

**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.6**

***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: 3](#_Toc164453083)

[Sarcina 1 4](#_Toc164453084)

[Concluzie: 10](#_Toc164453085)

Scopul lucrării:

Obținerea deprinderilor practice de implementare și de utilizare a tipului abstract de date „Stack” și „Queue” în limbajul C cu asigurarea operațiilor de prelucrare de bază.

Sarcina:

Să se scrie un program în limbajul C pentru implementarea și utilizarea tipului abstract de date „Stivă” și „Șir în așteptare” cu asigurarea operațiilor de prelucrare de bază. Funcția mаin() va afișa la ecran următorul meniu de opțiuni de bază:

1. Crearea unei stive dinamice

2. Citirea datelor referitoare la elementele stivei de la tastatură.

3. Afișarea datelor stivei într-un fișier stiva.txt/stiva.bin.

4. Căutarea elementului maximal din stivă după un câmp numeric.

5. Copierea datelor din stivă într-un șir în așteptare (queue).

6. Afișarea datelor șirului în așteptare într-un fișier queue.txt/queue.bin.

7. Determinarea lungimii stivei (numărul de elemente).

8. Modificarea câmpurilor unui element din stivă.

9. Afișarea(Adăugarea) datelor modificate a elementului din stivă la sfârșitul fișierului stiva.txt/stiva.bin.

10. Eliberarea memoriei alocate pentru stivă.

11. Eliberarea memoriei alocate pentru Queue. 0. Ieșire din program.

5. Structura Capitală cu câmpurile: denumirea, țară, populația, suprafața, primarul.

# Sarcina 1

**Codul:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <string.h>**

**#define MAX\_SIZE 100**

**// Structura pentru un element din stivă (Capitala)**

**typedef struct {**

**char denumirea[50];**

**char tara[50];**

**int populatie;**

**float suprafata;**

**char primarul[50];**

**} Capitala;**

**// Structura pentru stivă**

**typedef struct {**

**Capitala \*elements[MAX\_SIZE];**

**int top;**

**} Stiva;**

**// Initializarea stivei**

**Stiva\* initStiva() {**

**Stiva \*stack = (Stiva\*)malloc(sizeof(Stiva));**

**stack->top = -1;**

**return stack;**

**}**

**// Verificare dacă stiva este plină**

**int isFull(Stiva \*stack) {**

**return stack->top == MAX\_SIZE - 1;**

**}**

**// Verificare dacă stiva este goală**

**int isEmpty(Stiva \*stack) {**

**return stack->top == -1;**

**}**

**// Funcție pentru a pune un element în stivă**

**void push(Stiva \*stack, Capitala \*capitala) {**

**if (isFull(stack)) {**

**printf("Stiva este plina.\n");**

**return;**

**}**

**stack->elements[++stack->top] = capitala;**

**}**

**// Funcție pentru a scoate un element din stivă**

**Capitala\* pop(Stiva \*stack) {**

**if (isEmpty(stack)) {**

**printf("Stiva este goala.\n");**

**return NULL;**

**}**

**return stack->elements[stack->top--];**

**}**

**// Funcție pentru a citi o capitală de la tastatură**

**Capitala\* citesteCapitala() {**

**Capitala \*capitala = (Capitala\*)malloc(sizeof(Capitala));**

**printf("Introduceti denumirea: ");**

**scanf("%s", capitala->denumirea);**

**printf("Introduceti tara: ");**

**scanf("%s", capitala->tara);**

**printf("Introduceti populatia: ");**

**scanf("%d", &capitala->populatie);**

**printf("Introduceti suprafata: ");**

**scanf("%f", &capitala->suprafata);**

**printf("Introduceti numele primarului: ");**

**scanf("%s", capitala->primarul);**

**return capitala;**

**}**

**// Funcție pentru a căuta elementul maximal după un câmp numeric (populația)**

**Capitala\* cautareMaxim(Stiva \*stack) {**

**if (isEmpty(stack)) {**

**printf("Stiva este goala.