



## Tema 4 – Pregătire Examen

### Responsabili

- Marius Vintila
- Marius Iftimie
- George Muraru

- Data publicare: **5 Noiembrie 2019, ora 22:20**
- Deadline: **6 Ianuarie 2020, ora 23:55**

### Depunctări

Deadline-ul hard coincide cu cel soft.

În consecință, deadline-ul hard este **06.01.2020, ora 23:55**.

### Întrebări

Dacă aveți nelămuriri, puteți să ne contactați pe forumul dedicat temei de casă nr. 4.

La orice întrebare vom răspunde în maxim 24 de ore.

Nu se acceptă întrebări în ultimele 24 de ore înainte de deadline.

### Updates

- 15.12.2019: Update teste
- 3.1.2020: Update teste v2

### Obiective

- să se respecte formate stricte de intrare/ieșire
- să se însușească cunoștințele necesare pentru examen
- sa se înțeleagă si sa se utilizeze operații cu array-uri de pointeri
- sa se intealeaga si sa se utilizeze scrierea si citirea din fisiere binare

### Introducere

#### Resurse generale

- Regulament: seria CA
- Regulament: seria CB/CD
- Calendar
- Catalog laborator
- Debugging
- Coding style - CA
- Configuratie vmchecker - CA
- Test practic - CA
- Checker laborator CB/CD

#### Cursuri

Continutul Tematic

#### Laboratoare

01. Unelte de programare
02. Tipuri de date. Operatori.
03. Instrucțiunile limbajului C
04. Funcții
05. Tablouri. Particularizare - vectori
06. Matrice. Operații cu matrice
07. Optimizarea programelor folosind operații pe biți

Gigel vrea să se pregătească pentru examenul de la materia Programarea Calculatoarelor așa că a intrat pe drive-ul seriei și a ales un subiect de acolo. După ce a citit subiectul, și-a dat seama că ar trebui să se încălzească cu niște exerciții mai simple înainte de a trece la rezolvarea lui, iar treaba voastră este să-l ajutați în rezolvarea acestora.

## Cerința



**TOATE** modificările de cod se vor realiza **DOAR** în fișierele **project.c** și **project.h**. Modificarea oricărui alt fișier nu va fi luată în considerare.

Urmăriți indicațiile din fișierul **project.h** și completați funcțiile notate cu **TODO**. În fișierul **project.c** se pot implementa și funcții adiționale.

## Încălzire

### Task 1 (5p)

Se dă un vector cu  $n$  elemente numere întregi. Se cere întoarcerea unui nou vector, alocat dinamic, care să aibă pe poziția  $i$  produsul tuturor elementelor din vectorul primit, cu excepția elementului de pe poziția  $i$ .



Veți completa funcția `arrayProduct` din fișierul `project.c`

Exemplu:

Input:

```
4
1 2 3 4
```

Output:

```
24 12 8 6
```

### Task 2 (5p)

Se dă o matrice pătratică cu numere întregi. Să se rotească matricea cu 90 grade la stânga și să se întoarcă noua matrice,

- 08. Pointeri. Abordarea lucrului cu tablouri folosind pointeri
- 09. Alocarea dinamică a memoriei. Aplicații folosind tablouri și matrice
- 10. Prelucrarea șirurilor de caractere. Funcții. Aplicații
- 11. Structuri. Uniuni. Aplicație: Matrice rare
- 12. Operații cu fișiere. Aplicații folosind fișiere.
- 13. Parametrii liniei de comandă. Preprocesorul. Funcții cu număr variabil de parametri
- 14. Recapitulare

### Teme de casă (general)

- Indicații generale

### Teme de casă: seria CA

#### Teme CB/CD

- Tema 1
- Tema 2
- Tema 3
- Tema 4

### Table of Contents

- Tema 4 – Pregătire Examen
  - Responsabili
  - Obiective
  - Introducere
  - Cerința
    - Încălzire

alocată dinamic.



