Tema 2. Avioane

Responsabili

- Aldea George Dănuț
- Dabelea Ioana-Viviana
- Stoica Robert-Valentin

Termen de predare:

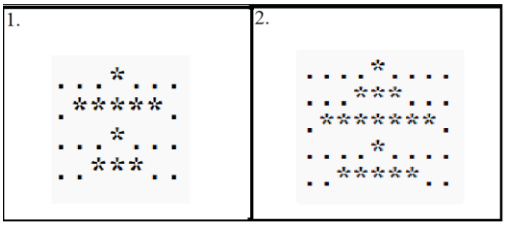
- Deadline soft: **07.12.2022 23:55**
- Deadline hard: **13.12.2022 23:55**

Introducere

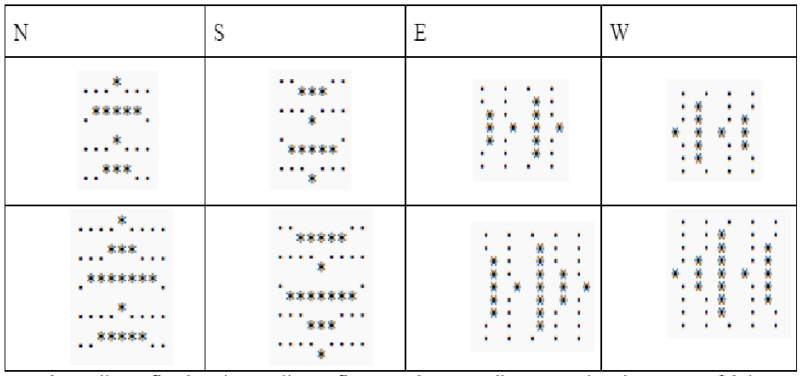
Andra şi Ştefan, doi studenţi la ACS, s-au hotărât să se joace faimosul joc de strategie, Avioanele. Fiind foarte competitivi din fire, aceştia au decis să complice puţin regulile jocului. Astfel, fiecare dintre ei o să deseneze **nr_avioane** avioane şi **nr_obstacole** obstacole, într-o matrice de dimensiune **N** * **N**.

Despre Avioane

Avioanele pot fi de 2 tipuri:



Ele pot avea capul îndreptat în 4 direcţii.



Ca lucrurile să fie şi mai complicate, fiecare avion o să aibă o anumită viteză **v**. Astfel, la un moment de timp **T**, un avion o să se deplaseze cu **v** * **T** unități în direcţia deja stabilită.

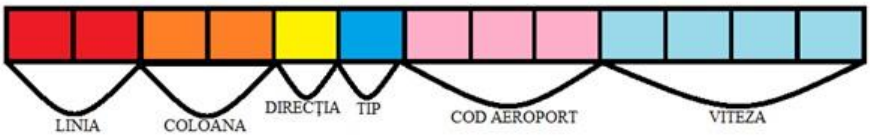
Despre Obstacole

Obstacolele sunt de dimensiune **1 x 1** şi sunt alese pentru a încurca adversarul.

Cerinţe

După ce fiecare dintre ei îşi desenează pe foaie avioanele, aceştia realizează că nu mai au timp să şi înceapă jocul. Ştiind că are o strategie bună, Ştefan doreşte să îşi păstrează informaţiile despre avioanele lui. Astfel, el stochează datele într-un vector generic **void *info**, păstrând pentru fiecare avion, în ordine, următoarele informaţii:

- linia, respectiv coloana pe care se găseşte capul avionului, păstrate pe câte **doi** octeţi **fiecare** (atât linia, cât şi coloana sunt numere naturale din intervalul [0, **N** - 1])
- direcţia spre care se deplasează avionul (N, S, E, W), păstrată pe **un** octet
- patru caractere care reprezintă codul avionului, astfel:
  - primul caracter reprezintă tipul avionului('1', '2')
  - următoarele trei caractere reprezintă codul aeroportului din care şi-au imaginat fiecare că o să zboare avionul
- viteza de deplasare (**v** poziţii/secundă), păstrată pe **patru** octeţi



Din păcate pentru el, Andra reușește să obţină acest vector, dar are nevoie de ajutorul vostru pentru a extrage informaţiile necesare.

