PROGRAMAREA CALCULATOARELOR ȘI LIMBAJE DE PROGRAMARE I
Tema #1 Funcții, Vectori și Matrici

Deadline soft: 12.11.2023 23:55. Deadline: 16.11.2023 23:55

Responsabili: Radu Nichita, Alex Deonise, Monica Bîrlădeanu, Andrei Crețț, Dominic Staicu, Andreea Drehuță, Gabriel Cărăuleanu

Contents

Problema 1 - Un produs infinit

Enunț

Gigel abia a început primul lui an la ACS și a dat de greu cu matematica, în special algebra. Deși participă activ la cursuri și seminarii, el nu reușește să se prindă cum se fac exercițiile cu vectori. Din păcate, acesta nu e prea bun nici la înmulțiri, dar se pricepe să scrie cod. Pentru că nu dorește să rămână în urmă nici cu programarea, alege să își rezolve problemele la matematică folosind limbajul C.

Tema lui curentă este de a calcula produsul scalar a doi vectori **a** și **b**. Fiind curios din fire, dar și pentru a-și îmbunătăți noțiunile de programare, Gigel se mai gândește la alte mărimi din matematică pe care le poate determina.

Se dau **N** perechi de numere, reprezentând coordonatele (a_i, b_i) a 2 vectori **a** și **b**. Gigel dorește să calculeze următoarele:

- **ps** = produsul scalar al vectorilor **a** și **b**
- **a_max** și **b_max** = al doilea maxim din vectorul **a**, respectiv **b**
- **n_a** și **n_b** = norma 2 pentru fiecare dintre vectorii **a** și **b**

Pentru a face lucrurile mai interesante, Gigel vine cu următoarea cerință pentru voi: citirea vectorilor **a** și **b** (în mod exclusiv) se va face având in vedere numere in **baza 8**.

Restricții și precizări

- $0 \leq N \leq 2^{32} - 1$
- $0 \leq a_i, b_i \leq 2^{32} - 1$
- dacă al doilea maxim nu există, se va afișa -1 pentru vectorul respectiv.
- se garantează că toate rezultatele intermediare sunt mai mici sau egale cu $2^{64} - 1$.

ATENȚIE! N poate fi foarte mare, astfel încât stocarea vectorilor nu este permisă. O rezolvare care reține valorile în 2 vectori a și b va obține maxim 2.66 puncte din 10 posibile, din cauza restricțiilor de memorie.

Date de intrare

Toate datele se vor citi de la **STDIN**.

```
N
a_0 b_0
a_1 b_1
a_2 b_2
...
a_{n - 1} b_{n -1}
```

Date de ieșire

Toate datele se vor afișa la **STDOUT**. Numerele reale se vor afișa cu **7 zecimale** exacte.

```
ps
a_max b_max
n_a n_b
```

Exemplu

Date de intrare

```
9
1 1
2 2
2 4
2 4
2 5
2 5
3 5
3 2
12 12
```

Date de ieșire

```
162
3 5
11.7898261 14.6969385
```

Explicație

1. vectorul A este format din tuplul (1, 2, 2, 2, 2, 2, 3, 3, 10) - elemente în baza 10.
2. vectorul B este format din tuplul (1, 2, 4, 4, 5, 5, 5, 2, 10) - elemente în baza 10.

Produsul scalar al vectorilor a și b este dat de formula:

$$a \cdot b = \sum_{i=1}^{9} a_i * b_i = (1*1 + 2*2 + 2*4 + ... + 12*12)_{(8)} = 162_{(10)}$$

Dacă sortăm coordonatele vectorilor A și B (ca numere) în ordine descrescătoare obținem:

- pentru vectorul A: (10, **3**, 3, 2, 2, 2, 2, 2, 1)
- pentru vectorul B: (10, **5**, 5, 5, 4, 4, 2, 2, 1)

Al doilea maxim pentru vectorul A este 3, în timp ce pentru vectorul B este 5.

