacie) {

if (\*cap == NULL) {

\*cap = farmacie;

} else {

struct Farmacie\* curent = \*cap;

while (curent->urmatorul != NULL) {

curent = curent->urmatorul;

}

curent->urmatorul = farmacie;

}

}

void afisareLista(struct Farmacie\* cap) {

struct Farmacie\* curent = cap;

while (curent != NULL) {

printf("Denumirea: %s\n", curent->denumirea);

printf("Adresa: %s\n", curent->adresa);

printf("Telefonul: %s\n", curent->telefonul);

printf("Orele de lucru: %s\n", curent->oreleDeLucru);

printf("Volumul de medicamente: %d\n", curent->volumulDeMedicamente);

printf("\n");

curent = curent->urmatorul;

}

}

struct Farmacie\* cautareFarmacie(struct Farmacie\* cap, char denumirea[]) {

struct Farmacie\* curent = cap;

while (curent != NULL) {

if (strcmp(curent->denumirea, denumirea) == 0) {

return curent;

}

curent = curent->urmatorul;

}

return NULL;

}

void modificareFarmacie(struct Farmacie\* cap, char denumirea[], char nouaAdresa[], char noulTelefon[], char noileOreDeLucru[], int noulVolumDeMedicamente) {

struct Farmacie\* farmacie = cautareFarmacie(cap, denumirea);

if (farmacie != NULL) {

strcpy(farmacie->adresa, nouaAdresa);

strcpy(farmacie->telefonul, noulTelefon);

strcpy(farmacie->oreleDeLucru, noileOreDeLucru);

farmacie->volumulDeMedicamente = noulVolumDeMedicamente;

} else {

printf("Farmacia cu denumirea %s nu a fost gasita.\n", denumirea);

}

}

struct Farmacie\* ultimulElement(struct Farmacie\* cap) {

struct Farmacie\* curent = cap;

while (curent != NULL && curent->urmatorul != NULL) {

curent = curent->urmatorul;

}

return curent;

}

int lungimeLista(struct Farmacie\* cap) {

int lungime = 0;

struct Farmacie\* curent = cap;

while (curent != NULL) {

lungime++;

curent = curent->urmatorul;

}

return lungime;

}

void interschimbareElemente(struct Farmacie\* cap, char denumirea1[], char denumirea2[]) {

struct Farmacie\* farmacie1 = cautareFarmacie(cap, denumirea1);

struct Farmacie\* farmacie2 = cautareFarmacie(cap, denumirea2);

if (farmacie1 != NULL && farmacie2 != NULL) {

struct Farmacie temp = \*farmacie1;

\*farmacie1 = \*farmacie2;

\*farmacie2 = temp;

} else {

printf("Farmaciile cu denumirile %s si %s nu au fost gasite.\n", denumirea1, denumirea2);

}

}

void sortareLista(struct Farmacie\*\* cap) {

int lungime = lungimeLista(\*cap);

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

struct Farmacie\* curent = \*cap;

struct Farmacie\* urmatorul = curent->urmatorul;

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

if (strcmp(curent->denumirea, urmatorul->denumirea) > 0) {

struct Farmacie temp = \*curent;

\*curent = \*urmatorul;

\*urmatorul = temp;

}

curent = urmatorul;

urmatorul = urmatorul->urmatorul;

}

}

}

void eliberareLista(struct Farmacie\*\* cap) {

struct Farmacie\* curent = \*cap;

while (curent != NULL) {

struct Farmacie\* urmatorul = curent->urmatorul;

free(curent);

curent = urmatorul;

}

\*cap = NULL;

}

int main() {

struct Farmacie\* lista = NULL;

int optiune;

do {

printf("Meniu de optiuni:\n");

printf("1. Crearea listei in memoria dinamica\n");

printf("2. Introducerea informatiei despre elementele listei de la tastatura\n");

printf("3. Afișarea informației despre elementele listei la ecran\n");

printf("4. Cautarea elementului in lista\n");

printf("5. Modificarea campurilor unui element din lista\n");

printf("6. Determinarea adresei ultimului element din lista\n");

printf("7. Determinarea lungimii listei (numarul de elemente)\n");

printf("8. Interschimbarea a doua elemente indicate in lista\n");

printf("9. Sortarea listei\n");

printf("10. Eliberarea memoriei alocate pentru lista\n");

printf("0. Iesire din program\n");

printf("Alege o optiune: ");

scanf("%d", &optiune);

switch (optiune) {

case 1:

eliberareLista(&lista);

printf("Lista a fost creata in memoria dinamica.\n");

break;

case 2: {

char denumirea[100];