Task 1 - 20 de puncte

```
void SolveTask1(void *info, int nr_avioane)
```

Funcţia primeşte ca parametri vectorul descris anterior, numărul de avioane și afișează pe ecran informațiile despre fiecare avion în ordinea următoare:

- pe prima linie 2 numere naturale care reprezintă coordonatele la care se găseşte capul avionului(linia, respectiv coloana) **ex.** : (1, 3)
- pe a doua linie un caracter care reprezintă direcţia de deplasare a avionului('N', 'S', 'E', 'V')
- pe a treia linie 4 caractere care reprezintă codul aeroportului
- pe ultima linie un număr natural care reprezintă viteza de deplasare a avionului

Între 2 avioane consecutive se va lăsa un rând gol.

```
Input:
```

```
1               // numărul task-ului

2               // numărul de avioane din vector


0 2 N           // linia, coloana şi direcţia primului avion

1BUC            // codul primului avion

1               // viteza primului avion


6 4 W

2AMS

2

Output:

(0, 2)

N

1BUC

1


(6, 4)

W

2AMS

2
```

Task 2 - 30 de puncte

```
void SolveTask2(void *info, int nr_avioane, int N, char **mat)
```

Funcţia primeşte ca parametri vectorul descris anterior, numărul de avioane, dimensiunea matricii de caractere **mat** care trebuie completată cu '*' sau '**.**' astfel: dacă la linia **i** şi coloana **j** se găseşte un avion, atunci mat[i][j] = '*', în caz contrar mat[i][j] = '.'.
Se garantează că toate avioanele se găsesc integral în interiorul matricii.

```
Input:

2
```

```
2


0 2 N

1BUC

1


6 4 W

2AMS

3


10              //valoarea lui N

Output:

..*.......

*****.....

..*.......

.***..*...

......*.*.

.....**.*.

....*****.

.....**.*.

......*.*.

......*...
```

Task 3 - 50 de puncte

```
void SolveTask3(void *info, int nr_avioane)
```

Funcţia primeşte ca parametri vectorul descris anterior, numărul de avioane şi sortează avioanele astfel:

- crescător după tipul avionului (1 sau 2)

- la tipuri egale, descrescător lexicografic după codul aeroportului(ex. BUD < OTP)
- la coduri egale, crescător după viteză

După ce au fost sortate, informaţiile despre avioane o să fie afişate la fel ca la Task-ul 1.

Sortarea o să se facă folosind **doar** funcţia **qsort**.

```
Input:

3

6


0 2 N

1BUC

1


6 4 W

2AMS

3


12 20 N

1CDB

5


13 25 S

1PRC

10


20 4 W

2BUC

7
```

```
45 54 E
1BUC
4
Output:
(13, 25)
S
1PRC
10

(12, 20)
N
1CDB
5

(0, 2)
N
1BUC
1

(45, 54)
E
1BUC
4

(20, 4)
W
2BUC
```

```
7



(6, 4)

W

2AMS

3
```

Task 4 - 50 de puncte

```
void SolveTask4(void *info, int nr_avioane, int nr_obstacole, int *x, int *y, int N)
```

Funcţia primeşte ca parametri vectorul descris anterior, numărul de avioane, numărul de obstacole puse de Andra, 2 vectori **x** şi **y** care păstrează linia, respectiv coloana unde se găsesc obstacolele şi dimensiunea matricii. Funcţia afişează numărul de avioane care ar putea să se deplaseze până la ieşirea de pe hartă în siguranţă (pe tot traseul nu au lovit niciun obstacol)

```
Input:

4

2



0 2 N

1BUC

1



6 4 W

2AMS

3



4          // numărul de obstacole

1 8      // linia şi coloana primului obstacol

5 1

7 7
```

```
9 3



10

Output:

1
```

Explicatie:

```
..*.......
*****...@.
..*.......
.***..*...
......*.*.
.@...**.*.
....*****.
.....**@*.
......*.*.
...@..*...
```

@ - locul unde se găsesc obstacolele

Bonus: Task 5 - 50 de puncte

```
void SolveTask5(void *info, int nr_avioane, int T, int nr_pct_coord, int *X, int *Y, int N)
```

Funcţia primeşte ca parametri vectorul descris anterior, numărul de avioane, un timp **T**, **nr_pct_coord** coordonate care sunt păstrate în vectorii **X** şi **Y** (X[0] şi Y[0] reprezintă linia, respectiv coloana primul obstacol) şi dimensiunea matricii. Funcţia afişează pentru fiecare moment de timp de la 0 la **T** câte avioane se află în punctele păstrate în cei 2 vectori descrişi anterior.

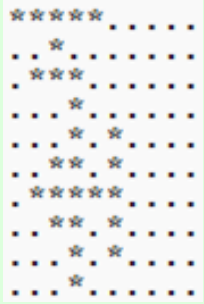
Un avion se mişcă cu **v** poziţii în fiecare moment de timp, în direcţia sa de deplasare.

```
Input:

5

2



0 2 N

1OTP

1
```

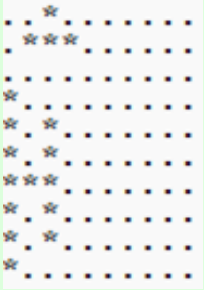
```
6 4 W

2LTN

3



2           // T


5           // nr_pct_coord

1 4

5 1

7 7

6 4

9 6



10

Output:

0: 2

1: 1

2: 0
```

Pentru **T** = 0 poziţia avioanelor o să fie:

Pentru **T** = 1 poziţia avioanelor o să fie:



Pentru **T** = 2 poziţia avioanelor o să fie:



Se poate observa că, dacă un avion se află în mai multe puncte date, acesta o să fie numărat o singură dată.

