POINTER LA POINTER.

Dacă vom considera că **T1** este un tip oarecare de date, în loc de:

```
T1 tab[100];
```

putem folosi în program

```
T1* tab;
```

cu alocarea de memorie corespunzătoare:

```
tab=(T1*)aloca1d(100*sizeof(T1));
```

Dacă, aşa cum am considerat şi în lucrarea anterioară, **T1** este definit prin **typedef** ca un pointer la **T**, adică prin definiţia de tip:

```
typedef T* T1;
```

atunci vom avea, de fapt, declaraţia:

```
T** tab;
```

Adică, un pointer la pointer la T.

În acest caz, trebuie să facem alocare dinamică atât pentru tabloul de pointeri (pe care l-am înlocuit cu un pointer), cât şi pentru fiecare element al tabloului (care este un pointer).

După ce am terminat de lucrat cu un anumit pointer, trebuie să eliberăm zona de memorie alocată pentru fiecare pointer (ca element al unui tablou) şi pentru tabloul de pointeri în ansamblul său.

Un pointer la pointer poate fi echivalat cu un tablou bidimensional (o matrice). Atunci când ne referim la matrice alocate dinamic, ne referim de fapt la tipul de dată pointer la pointer la **T** (**T** este tipul elementelor matricei).

TEMA 1

Problema nr. 1.1

Să se scrie un program care citește de la tastatură o matrice alocată dinamic și o scrie în fișierul **matrice.out**.

Problema nr. 1.2

Să se acrie un prgram care citește de la tastatură două matrice alocate dinamic, le afișează pe monitor și apoi calculează suma celor două matrice și afișează matricea rezultată.

Programul va fi scris în două variante:

Prima variantă: funcția de citire primește ca parametri doi intregi și un pointer la pointer și nu returnează nimic.

A doua variantă: funcția de citire primește ca parametri doi întregi și returnează un pointer la pointer.

În ambele cazuri, funcția de calcul a sumei primește ca parametri doi pointeri la pointer și doi întregi și returnează un pointer la pointer.

TEMA 2

Problema nr. 2.1

Se dă o matrice reală, de dimensiune **n** x **m**, care se citeşte din fişierul **in.txt** şi pentru care se face alocare dinamică. Să se ordoneze crescător liniile matricei luând în considerare elementul maxim al liniei.

Date de test:

Matricea care trebuie ordonată:

2	5	6
1	1	0
3	1	–3

Matricea ordonată:

1	1	0
3	1	-3
2	5	6

Programul se va rezolva prin intermediul unui proiect şi se vor scrie funcţii pentru: alocare cu verificarea alocării (funcţia **aloca1d** din curs), alocare dinamică pentru o matrice (privită ca pointer la pointer), dealocare memorie pentru matrice, citire matrice, afişare matrice, ordonare matrice, determinare element maxim de pe o linie, interschimbare a două linii. Toate funcţiile vor folosi operaţii de lucru cu pointeri.

Funcţia de interschimbare linii are prototipul:

```
void swap(TIP **l1, TIP **l2);
```

unde TIP este tipul elementelor matricei (stabilit de utilizator).

Apelul funcţiei **swap** se face prin instrucţiunea:

```
swap(&a[i], &a[i+1])
```

dacă matricea considerată este **a**.

Problema nr. 2.2

Să se definească tipul de dată MATRICE (asociat unei matrice cu elemente reale) ca o structură care cuprinde:

- numele matricei memorat prin intermediul unui pointer (numele este format din mai multe caractere);
- numărul de linii şi numărul de coloane care sunt numere întregi fără semn;
- elementele matricei (de tip real) stocate într-o zonă de memorie alocată dinamic (printr-un pointer la pointer)

Să se scrie un program care pentru o matrice

1. citeşte de la tastatură numele matricei (unul sau mai multe caractere);
2. stochează toate informaţiile referitoare la matrice într-o structură de tip MATRICE, citind de la tastatură numărul de linii şi de coloane şi elementele matricei;
3. afişează pe linii matricea citită (fiecare element cu câte 3 zecimale);
4. afişează un meniu care dă posibilitatea utilizatorului să aleagă una din următoarele prelucrări:

a) ordonarea matricei astfel încât elementele de pe diagonala principală să fie în secvenţă crescătoare

b) ordonarea matricei astfel încât elementele de pe diagonala secundară să fie în secvenţă descrescătoare

c) ordonarea matricei astfel încât elementele de pe linia mediană verticală să fie în secvenţă crescătoare (în cazul în care matricea are un număr impar de coloane).

Programul poate face o singură prelucrare în funcţie de opţiunea utilizatorului şi va trebui scris astfel încât să permită prelucrarea mai multor seturi de date.

Observaţii:

1) Punctajul maxim se acordă pentru rezolvarea **CORECTĂ** a fiecărei subprobleme.

Barem de notare

1.Încărcarea informaţiilor într-o structură de tip MATRICE (funcţia primeşte ca parametru numele matricei – un pointer şi returnează o structură de tip MATRICE)	1,0
2. Alocarea corectă de spaţiu de memorie pentru pointeri (nume şi matrice)	1,0
3. Citirea elementelor matricei (într-o funcţie care primeşte ca parametri două numere naturale şi returnează un pointer la pointer la real)	1,0
4. Afişarea pe monitor a matricei citite (funcţia are un parametru – un pointer la o structură de tip MATRICE şi nu returnează nimic)	0,5
5. Scrierea meniului de prelucrare (funcţia nu are nici un parametru şi returnează un număr care reprezintă numărul opţiunii de prelucrare)	0,5
6. posibilitatea de reluare a programului (prelucrarea mai multor seturi de date)	0,5
7. Folosire proiect (corect)	0,5
8. Fişier header (corect şi complet)	0,4
8a. Definirea corectă a structurii de tip MATRICE	0,1
9. Funcţia main (complet – inclusiv eliberarea corectă a zonelor de memorie folosite)	1,0
10. Interschimbarea valorilor a doi vectori de reali (funcţia primeşte ca parametri doi pointeri la pointer la real şi nu returnează nimic) – funcţie folosită în funcţiile de ordonare	0,5
11. Ordonarea (prin metoda bulelor) după diagonala principală (funcţia primeşte ca parametru un pointer la o structură de tip MATRICE şi nu returnează nimic)	1,0
12. Ordonarea (prin metoda bulelor) după diagonala secundară (funcţia primeşte ca parametru un pointer la o structură de tip MATRICE şi nu returnează nimic)	1,0
13. Ordonarea (prin metoda bulelor) după linia mediană verticală (funcţia primeşte ca parametru un pointer la o structură de tip MATRICE şi nu returnează nimic)	1,0
TOTAL TABEL	**10** p