char adresa[100];

char telefonul[20];

char oreleDeLucru[100];

int volumulDeMedicamente;

printf("Introduceti denumirea farmaciei: ");

scanf(" %[^\n]", denumirea);

printf("Introduceti adresa farmaciei: ");

scanf(" %[^\n]", adresa);

printf("Introduceti telefonul farmaciei: ");

scanf(" %[^\n]", telefonul);

printf("Introduceti orele de lucru ale farmaciei: ");

scanf(" %[^\n]", oreleDeLucru);

printf("Introduceti volumul de medicamente al farmaciei: ");

scanf("%d", &volumulDeMedicamente);

struct Farmacie\* farmacie = creareFarmacie(denumirea, adresa, telefonul, oreleDeLucru, volumulDeMedicamente);

inserareLaInceput(&lista, farmacie);

printf("Farmacia a fost introdusa in lista.\n");

break;

}

case 3:

afisareLista(lista);

break;

case 4: {

char denumirea[100];

printf("Introduceti denumirea farmaciei cautate: ");

scanf(" %[^\n]", denumirea);

struct Farmacie\* farmacie = cautareFarmacie(lista, denumirea);

if (farmacie != NULL) {

printf("Farmacia a fost gasita.\n");

printf("Denumirea: %s\n", farmacie->denumirea);

printf("Adresa: %s\n", farmacie->adresa);

printf("Telefonul: %s\n", farmacie->telefonul);

printf("Orele de lucru: %s\n", farmacie->oreleDeLucru);

printf("Volumul de medicamente: %d\n", farmacie->volumulDeMedicamente);

} else {

printf("Farmacia cu denumirea %s nu a fost gasita.\n", denumirea);

}

break;

}

case 5: {

char denumirea[100];

char nouaAdresa[100];

char noulTelefon[20];

char noileOreDeLucru[100];

int noulVolumDeMedicamente;

printf("Introduceti denumirea farmaciei de modificat: ");

scanf(" %[^\n]", denumirea);

printf("Introduceti noua adresa a farmaciei: ");

scanf(" %[^\n]", nouaAdresa);

printf("Introduceti noul telefon al farmaciei: ");

scanf(" %[^\n]", noulTelefon);

printf("Introduceti noile ore de lucru ale farmaciei: ");

scanf(" %[^\n]", noileOreDeLucru);

printf("Introduceti noul volum de medicamente al farmaciei: ");

scanf("%d", &noulVolumDeMedicamente);

modificareFarmacie(lista, denumirea, nouaAdresa, noulTelefon, noileOreDeLucru, noulVolumDeMedicamente);

printf("Farmacia a fost modificata.\n");

break;

}

case 6: {

struct Farmacie\* ultimaFarmacie = ultimulElement(lista);

if (ultimaFarmacie != NULL) {

printf("Adresa ultimului element din lista: %p\n", ultimaFarmacie);

} else {

printf("Lista este goala.\n");

}

break;

}

case 7: {

int lungime = lungimeLista(lista);

printf("Lungimea listei: %d\n", lungime);

break;

}

case 8: {

char denumirea1[100];

char denumirea2[100];

printf("Introduceti denumirea primei farmacii de interschimbat: ");

scanf(" %[^\n]", denumirea1);

printf("Introduceti denumirea celei de-a doua farmacii de interschimbat: ");

scanf(" %[^\n]", denumirea2);

interschimbareElemente(lista, denumirea1, denumirea2);

printf("Farmaciile au fost interschimbate.\n");

break;

}

case 9:

sortareLista(&lista);

printf("Lista a fost sortata.\n");

break;

case 10:

eliberareLista(&lista);

printf("Memoria alocata pentru lista a fost eliberata.\n");

break;

case 0:

printf("Programul a fost inchis.\n");

break;

default:

printf("Optiunea aleasa nu este valida. Va rugam sa reintroduceti o optiune valida.\n");

break;