În cazul în care, la un moment de timp **T**, 2 avioane se află în acelaşi punct păstrat în cei 2 vectori, ambele se numără.

Pentru acest task **nu** se garantează că avioanele se află **complet** în matrice şi că avioanele nu se suprapun.

Precizări suplimentare

- **N** < = 100
- **nr_avioane** < = 50
- **nr_obstacole** < = 20
- **nr_pct_coord** < = 20
- **T** < = 50
- Pentru primele 4 task-uri se garantează faptul că avioanele se află complet în matrice.

Resurse

- Checker-ul, scheletul de cod şi testele o să se găsească în arhiva avioane.zip.
- Pentru a instala programele necesare verificării codingstyle-ului daţi, pe rand, comenzile:

        - **chmod +x install-linters.sh check.sh**
        - **./install-linters.sh**
    - In cazul in care exista erori la sectiunea "Run cpplint" sau "Run clang-tidy", checkerul va scădea **10** puncte din scorul total
    - Puteţi să creaţi şi alte funcţii pentru a vă ajuta la rezolvarea task-urilor.
    - Pentru rularea checkerului se va da comanda:
        - **./check.sh**
    - În cadrul arhivei o să mai găsiţi şi 3 directoare:
        - input: datele de intrare pentru fiecare task
        - output: rezultatele **voastre** în urma rulări checker-ului
        - ref: răspunsurile **corecte** pentru fiecare test

NU modificaţi codul din fişierele *intern.c* şi *tema2.c*
Rezolvările voastre o să fie făcute în cadrul fişierelor *task1.c*, *task2.c*, *task3.c*, *task4.c*, *task5.c* în interiorul funcţiilor existente deja.

Trimitere temă

Tema va fi trimisă folosind vmchecker, cursul **Programarea Calculatoarelor (CB & CD)**.
Toate temele sunt testate în mod automat pe VMChecker. Autentificarea se face folosind numele de utilizator și parola de pe moodle / Teams.
Din meniul *drop-down* selectați cursul corespunzător; în cazul de față: **Programarea Calculatoarelor (CB & CD)**. În meniul *didebar*, din partea stângă a paginii, selectați tema pentru care veți face submisia.
Arhiva temei se va încărca pe checker folosind formularul de submisie din tabul **Trimitere solutii**.
Rezultatele vor fi disponibile în tabul **Rezultate**.
Citiți cu atenție informațiile afișate în **Rezultate** pentru a vă asigura că tema a fost rulată cu succes; o eroare comună este dată de faptul că conținutul arhivei nu respectă structura dorită (ex. fișierele sunt într-un alt director).
Punctajul final al temei este afișat la finalul informațiilor afișate în **Rezultate**.
Conținutul arhivei va fi următorul:

1. Fișierele **task1.c**, **task2.c**, **task3.c**, **task4.c**, **task5.c**, **utils.h** care conține implementarea temei.
2. Fișierul **Makefile**.
3. Un fișier README în care descrieți rezolvarea temei.

1. Arhiva trebuie să fie de tipul **zip**.
2. Puteți utiliza regula **zip** din fișierul **Makefile** pentru a vă genera arhiva zip. Aceasta va adăuga fișierele Makefile, README și orice fișierele **task1.c**, **task2.c**, **task3.c**, **task4.c**, **task5.c**, **utils.h** din directorul curent.

```
make zip
```

Nu includeţi fisierele checkerului ăn arhiva voastră. **Nu folosiţi Makefile.checker** pe post de Makefile în arhiva voastrî: asta va duce la recursivitate infinita pe vmchecker. Puteti sa folosiţi direct makefile-ul prezent în arhivă (**Makefile, nu Makefile.checker**).

În cazul în care testele vă trec local, însă pică pe vmchecker cel mai probabil aveţi o sursă de "undefined behavior in cod". Pentru a vă asigura că scapati de aceste probleme, compilaţi cu flagul de compilare `-Wall` si rezolvaţi toate warning-urile.

Listă depunctări

Lista nu este exhaustivă.

- O temă care nu compilează și nu a rulat pe **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
- **NU acceptăm teme copiate**. În cazul unei teme copiate se scade punctajul aferent temei din punctajul total.
- [-20.0]: Nerezolvarea tuturor erorilor și warningurilor de coding style

Întrebări

Dacă aveți nelămuriri, puteți să ne contactați pe forum tema 2 sau pe canalul Tema 2. La orice întrebare vom răspunde în maxim 24 de ore. **Nu** se acceptă întrebări în ultimele 24 de ore înainte de deadline.