Veți completa funcția `rotateMatrix` din fișierul `project.c`

Exemplu:

Input:

```
3
1 2 3
4 5 6
7 8 9
```

Output:

```
3 6 9
2 5 8
1 4 7
```

### Task 3 (5p)

Se dă o matrice având dimensiunile  $n$  și  $m$ , cu elemente întregi și  $k$  query-uri de forma  $\langle x1, y1, x2, y2 \rangle$ , unde  $x1, y1, x2, y2$  sunt numere întregi ce reprezintă coordonate ale matricei. Se cere calcularea sumei elementelor din submatricea definită de cele 2 coordonate.



Veți completa funcția `subMatrixesSums` din fișierul `project.c`.

$0 \leq x1, x2 < nrLinii$ ;  $0 \leq y1, y2 < nrColoane$

Exemplu:

Input:

```
3 4
1 2 3 4
5 6 7 8
9 10 11 12
3
1 2 2 2
2 2 2 3
0 0 2 2
```

- Task 1 (5p)
- Task 2 (5p)
- Task 3 (5p)
- Rezolvarea Subiectului
  - Task 4 (5p)
    - Struc
    - Aloca struc
  - Task 5 (5p)
    - Citire datel
    - Afișar datel
  - Task 6 (5p)
  - Task 7 (5p)
  - Task 8 (5p)
- Punctaj
- Listă depunctări
- Trimitere temă

Output:

```
18 23 54
```

## Rezolvarea Subiectului

Pentru subiectul pe care l-a ales, Gigel are de implementat o parte din sistemul dezvoltat de către compania de transport Uber.

### Task 4 (5p)

#### Structuri

Pentru a implementa acest sistem, Gigel trebuie sa declare următoarele structuri:

- Structura pentru sofer, ce va conține următoarele câmpuri:
  - nume, reprezentat ca șir de caractere ce va conține maxim 20 de caractere utile;
  - număr mașină, șir de caractere format din 8 caractere utile (de forma JJNNNLLL; J - Județ, N - Număr, L - Litera);
  - locația curentă pe hartă, salvată ca două numere reale cu precizie dublă;
  - numărul de curse efectuate de către sofer, întreg fără semn;
  - informații despre toate cursele efectuate, vector de adrese pe structura de mai jos.
- Structura pentru cursa, ce va avea următoarele câmpuri:
  - locația de la începutul cursei, salvată ca două numere reale cu precizie dublă;
  - starea cursei reprezentată pe un byte, în care se retine: cursă anulată, cursă în desfășurare, număr stele (0→5)



Veți completa structurile din fișierul project.h

#### Alocarea structurilor

Mai departe, el trebuie sa implementeze doua funcții pentru alocarea dinamica a structurilor, una pentru alocarea unui sofer ce primește un număr de curse, iar

cealaltă pentru alocarea unui vector de șoferi ce primește numărul de șoferi și numărul de curse pentru fiecare șofer într-un vector.



Veți completa funcțiile `allocDriver` și `allocDrivers` din fișierul `project.c`



**ATENȚIE!** Atât vectorul de șoferi cât și vectorul de curse al fiecărui șofer sunt **vectori de referințe!!!**

## Task 5 (5p)

### Citirea datelor

Pentru a-și putea pune în practică sistemul, Gigel trebuie să citească datele șoferilor, ce sunt salvate în fișiere binare. Astfel, el are de completat o funcție care primește un fișier binar deja deschis pentru citire și din care trebuie să citească, respectând următoarea ordine:

- un întreg, reprezentând numărul de șoferi;
- un vector de numere întregi, având lungimea numărului de șoferi și reprezentând numărul de curse pentru fiecare șofer;
- apoi datele fiecărui șofer:
  - numele;
  - numărul mașinii;
  - locația curentă pe hartă;
  - și detaliile pentru fiecare cursă a acestuia:
    - locația de start;
    - starea.