Pentru a calcula norma 2 a unui vector n-dimensional **v** se va folosi formula:

$$\|v\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2}$$

Astfel, pentru vectorii a și b obținem:

$$\|a\|_2 = \sqrt{\sum_{i=1}^{9} a_i^2} = \sqrt{(1*1 + 2*2 + 2*2 + ... + 10*10)} = 11.789826$$

$$\|b\|_2 = \sqrt{\sum_{i=1}^{9} b_i^2} = \sqrt{(1*1 + 2*2 + 4*4 + ... + 10*10)} = 14.696938$$

Problema 2 - Perfecționarea punctajelor

Enunț

În Universitatea **CodeInVim**, bursele se acordă studenților cu cele mai multe puncte. Vă propunem să învățați să calculați câte puncte poate strânge un student.

Vom folosi un scenariu simplu în care presupunem că pe parcursul unui an un student are **N** materii. Fiecare materie **i** este caracterizată printr-o notă x_i (obținută de student) și un număr de credite c_i (care reflectă dificultatea materiei).

Punctajul îl vom defini astfel:

$$P = \sum_{i=0}^{N-1} (x_i * c_i) \tag{1}$$

Mihai tocmai și-a aflat toate notele din anul I și se gândește să păstreze la îndemână cursurile pentru materiile la care va merge la măriri, deoarece vrea neapărat să fie printre bursieri.

El se întreabă care este numărul minim **m** de materii (cu notă diferită de 10) la care ar fi trebuit să ia 10, astfel încât punctajul lui să fie mai mare sau egal cu un **p_min** dat.

Restricții și precizări

- $1 \leq N \leq 100$;
- $1 \leq x_i \leq 10$ (x_i întreg)
- $1 \leq c_i \leq 6$ (c_i întreg)

HINT: Ce ați alege dacă ați avea voie să modificați **o singură notă**?

Date de intrare Toate datele se vor citi de la **STDIN**.

```
n
x_0 x_1 x_2 ... x_{n-1}
c_0 c_1 c_2 ... c_{n-1}
p_min
```

Date de ieșire

Toate datele se vor afișa la **STDOUT**. Se va scrie numărul căutat sau **-1** dacă problema nu are soluție.

```
m
```

Exemplul 1

Date de intrare

```
5
10 10 5 7 5
1 2 5 6 4
118
```

Date de ieșire

```
1
```

Explicație

Punctajul obținut este **10 * 1 + 10 * 2 + 5 * 5 + 7 * 6 + 5 * 4 = 117** Se pot mări:

- 5 (pondere 5) la 10 => punctajul crește cu 5 * (10 - 5) = 25;
- 7 (pondere 6) la 10 => punctajul crește cu 6 * (10 - 7) = 18;
- 5 (pondere 4) la 10 => punctajul crește cu 4 * (10 - 5) = 20.

Întrucât putem alege una dintre aceste variante, rezultă că numărul minim de măriri de care avem nevoie pentru a obține cel puțin 118 puncte este 1.

Exemplul 2

Date de intrare

```
5
10 10 5 7 5
1 2 5 6 4
160
```

Date de ieșire

```
2
```

Explicație

- Punctajul obținut și notele ce se pot mări sunt aceleași note ca în primul exemplu (deoarece notele și numărul de credite sunt identice).
- În acest caz, o singură mărire nu este suficientă: scor maxim 117 + 25 = 142. De aceea mai este necesară încă o mărire care să asigure diferența rămasă. Răspuns **2**.

