|  |
| --- |
| **Ministerul Educaţiei și Cercetării**  **al Republicii Moldova**  **Universitatea Tehnică a Moldovei**  **Facultatea Calculatoare, Informatică și Microelectronică**  **RAPORT**  Lucrarea de laborator nr.4  *La Structuri de date si Algoritmi*  A efectuat:  st. gr. TI-216 Roșca Dorin  A verificat: Mititelu Vitalie  Chişinău – 2022 |

|  |
| --- |
| **Lucrare de laborator nr. 4.**  **Tema:**  Algoritmi de prelucrare a listelor liniare simplu înlănțuite (listelor unidirecționale)  **Scopul lucrării**:  Obținerea deprinderilor practice de implementare și de utilizare a tipului abstract de date (TAD) „Listă simplu înlănțuită” în limbajul C cu asigurarea operațiilor de prelucrare de bază ale listei.  **Sarcina (conform variantelor)**:  **Să se scrie 3 fișiere-text în limbajul C pentru implementarea și utilizarea TAD „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 sau .cpp, 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 memori  **Varianta 14:**  Fisierul cu Extensia .h:  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #include <conio.h>  typedef struct Universitate  {  char denumire[100];  char adresa[100];  int telefon;  int nr\_fac;  int nr\_elev;  struct Universitate \* next;//prima Universitate din lista  }Universitate;  int SIZE;  Universitate \* UNIVERSITATE = NULL;  int firstRun = 1;  int Creare(){  Universitate \*p, \*q;  int i=1;  printf("Introdu numarul de Universitati: ");  scanf("%d", &SIZE);  fflush(stdin);  for(i=1; i<=SIZE; ++i)  {  q = (Universitate \*)malloc(sizeof(Universitate));  printf("Introduceti datele despre Universitate %d:\n", i);  printf("Denumirea: "); gets(q->denumire);fflush(stdin);  printf("Adresa: "); gets(q->adresa);fflush(stdin);  printf("Nr. de telefon: "); scanf("%d", &q->telefon);fflush(stdin);  printf("Nr. facultati: "); scanf("%d", &q->nr\_fac);fflush(stdin);  printf("Nr. de Elevi: "); scanf("%f", &q->nr\_elev);fflush(stdin);  q->next = NULL;  if(UNIVERSITATE == NULL)  UNIVERSITATE = q;//q a devenit primul si unicul element al listei  else  p->next = q;//facem legatura- elementul care a fost ultimul va contine adresa nodului nou adaugat  p = q;//adresa ultimului element  printf("\n");  }  printf("\nDatele despre %d Universitati au fost inscrise in lista!\n", i-1);  return 1;  }  void Afisare(){  Universitate \*p;  if(UNIVERSITATE == NULL)  printf("Lista este vida! Mai intai alocati memorie!\n");  else{  printf("Lista Universitatilor: \n");  p = UNIVERSITATE;  for(int i=1; i<=SIZE; ++i)  {  printf("\nUniversitatea %d:\n", i);  printf("\t1.Denumirea: %s\n", p->denumire);  printf("\t2.Adresa: %s\n", p->adresa);  printf("\t3.Nr. de telefon: %d\n", p->telefon);  printf("\t4.Nr. de facultati: %d\n", p->nr\_fac);  printf("\t5.Nr. de Elevi: %.2f\n", p->nr\_elev);  p = p->next;  }  }  }  int subMeniu()  {  printf("Dupa denumire..............1\n");  printf("Dupa adresa................2\n");  printf("Dupa nr. de telefon........3\n");  printf("Dupa nr. de facultati......4\n");  int optiune;  scanf("%d", &optiune);  fflush(stdin);  return optiune;  }  Universitate \*Cautare(){  if(UNIVERSITATE== NULL){  printf("Lista este vida!