

**MINISTERUL EDUCAȚIEI, CULTURII ȘI CERCETĂRII**

**AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Informatică şi Ingineria Sistemelor**

**gr. IA-233, Chistol Maxim**

**Raport**

**pentru lucrarea de laborator Nr.3**

***la cursul de “Structuri de Date si Algoritmi”***

Verificat:

**Maria Guțu,** *Lector Universitar.*

Departamentul Informatică şi IS,

Facultatea FCIM, UTM

**Chișinău 2024**

Cuprins:

[Scopul lucrării: 4](#_Toc147414571)

[Sarcina 1 4](#_Toc147414572)

[Sarcina 2 5](#_Toc147414573)

[Sarcina 3 5](#_Toc147414574)

[Sarcina 4 5](#_Toc147414575)

[Sarcina 5 6](#_Toc147414576)

[Sarcina 6 6](#_Toc147414577)

[Sarcina 7 6](#_Toc147414578)

[Sarcina 8 7](#_Toc147414579)

[Sarcina 9 7](#_Toc147414580)

[Sarcina 10 8](#_Toc147414581)

[Sarcina 11 8](#_Toc147414582)

[Sarcina 12 8](#_Toc147414583)

[Sarcina 13 9](#_Toc147414584)

[Sarcina 14 9](#_Toc147414585)

[Sarcina 15 9](#_Toc147414586)

[Sarcina 16 10](#_Toc147414587)

[Sarcina 17 10](#_Toc147414588)

[Sarcina 18 10](#_Toc147414589)

[Sarcina 19 11](#_Toc147414590)

[Sarcina 20 11](#_Toc147414591)

[Sarcina 21 11](#_Toc147414592)

[Sarcina 22 11](#_Toc147414593)

[Sarcina 23 12](#_Toc147414594)

[Sarcina 24 12](#_Toc147414595)

[Sarcina 25 12](#_Toc147414596)

[Sarcina 26 13](#_Toc147414597)

[Sarcina 27 13](#_Toc147414598)

[Sarcina 28 13](#_Toc147414599)

[Sarcina 29 14](#_Toc147414600)

[Sarcina 30 14](#_Toc147414601)

[Sarcina 31 14](#_Toc147414602)

[Sarcina 32 14](#_Toc147414603)

[Concluzie: 15](#_Toc147414604)

Scopul lucrării:

Programarea algoritmilor de prelucrare a tablourilor 1D & 2D, aplicând tehnicile și metodele de sortare prin utilizarea funcțiilor, pointerilor, alocării dinamice a memoriei în limbajul C.

# Sarcina 1

Elaborați un program C care va crea un meniu recursiv. Acesta trebuie să cuprindă următoarele funcții în C (cu apelare ulterioară ale acestora în funcția main()):

1. Alocarea dinamică a memoriei;

2. Introducerea valorilor tabloului de la tastatură;

3. Completarea tabloului cu valori random;

4. Sortarea elementelor tabloului conform variantelor:

1) Bubble Sort;

2) Selection Sort;

3) Insertion Sort;

5. Eliberarea memoriei și ieșirea din program.

Conform variantei: Se citeşte o variabilă n, k şi o matrice A=(ai,j)nxn pătratică. Să se scrie un program care va crea un vector și va sorta crescător/descrescător elementele lui. Elementele vectorului sunt elementele matricei a căror suma cifrelor este egală cu k.