Problema 3 - Gigel și tabla de șah

Enunț

După ce a dovedit și statisticile în ceea ce privește numărul de litere, Gigel își propune să termine tema la PCLP în timp util, însă fratele său mai mic, Gigelinho, îi distrage atenția. Astfel, pentru a-l ține ocupat, Gigel pune bilete pe care sunt scrise numere întregi. Apoi, cu ajutorul unei piese speciale de șah, amplasată în colțul din stânga-sus al tablei, Gigelinho trebuie să mute piesa respectând regulile:

- Dacă piesa se află pe un pătrat alb, atunci aceasta va fi mutată de-a lungul liniei pe care se află, cu un număr de pătrate egal cu numărul scris pe bilet, astfel: dacă numărul este pozitiv, atunci piesa se va muta la dreapta, iar în caz contrar, piesa se va muta la stânga.
- Dacă piesa se află pe un pătrat negru, atunci aceasta va fi mutată de-a lungul coloanei pe care se află, cu un număr de pătrate egal cu numărul scris pe bilet, astfel: dacă numărul este pozitiv, atunci piesa se va muta în jos, iar în caz contrar, piesa se va muta în sus.
- În ambele situații, biletul este folosit doar o dată (în cazul în care piesa revine pe aceeași poziție, se oprește).

La final, Gigelinho trebuie să determine distanța totală parcursă de piesă, precum și coordonatele câmpului pe care se află piesa la momentul final (ca la jocul de șah - vezi figura 1).

8	a8	b8	c8	d8	e8	f8	g8	h8
7	a7	b7	c7	d7	e7	f7	g7	h7
6	a6	b6	c6	d6	e6	f6	g6	h6
5	a5	b5	c5	d5	e5	f5	g5	h5
4	a4	b4	c4	d4	e4	f4	g4	h4
3	a3	b3	c3	d3	e3	f3	g3	h3
2	a2	b2	c2	d2	e2	f2	g2	h2
1	a1	b1	c1	d1	e1	f1	g1	h1
	a	b	c	d	e	f	g	h

Figure 1: Coordonatele unui câmp pe tabla de șah 8 x 8.

Restricții și precizări

- $1 \leq n \leq 1000$;
- Pentru coloană, în cazul în care coloana nu poate fi reprezentată folosind un caracter, va fi folosită reprezentarea existentă în Microsoft Excel (după caracterul **Z**, va urma **AA** drept index de coloană).
- Se consideră că pătratul din colțul stânga-sus al tablei este alb.
- Se consideră că tabla are structură toroidală (în cazul în care piesa iese din tablă în urma unei mutări).
- $-2^{30} \leq num \leq 2^{30}$, oricare ar fi num un număr de pe bilet.

Date de intrare

Toate datele se vor citi de la **STDIN**.

```
n
a_00 a_01 a_02 ... a_0(n-1)
a_10 a_11 a_12 ... a_1(n-1)
...
a_(n-1)0 a_(n-1)1 a_(n-1)2 ... a_(n-1)(n-1)
```

Date de ieșire

Toate datele se vor afișa la **STDOUT**. Se vor afișa distanța parcursă de piesă și respectiv coordonatele finale ale acesteia, în formatul precizat.

```
distance
y_final x_final
```

Exemplul 1

Date de intrare

```
2
1 1
0 -1
```

Date de ieșire

```
3
1 A
```

Explicație

Piesa parcurge următoarele transformări:

$$a_{00}(+1/alb) \rightarrow a_{01}(+1/negru) \rightarrow a_{11}(-1/alb) \rightarrow a_{10}(0/negru) \rightarrow STOP$$

Exemplul 2

Date de intrare

```
4
1 2 -1 0
2 -1 0 -1
1 -1 1 -2
1 -1 1 0
```

Date de ieșire

```
8
4 A
```

Explicație

Piesa parcurge următoarele transformări:

$$a_{00}(+1/alb) \rightarrow a_{01}(+2/negru) \rightarrow a_{21}(-1/negru) \rightarrow a_{11}(-1/alb) \rightarrow a_{10}(+2/negru)$$

$$\rightarrow a_{30}(+1/negru) \rightarrow a_{00}(0/alb) \rightarrow STOP$$

- Distanța parcursă este $1 + 2 + 1 + 1 + 2 + 1 = \mathbf{8}$
- Poziția finală: linia din matrice 0, corespunzătoare liniei de pe tabla de șah **4** și coloana 0, corespunzătoare indexării cu **A**