\n");  }  char denumire[20], adresa[20];  Universitate \*p, \*q = NULL;  int telefon, nr\_fac;  int optiune = subMeniu();  switch(optiune){  case 1:{  printf("Introduceti denumirea cautata: ");  gets(denumire);  fflush(stdin);  printf("Universitati cu denumirea| %s:\n\n", denumire);  p = UNIVERSITATE;  while(p){  if(strcmp(p->denumire, denumire) == 0){  printf("\t1.Denumirea: %s\n", p->denumire);  printf("\t2.Adresa: %s\n", p->adresa);  printf("\t3.Nr. de telefon: %d\n", p->telefon);  printf("\t4.Nr. de facultati: %d\n", p->nr\_fac);  printf("\t5.Nr. de elevi: %.2f\n", p->nr\_elev);  if(q == NULL)  q = p;//adresa ultimului element al listei  }  p = p->next;  }  if(q == NULL)  printf("Nu am gasit asa Universitate!\n\n");  return q;  }  break;  case 2:{  printf("Introduceti adresa cautata: ");  gets(adresa);  fflush(stdin);  printf("Universitati cu adresa| %s:\n\n", adresa);  p = UNIVERSITATE;  while(p){  if(strcmp(p->adresa, adresa) == 0){  printf("\t1.Denumirea: %s\n", p->denumire);  printf("\t2.Adresa: %s\n", p->adresa);  printf("\t3.Nr. de telefon: %d\n", p->telefon);  printf("\t4.Nr. de facultati: %d\n", p->nr\_fac);  printf("\t5.Nr. de elevi: %.2f\n", p->nr\_elev);  if(q == NULL)  q = p;  }  p = p->next;  }  if(q == NULL)  printf("Nu am gasit asa Universitate!\n\n");  return q;  }  break;  case 3:  {  printf("Introduceti nr. de telefon cautat: ");  scanf("%f", &telefon);  printf("Lista universitatilor cu nr. de telefon| %d:\n\n", telefon);  p = UNIVERSITATE;  while(p){  if(p->telefon == telefon){  printf("\t1.Denumirea: %s\n", p->denumire);  printf("\t2.Adresa: %s\n", p->adresa);  printf("\t3.Nr. de telefon: %d\n", p->telefon);  printf("\t4.Nr. de facultati: %d\n", p->nr\_fac);  printf("\t5.Nr. de elevi: %.2f\n", p->nr\_elev);  if(q == NULL)  q = p;  }  p = p->next;  }  if(q == NULL)  printf("Nu am gasit asa Universitate!\n\n");  return q;  }  break;  case 4:  {  printf("Introduceti nr. de telefon cautat: ");  scanf("%f", &nr\_fac);  printf("Lista Universitatilor cu nr. de telefon| %d:\n\n", nr\_fac);  p = UNIVERSITATE;  while(p){  if(p->nr\_fac == nr\_fac)  {  printf("\t1.Denumirea: %s\n", p->denumire);  printf("\t2.Adresa: %s\n", p->adresa);  printf("\t3.Nr. de telefon: %d\n", p->telefon);  printf("\t4.Nr. de facultati: %d\n", p->nr\_fac);  printf("\t5.Nr. de elevi: %.2f\n", p->nr\_elev);  if(q == NULL)  q = p;  }  p = p->next;  }  if(q == NULL)  printf("Nu am gasit asa Universitate!\n");  return q;  }  break;  default:  printf("Criteriu incorect!\n");  }  }  void Modificare()  {  Universitate \*p = Cautare();  if(p)  {  printf("Introduceti datele schimbate:\n");  printf("\t1.Denumirea: "); gets(p->denumire); fflush(stdin);  printf("\t2.Adresa: ");gets(p->adresa); fflush(stdin);  printf("\t3.Nr. de telefon: ");scanf("%d", &p->telefon);  printf("\t4.Nr. de facultati: ");scanf("%d", &p->nr\_fac);  printf("\t5.Nr. de elevi: ");scanf("%f", &p->nr\_elev); fflush(stdin);  }  }  void Ult\_elem()  {  Universitate \*p;  p = UNIVERSITATE;  while(p)  {  p = p->next;  if(p->next==NULL)  printf("Adresa ultimului element din lista este: %d", p);  }  }  int Lungimea()  {  Universitate \*p;  int lungime=0;  p=UNIVERSITATE;  while(p)  {  p=p->next;  lungime++;  }  printf("\n\nAvem %d elemente in lista", lungime);  }  void Copiere(Universitate \*p, Universitate \*q)  {  strcpy(p->denumire, q->denumire);  strcpy(p->adresa, q->adresa);  p->telefon = q->telefon;  p->nr\_fac = q->nr\_fac;  p->nr\_elev = q->nr\_elev;  }  void Interschimb(){  Universitate \*p, \*q, \*r;  if(UNIVERSITATE == NULL)  {  printf("Lista este vida! Mai intai vom aloca memoria si vom crea lista \n");  Creare();  }  int primul;  int al\_doilea;  int temp;  Afisare();  printf("Introduceti numarul de ordine a primului element: ");  scanf("%d", &primul);  printf("Introduceti numarul de ordine a celui de-al doilea element: ");  scanf("%d", &al\_doilea);  printf("Elementele");  p = UNIVERSITATE;  q = UNIVERSITATE;  int i = 1;  int j= 1;  while(p && i != primul)  {  p = p->next;  i++;  }  while(q && j != al\_doilea)  {  q=q->next;  j++;  }  if(p == NULL)  {  Afisare();  printf("Dati un numarul de ordine de la 1 la %d pentru primul element!\n",i-1);  }  else if(q == NULL)  {  Afisare();  printf("Dati un numarul de ordine de l1 la %d pentru al 2-lea elemnt!\n",j-1);  }  else  {  r = (Universitate \*)malloc(sizeof(UNIVERSITATE));  strcpy(r->denumire, p->denumire);  strcpy(r->adresa, p->adresa);  r->telefon = p->telefon;  r->nr\_fac = p->nr\_fac;  r->nr\_elev = p->nr\_elev;  //r=p;  Copiere(p,q);  Copiere(q,r);  free(r);  Afisare();  }  }  void Sortare()  {  int temp;  if(UNIVERSITATE == NULL)  {  printf("Lista este vida! Mai intai vom aloca memoria si vom crea lista \n");  Creare();  }  int optiune = subMeniu();  Universitate \*p, \*q = (Universitate\*)malloc(sizeof(Universitate));  int sortat;//logica  do  {  sortat = 1;  p = UNIVERSITATE;  while(p->next)  {  if(optiune == 1 && strcmp(p->denumire, p->next->denumire)>0 ||  optiune == 2 && strcmp(p->adresa, p->next->adresa)>0 ||  optiune == 3 && p->telefon > p->next->telefon ||  optiune == 4 && p->nr\_fac > p->next->nr\_fac)  {  Copiere(q, p);  Copiere(p, p->next);  Copiere(p->next, q);  sortat = 0;  }  p = p->next;  }  }  while(sortat == 0);  Afisare();  printf("\nSortarea dupa criteriul ales a fost realizata cu succes!\n");  }  void Eliberare\_mem()  {  Universitate \*p, \*q;  if(UNIVERSITATE == NULL)  {  printf("Lista este vida! Alocati memorie\n");  Creare();  }  while(UNIVERSITATE)  {  p = UNIVERSITATE;  UNIVERSITATE = UNIVERSITATE->next;  free(p);  }  printf("Lista a fost stearsa cu succes!\n");  }  Codul programului In limbajul C:      **Rezultatul aparut in Consola:**    **Crearea Liste:**    **Afisarea Listei:**    **Cautare 1:**    **Cautare 2:**    **Modificare:**    **Determinarea Ultimului Element:**    **Determinarea Numarului de Elemente:**    **Interschimbarea a 2 elemente:**    **Sortate 1:**    **Sortare 2:**    **Sortare 3:**    **Eliberarea Memoriei::**     * **Concluzii:** În urma efectuării lucrării de laborator nr. 4 la SDA am studiat lucrului cu listele liniare simplu înlanțuite în limbajul de programare C . * Am realizat principalele operații de lucru cu listele, ca de exemplu: crearea și alocarea dinamică a memoriei, afișarea, căutarea unui element, modificarea câmpului unui element, sortarea și altele. * De asemenea am însușit lucrul cu fișierele cu extensia .h. Am creat un asemenea fișier în care am plasat toate funcțiile necesare la lucrul cu listele, apoi in codul principal doar am declarat biblioteca creata(function.h în cazul meu), astfel am putea lucra cu mai multe programe folosind aceeași bibliotecă. |
|  |