\n");**

**return NULL;**

**}**

**Capitala \*maxCapitala = stack->elements[0];**

**for (int i = 1; i <= stack->top; i++) {**

**if (stack->elements[i]->populatie > maxCapitala->populatie) {**

**maxCapitala = stack->elements[i];**

**}**

**}**

**return maxCapitala;**

**}**

**// Funcție pentru a determina lungimea stivei (numărul de elemente)**

**int lungimeStiva(Stiva \*stack) {**

**return stack->top + 1;**

**}**

**// Funcție pentru a afișa o capitală**

**void afisareCapitala(Capitala \*capitala) {**

**printf("Denumire: %s\n", capitala->denumirea);**

**printf("Tara: %s\n", capitala->tara);**

**printf("Populatie: %d\n", capitala->populatie);**

**printf("Suprafata: %.2f\n", capitala->suprafata);**

**printf("Primar: %s\n", capitala->primarul);**

**}**

**// Funcție pentru eliberarea memoriei alocate pentru o capitală**

**void elibereazaCapitala(Capitala \*capitala) {**

**free(capitala);**

**}**

**// Funcție pentru eliberarea memoriei alocate pentru stivă**

**void elibereazaStiva(Stiva \*stack) {**

**while (!isEmpty(stack)) {**

**elibereazaCapitala(pop(stack));**

**}**

**free(stack);**

**}**

**int main() {**

**Stiva \*stack = initStiva();**

**int optiune;**

**do {**

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

**printf("1. Adaugare capitala in stiva\n");**

**printf("2. Afisare capitala maxima\n");**

**printf("3. Afisare lungime stiva\n");**

**printf("4. Eliberare memorie stiva\n");**

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

**printf("Introduceti optiunea: ");**

**scanf("%d", &optiune);**

**switch (optiune) {**

**case 1: {**

**Capitala \*nouaCapitala = citesteCapitala();**

**push(stack, nouaCapitala);**

**break;**

**}**

**case 2: {**

**Capitala \*maxCapitala = cautareMaxim(stack);**

**if (maxCapitala != NULL) {**

**printf("\nCapitala cu cea mai mare populatie este:\n");**

**afisareCapitala(maxCapitala);**

**}**

**break;**

**}**

**case 3: {**

**printf("\nLungimea stivei este: %d\n", lungimeStiva(stack));**

**break;**

**}**

**case 4: {**

**elibereazaStiva(stack);**

**printf("\nMemoria alocata pentru stiva a fost eliberata.\n");**

**break;**

**}**

**case 0:**

**printf("Programul s-a incheiat.\n");**

**break;**

**default:**

**printf("Optiune invalida. Va rugam sa reintroduceti.\n");**

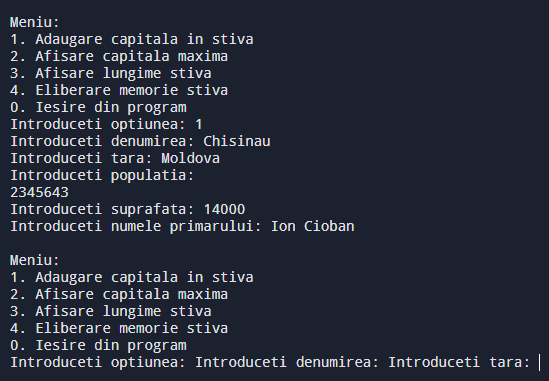
**}**

**} while (optiune != 0);**

**return 0;**

**}**

**Rezultatele:**

****

# Concluzie:

Implementarea și utilizarea tipurilor abstracte de date „Stivă” și „Șir în așteptare” în limbajul C este esențială pentru gestionarea eficientă a datelor în diverse aplicații. Aceste structuri oferă modalități flexibile și eficiente de stocare și manipulare a datelor, bazate pe principii bine definite precum LIFO și FIFO. Prin intermediul funcțiilor de bază precum push, pop, enqueue și dequeue, programatorii pot adăuga, elimina și accesa elementele într-un mod clar și coerent. Gestionarea memoriei este o preocupare majoră în implementarea acestor structuri, deoarece este necesară eliberarea memoriei alocate pentru a preveni scurgerile de memorie și pentru a menține performanța aplicației. Înțelegerea conceptelor subiacente și a modului în care aceste structuri de date funcționează este fundamentală pentru dezvoltarea de programe fiabile și eficiente. Utilizarea lor într-un mediu real poate simplifica și optimiza procesele de programare și poate contribui la dezvoltarea unor aplicații mai robuste și mai scalabile. În plus, capacitatea de a gestiona datele într-un mod organizat și coerent poate îmbunătăți în mod semnificativ performanța și fiabilitatea aplicațiilor software. În concluzie, învățarea și practicarea utilizării acestor tipuri abstracte de date reprezintă un pas important în evoluția competențelor de programare și în dezvoltarea de soluții software de calitate.