Problema 4 - Nonogram checker

Enunț

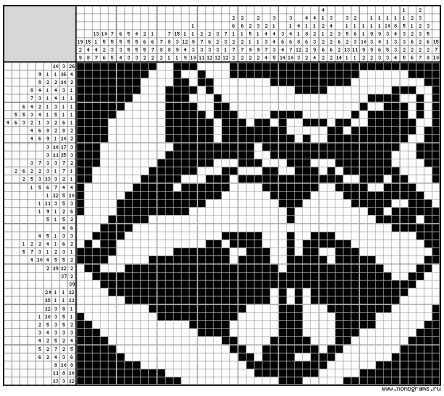

Figure 2: Grid de Nonogram.
Sursa imaginii: https://www.nonograms.org/nonograms/i/26645

Nonogram este un joc, similar Sudoku, care își propune completarea cu căsuțe albe și negre a unui grid de dimensiune variabilă (N linii x M coloane), pe baza unor restricții precizate pentru fiecare linie și coloană.

Astfel, pentru fiecare linie și fiecare coloană se precizează numărul de grupuri de căsuțe negre precum și dimensiunea acestora, în ordinea în care ele apar (de la stânga la dreapta pentru linii și de sus în jos pentru coloane). Două grupuri consecutive vor fi întotdeauna despărțite prin cel puțin o căsuță albă.

Gigel își dorește să creeze un program care să-l ajute să trișeze la acest joc, verificând pentru el dacă a completat corect un astfel de grid. El vă recomandă să încercați jocul, nu doar pentru că e fain, ci și pentru că așa înțelegeți cel mai ușor mecanismul de joc.

Restricții și precizări

- $1 \leq N, M \leq 100$
- Puteți juca Nonogram la adresa https://www.nonograms.org.

Date de intrare

Datele de intrare se vor citi de la **STDIN** altfel:

- Pe prima linie se va găsi un număr T, reprezentând numărul de puzzle-uri ce vor fi verificate;
- Pe următoarele linii puzzle-urile ce urmează a fi verificate.

Pentru fiecare puzzle:

- Pe prima linie, se vor găsi două numere naturale N și M reprezentând dimensiunile gridului;
- Pe următoarele N linii restricțiile pentru fiecare dintre linii, în ordinea crescătoare a indicelui liniei;
- Pe următoarele M linii restricțiile pentru fiecare dintre coloane, în ordinea crescătoare a indicelui coloanei;
  - Pentru o linie/coloană cu K grupuri colorate, restricțiile vor fi specificate prin intermediul a K+1 numere naturale reprezentând, în ordine: numărul de grupuri colorate de pe acea linie sau coloană (valoarea lui K), dimensiunea grupului pentru fiecare grup de căsuțe colorate.
- Pe următoarele N linii și M coloane, grid-ul completat de Gigel, dat ca o matrice de 1 și 0 în care o valoare de 0 reprezentă o căsuță albă, iar o valoare de 1 reprezintă o căsuță colorată.

Date de ieșire

Programul va afișa pentru fiecare puzzle la **STDOUT** pe câte o linie distinctă mesajul **Corect** dacă gridul este corect sau **Eroare** dacă gridul nu respectă restricțiile date.

Exemplul 1

Date de intrare

```
1
10 10
2 2 1
2 1 1
1 1
1 7
1 10
2 8 1
2 8 1
2 8 1
1 10
1 7
1 7
1 7
1 8
2 1 7
1 7
2 1 7
2 1 7
2 1 5
3 1 1 1
1 5
0 0 0 0 0 1 1 0 1 0
0 0 0 1 0 0 0 1 0 0
0 0 1 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 0 0
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 0 0 0
```

Date de ieșire

```
Corect
```

Explicație Datele de intrare reprezintă grid-ul din Figura 3, care este completat corect.