Funcția va returna vectorul de șoferi, iar numărul de șoferi va fi returnat prin intermediul unui parametru.

### Afișarea datelor

Din motive de verificare ale datelor, Gigel are de implementat și o funcție pentru afișarea unui vector de șoferi. Acesta va trebui să afișeze într-un fișier binar deschis pentru scriere toate datele șoferilor, în aceeași ordine ca și în cazul citirii:

- numele;
- numărul mașinii;

- locația curentă pe hartă;
- și detaliile pentru fiecare cursă a acestuia:
  - locația de start;
  - starea.



Veți completa funcțiile `readDrivers` și `printDrivers` din fișierul `project.c`

## Task 6 (5p)

În acest moment, Gigel, fiind puțin curios, dorește să vadă care este angajatul cu cel mai bun rating mediu (sumă număr stele / (număr curse total - anulate - în desfășurare)). Pentru acest lucru, el are de implementat o funcție care primește vectorul de șoferi și întoarce numele șoferului cu cel mai bun rating.



Veți completa funcția `maxRatingDriverName` din fișierul `project.c`

## Task 7 (5p)

În acest moment, lui Gigel îi mai lipsește un singur lucru de la soluția lui: algoritmul de găsim al celor mai apropiați șoferi de o locație de start, așa că voi îl veți ajuta să completeze aceasta funcție. Funcția primește vectorul de șoferi, locația de start și numărul de șoferi care vor fi căutați, prin comparare cu distanța euclidiană.



Veți completa funcția `getClosestDrivers` din fișierul `project.c`

## Task 8 (5p)

Pentru a-și finaliza acest proiect, Gigel trebuie să elimine datele șoferilor din memorie, astfel încât el are de implementat două funcții: o funcție care eliberează memoria ocupată de către un șofer, iar alta care eliberează memoria ocupată de un vector de șoferi.



Veți completa funcțiile `freeDriver` și `freeDrivers`





din fișierul `project.c`

## Punctaj


- **[40p]** Teste
- **[5p]** Fișier README în care să se descrie implementarea
- **[5p]** [Coding Style](#).


TOTAL: **50p**

## Listă depunctări


- o temă care nu compilează și nu a rulat pe  **v2.vmchecker** nu va fi luată în considerare
- o temă care nu rezolvă cerința și trece testele prin alte mijloace nu va fi luată în considerare
- [-1.0]: warning-uri la compilare (este obligatorie folosirea în fișierul **Makefile** a flag-ului de compilare **-Wall** pentru regula **build**)
- [-1.0]: linii mai lungi de 80 de caractere
- [-1.0]: funcții mai lungi de 100 de linii
- [-0.5]: folosirea de  **magic numbers**
- [-0.5]: numele variabilelor nu sunt sugestive
- [-0.5]: cod comentat
- [-0.5]: trailing whitespaces
  - în cadrul cursului de programare nu avem ca obiectiv rezolvarea în cel mai eficient mod posibil a programelor; totuși, ne dorim ca abordarea să nu fie una ineficientă, de genul să nu folosiți instrucțiuni repetitive acolo unde clar era cazul, etc.

## Trimitere temă

Tema va fi trimisă folosind  **V2.Vmchecker**, cursul **Programarea Calculatoarelor (CB & CD)**.

Găsiți arhiva cu checker-ul și scheletul temei  [aici](#).

Formatul arhivei care va fi încărcată pe platforma de testare a temelor va fi următorul:

1. fișierele `project.c`, `include/project.h`.
2. Un fișier  **README** în care vă descrieți rezolvarea fiecărui task.



1. Arhiva trebuie să fie de tipul **ZIP** și să conțină toate fișierele menționate **IN RADACINA**



ARCHIVEI.

programare/teme\_2019/tema4\_2019\_cbd.txt · Last modified: 2020/01/04 00:24 by marius.vintila

Old revisions

Media Manager

Back to top