**Codul:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <stdbool.h>**

**#include <time.h>**

**void meniu(int \*\*a, int \*arr, int n);**

**void citire(int \*\*a,int \*arr, int n);**

**void alocare(int \*\*a,int \*arr, int n);**

**void afisare(int \*\*a,int \*arr, int n);**

**void random\_arr(int \*\*a,int \*arr, int n);**

**void sortare\_bubble1Dcres(int \*\*a, int \*arr, int n);**

**void sortare\_selection1Dcres(int \*\*a, int \*arr, int n);**

**void sortare\_insertion1Dcres(int \*\*a, int \*arr, int n);**

**int sumaCifrelor(int numar);**

**void fornew1D(int \*\*a, int \*arr, int n);**

**void freemem2D(int \*\*a,int \*arr, int n);**

**void freemem1D(int \*\*a, int \*arr, int n);**

**void afisare1D(int \*\*a, int \*arr, int index);**

**void sortare\_bubble1Ddesc(int \*\*a, int \*arr, int n);**

**void sortare\_selection1Ddesc(int \*\*a, int \*arr, int n);**

**void sortare\_insertion1Ddesc(int \*\*a, int \*arr, int n);**

**void alocare1D(int \*\*a, int \*arr, int n);**

**int main() {**

**int n, k, numar, \*\*a = NULL, \*arr=NULL;**

**printf("----------Introduceti dimensiunile la ARRRAY!!!------- :\n ");**

**printf("Dati nr de linii si coloane n:\n");**

**scanf("%i", &n);**

**printf("Dati suma numerelor k:\n");**

**scanf("%i", &k);**

**meniu(a, arr, n);**

**return 0;**

**}**

**void meniu(int \*\*a, int \*arr, int n) {**

**int option;**

**printf("\n--------Lista de optiuni--------\n"**

**"1. Alocarea dinamica a memoriei\n"**

**"2. Introducerea elementelor tabloului de la tastatura\n"**

**"3. Elemente random pentru 2D\n"**

**"4. Sortarea elementelor tabloului Bubble sort crescator\n"**

**"5. Sortarea elementelor tabloului Bubble sort descrescator\n"**

**"6. Sortarea elementelor tabloului Selection sort crescator\n"**

**"7. Sortarea elementelor tabloului Selection sort descrescator\n"**

**"8. Sortarea elementelor tabloului Insertion sort crescator\n"**

**"9. Sortarea elementelor tabloului Insertion sort descrescator\n"**

**"10. Afisarea elementelor tabloului la ecran 2D\n"**

**"11. Afisarea elementelor tabloului la ecran 1D\n"**

**"12. Alocarea dinamica a memoriei vectorului 1D\n"**

**"13, Calcularea sumei\n"**

**"0. Iesire din program\n");**

**printf("Selectati si introduceti nr optiunii dorite: ");**

**scanf("%i", &option);**

**switch (option) {**

**case 0: {**

**printf("Se efectueaza iesire din program...");**

**freemem2D(a, arr, n);**

**freemem1D(a, arr, n);**

**break;**

**}**

**case 1: {**

**alocare(a, arr, n);**

**break;**

**}**

**case 2: {**

**citire(a, arr, n);**

**break;**

**}**

**case 3: {**

**random\_arr(a, arr, n);**

**break;**

**}**

**case 4: {**

**sortare\_bubble1Dcres(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 5: {**

**sortare\_bubble1Ddesc(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 6: {**

**sortare\_selection1Dcres(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 7: {**

**sortare\_selection1Ddesc(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 8: {**

**sortare\_insertion1Dcres(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 9: {**

**sortare\_insertion1Ddesc(a, arr, n);**

**meniu(a, arr, n);**

**break;**

**}**

**case 10: {**

**afisare(a, arr, n);**

**break;**

**}**

**case 11: {**

**afisare1D(a, arr, n);**

**break;**

**}**

**case 12:**

**alocare1D(a, arr, n);**

**break;**

**case 13:**

**fornew1D(a, arr, n);**

**break;**

**default: {**

**printf("\n-----------Nu exista optiunea!!!