}

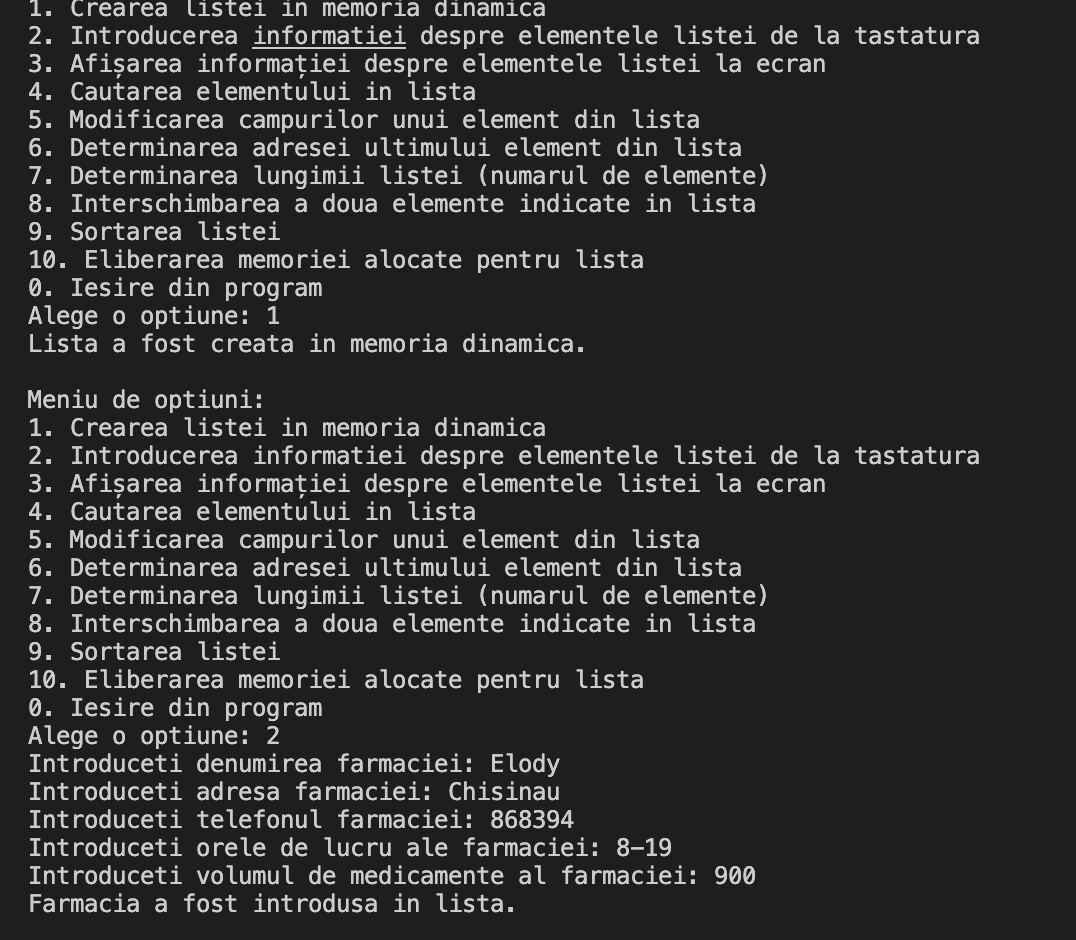
printf("\n");

} while (optiune != 0);

return 0;

}

**Output:**

****

În concluzie, lucrarea de laborator nr 8 , a avut ca scop obținerea deprinderilor practice de implementare și utilizare a tipului abstract de date "Listă simplu înlănțuită" în limbajul C, cu asigurarea operațiilor de prelucrare de bază ale listei. Prin acest exercițiu, am învățat și ne-am familiarizat cu conceptele de bază ale listelor simplu înlănțuite și am dobândit competențe în implementarea acestui tip de structură de date.

Prin intermediul fișierului antet (.h) am definit structura de date a elementului listei și prototipurile funcțiilor necesare pentru operațiile de bază. În fișierul de implementare (.c), am realizat codurile sursă pentru aceste funcții, care permit adăugarea, căutarea, modificarea, sortarea și alte operații asupra listei simplu înlănțuite.

De asemenea, am creat un fișier al utilizatorului (main.c) în care am implementat funcția main pentru a demonstra funcționalitatea listei și pentru a permite interacțiunea utilizatorului cu aceasta. Am creat un meniu de opțiuni prin intermediul căruia utilizatorul poate efectua diverse acțiuni asupra listei, cum ar fi crearea, introducerea, afișarea, căutarea, modificarea, sortarea și eliberarea memoriei.

Prin această lucrare de laborator, am obținut deprinderi practice în implementarea și utilizarea listelor simplu înlănțuite, ceea ce ne permite să gestionăm eficient și flexibil colecții de date într-un mod dinamic și optimizat. Aceste cunoștințe și deprinderi pot fi aplicate în diverse domenii de programare și pot fi extinse pentru a implementa și alte tipuri de structuri de date complexe.

În final, acest exercițiu ne-a ajutat să dezvoltăm abilități practice în limbajul C, să înțelegem mecanismele de lucru ale listelor simplu înlănțuite și să dobândim cunoștințe și experiență în implementarea și utilizarea lor într-un mod eficient și corect.