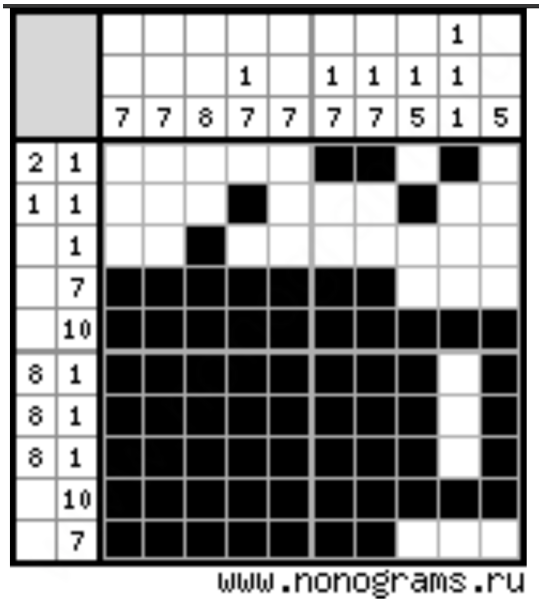

Figure 3: Grid de Nonogram din exemplul problemei 4.
Sursa imaginii: https://www.nonograms.org

Exemplul 2

Date de intrare

```
1
10 10
2 2 1
2 1 1
1 1
1 7
1 10
2 8 1
2 8 1
2 8 1
1 10
1 7
1 7
1 7
1 8
2 1 7
1 7
2 1 7
2 1 7
2 1 5
3 1 1 1
1 5
0 0 0 0 0 1 1 0 1 0
0 0 0 1 0 0 0 1 0 0
0 0 1 0 0 0 0 0 0 0
1 1 1 1 1 1 1 0 0 0
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 0 1
1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 0 0 1
```

Date de ieșire

```
Eroare
```

Explicație: Input-ul reprezintă același grid ca în Exemplul 1, dar cu o căsuță neagră în plus.

Regulament

Regulamentul general al temelor se găsește pe ocw (Temele de casă). Vă rugăm să îl citiți integral înainte de continua cu regulile specifice acestei teme.

Arhivă

Soluția temei se va trimite ca o arhivă **zip**. Numele arhivei trebuie să fie de forma **Grupă_NumePrenume_TemaX.zip** - exemplu: `311CA_NichitaRadu_Tema1.zip`.

Arhiva trebuie să conţină în directorul **RĂDĂCINĂ** doar următoarele:

- Codul sursă al programului vostru (fișierele **.c** și eventual **.h**).
- Un fișier **Makefile** care să conțină regulile **build** și **clean**.
  - Regula **build** va compila codul vostru și va genera următoarele executabile:
    - **infinite_product** pentru problema 1
    - **codeinvim** pentru problema 2
    - **gigel_and_the_checkboard** pentru problema 3
    - **nomogram** pentru problema 4
  - Regula **clean** va șterge **toate** fișierele generate la build (executabile, binare intermediare etc).
- Un fișier **README** care să conțină prezentarea implementării alese de voi. **NU** copiați bucăți din enunț.

Arhiva temei **NU** va conține: fișiere binare, fișiere de intrare/ieșire folosite de checker, checkerul, orice alt fișier care nu este cerut mai sus.

Numele şi extensiile fişierelor trimise **NU** trebuie să conţină spaţii sau majuscule, cu exceptia fișierului README (care are numele scris cu majuscule și nu are extensie).

Nerespectarea oricărei reguli din secțiunea **Arhivă** aduce un punctaj **NUL** pe temă.

Checker

Pentru corectarea aceste teme vom folosi scriptul **check** din arhiva **check_first_blood_remastered.zip** din secțiunea de resurse asociată temei. Vă rugăm să citiți **README.md** pentru a știi cum să instalați și utilizați checkerul.