-----------\n");**

**meniu(a, arr, n);**

**}**

**}**

**}**

**void alocare(int \*\*a,int \*arr, int n) {**

**a = malloc(n \* sizeof(int \*));**

**for (int i = 0; i < n; i++) {**

**a[i] = calloc(n, sizeof(int));**

**}**

**if (a == NULL) {**

**printf(" Eroare. Nu s-a alocat memorie.\n");**

**exit(1);**

**} else {**

**printf("Succes.\n");**

**meniu(a, arr, n);**

**}**

**}**

**void alocare1D(int \*\*a, int \*arr, int n){**

**arr = malloc(n \* sizeof(int \*));**

**for(int i = 0; i < n; i++){**

**arr[i] = calloc(n, sizeof(int));**

**}**

**if (arr == NULL) {**

**printf("Eroare. Nu s-a alocat memorie.\n");**

**exit(1);**

**} else {**

**printf("Succes.\n");**

**meniu(a, arr, n);**

**}**

**}**

**void citire(int \*\*a, int \*arr, int n) {**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j < n; j++) {**

**scanf("%i", &a[i][j]);**

**}**

**}**

**meniu(a, arr, n);**

**}**

**void afisare(int \*\*a, int \*arr, int n) {**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j < n; j++) {**

**printf("%i ", a[i][j]);**

**}**

**printf("\n\n");**

**}**

**meniu(a, arr, n);**

**}**

**void random\_arr(int \*\*a, int \*arr, int n) {**

**srand(time(NULL));**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j < n; j++) {**

**a[i][j] = rand() % 100;**

**}**

**}**

**printf("Tabloul a fost completat cu valori aleatorii <<100%%>>");**

**meniu(a, arr, n);**

**}**

**void sortare\_bubble1Dcres(int \*\*a, int \*arr, int n) {**

**int aux;**

**for (int i = 0; i < n; i++) {**

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

**if (arr[j] > arr[j + 1]) {**

**aux = arr[j];**

**arr[j] = arr[j + 1];**

**arr[j + 1] = aux;**

**}**

**}**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**void sortare\_insertion1Dcres(int \*\*a, int \*arr, int n) {**

**int key, j;**

**for (int i = 1; i < n; i++) {**

**key = arr[i];**

**j = i - 1;**

**while (j >= 0 && arr[j] > key) {**

**arr[j + 1] = arr[j];**

**j = j - 1;**

**}**

**arr[j + 1] = key;**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**void sortare\_selection1Dcres(int \*\*a, int \*arr, int n) {**

**int minIndex, temp;**

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

**minIndex = i;**

**for (int j = i + 1; j < n; ++j) {**

**if (arr[j] < arr[minIndex]) {**

**minIndex = j;**

**temp = arr[minIndex];**

**arr[minIndex] = arr[i];**

**arr[i] = temp;**

**}**

**}**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**int sumaCifrelor(int numar) {**

**int suma = 0;**

**while (numar != 0) {**

**suma += numar % 10;**

**numar /= 10;**

**}**

**return suma;**

**}**

**void fornew1D(int \*\*a, int \*arr, int n) {**

**int index = 0, k;**

**for (int i = 0; i < n; ++i) {**

**for (int j = 0; j < n; ++j) {**

**if (sumaCifrelor(a[i][j]) == k) {**

**arr[index++] = a[i][j];**

**}**

**}**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**void freemem2D(int \*\*a, int \*arr, int n) {**

**for (int i = 0; i < n; i++)**

**free(a[i]);**

**free(a);**

**}**

**void afisare1D(int \*\*a, int \*arr, int n) {**

**for (int i = 0; i < n; i++) {**

**printf("%i", arr[i]);**

**}**

**printf("\n");**

**meniu(a, arr, n);**

**}**

**void freemem1D(int \*\*a, int \*arr, int n) {**

**free(arr);**

**}**

**void sortare\_bubble1Ddesc(int \*\*a, int \*arr, int n) {**

**int aux;**

**for (int i = 0; i < n; i++) {**

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

**if (arr[j] < arr[j + 1]) {**

**aux = arr[j];**

**arr[j] = arr[j + 1];**

**arr[j + 1] = aux;**

**}**

**}**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**void sortare\_selection1Ddesc(int \*\*a, int \*arr, int n) {**

**int maxIndex, temp;**

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

**maxIndex = i;**

**for (int j = i + 1; j < n; ++j) {**

**if (arr[j] > arr[maxIndex]) {**

**maxIndex = j;**

**temp = arr[maxIndex];**

**arr[maxIndex] = arr[i];**

**arr[i] = temp;**

**}**

**}**

**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**void sortare\_insertion1Ddesc(int \*\*a, int \*arr, int n) {**

**int key, j;**

**for (int i = 1; i < n; i++) {**

**key = arr[i];**

**j = i - 1;**

**while (j >= 0 && arr[j] < key) {**

**arr[j + 1] = arr[j];**

**j = j - 1;**

**}**

**arr[j + 1] = key;**

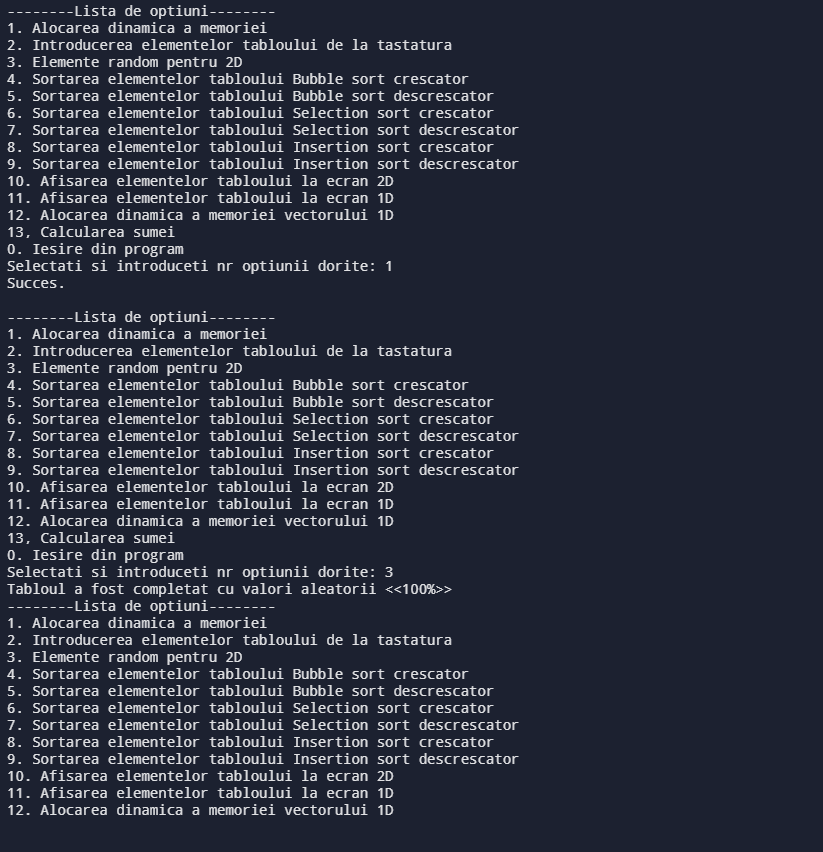
**}**

**afisare1D(a, arr, n);**

**meniu(a, arr, n);**

**}**

**Rezultatele:**

****

# Concluzie:

În cadrul acestui laborator, am abordat programarea algoritmilor de prelucrare a tablourilor 1D și 2D în limbajul C, concentrându-ne pe utilizarea funcțiilor, pointerilor și alocării dinamice a memoriei. Scopul principal al lucrării a fost să implementăm un program C care să ofere un meniu recursiv și să acopere funcțiile esențiale pentru manipularea tablourilor.

Pentru atingerea obiectivului, am implementat următoarele funcții:

Alocarea dinamică a memoriei: Această funcție ne-a permis să alocăm spațiu în memorie pentru tablourile noastre, asigurând flexibilitate și eficiență în gestionarea resurselor.

Introducerea valorilor tabloului de la tastatură: Am creat o funcție care permite utilizatorului să introducă manual valorile tabloului de la tastatură, facilitând personalizarea conținutului acestuia.

Completarea tabloului cu valori random: Prin această funcție, am introdus posibilitatea de a umple tabloul cu valori aleatorii, adăugând un element de variabilitate și testare în program.

Sortarea elementelor tabloului: Am implementat algoritmi de sortare esențiali, respectiv Bubble Sort, Selection Sort și Insertion Sort. Acești algoritmi permit ordonarea eficientă a elementelor din tablou, asigurând astfel o manipulare eficientă a datelor.

Eliberarea memoriei și ieșirea din program: O funcție crucială pentru gestionarea resurselor, aceasta se ocupă de eliberarea spațiului alocat dinamic în memorie și permite o terminare controlată a programului.

Prin combinarea acestor funcționalități, am creat un program C robust și modular, capabil să gestioneze operațiuni diverse asupra tablourilor. Acest proiect ne-a oferit o perspectivă detaliată asupra programării în limbajul C, concentrându-ne pe aspecte precum alocarea dinamică a memoriei și implementarea algoritmilor de sortare.