Punctaj

Distribuirea punctajului:

- Problema 1: 10p
- Problema 2: 20p
- Problema 3: 25p
- Problema 4: 25p
- Claritatea și calitatea codului: 10p
- Claritatea explicaţiilor din README: 10p
- Modularizare + implementări deosebite : 10p (punctaj bonus, acordat la corectarea manuală)

ATENȚIE! Punctajul maxim pe temă este **100p**. Acesta reprezintă 0.5p din nota finală la această materie. La această temă se pot obține până la la **110p** (există un bonus de până la 10p acordat pe baza modularizării și a unor implementări deosebite, ce va fi acordat la corectarea manuală), adică un punctaj maxim de 0.55p din nota finală.

Reguli și precizări

- Punctajul pe teste este cel acordat de script-ul **check**, rulat pe **Moodle**. Echipa de corectare își rezervă dreptul de a depuncta pentru orice încercare de a trece testele fraudulos (de exemplu prin hardcodare).
- Punctajul pe calitatea explicațiilor și a codului se acordă în mai multe etape:
  - **corectare automată**
    - Checkerul va încerca să detecteze în mod automat probleme legate de coding style și alte aspecte de organizare a codului.
    - Acesta va puncta cu maxim 20p dacă nu sunt probleme detectate.
    - Punctajul se va acorda proporțional cu numărul de puncte acumulate pe teste din cele 80p.
    - Checkerul poate să aplice însă și penalizări (exemplu pentru warninguri la compilare) sau alte probleme descoperite la runtime.
  - **corectare manuală**
    - Tema va fi corectată manual și se vor verifica și alte aspecte pe care checkerul nu le poate prinde. Recomandăm să parcurgeți cu atenție tutorialul de coding-style de pe ocw.cs.pub.ro.
    - Codul sursă trebuie să fie însoțit de un fișier README care trebuie șă conțină informațiile utile pentru înțelegerea funcționalității, modului de implementare și utilizare a programului. Acesta evaluează, de asemenea, abilitatea voastră de a documenta complet și concis programele pe care le produceți și va fi evaluat, in mod analog CS, de către echipa de asistenți. In funcție de calitatea documentației, se vor aplica depunctări sau bonusuri.
    - La corectarea manuală se va acorda un bonus de maximum 10 puncte pentru modularizare.
    - Deprinderea de a scrie cod sursă de calitate, este un obiectiv important al materiei. Sursele greu de înteles, modularizate neadecvat sau care prezintă hardcodări care pot afecta semnificativ mentenabilitatea programului cerut, pot fi depunctate adițional.
    - În această etapă se pot aplica depunctări mai mari de 30p.
    - O temă care **NU** compilează cu -Wall -Wextra este depunctată la corectarea manuală cu 5p (punctajul echivalent pentru warnings).

Alte precizări

- Implementarea se va face in limbajul C, iar tema va fi compilată şi testată **DOAR** într-un mediu **LINUX**. Nerespectarea acestor reguli aduce un punctaj **NUL**.
- Tema trebuie trimisă sub forma unei arhive pe site-ul cursului curs.upb.ro.
- Tema poate fi submisă de oricâte ori fără depunctări până la deadline. Mai multe detalii se găsesc în regulamentul de pe ocw.
- O temă care **NU** compilează **NU** va fi punctată.
- O temă care compilează, dar care **NU** trece niciun test **NU** va fi punctată.
- Punctajul pe teste este cel acordat de **check** rulat pe **TODO**. Echipa de corectare își rezervă dreptul de a depuncta pentru orice încercare de a trece testele fraudulos (de exemplu prin hardcodare).
- Ultima temă submisă pe Moodle poate fi rulată de către responsabili de mai multe ori în vederea verificării faptului că nu aveți buguri în sursă. Vă recomandăm să verificați **local** tema de mai multe ori pentru a verifica că punctajul este mereu același, apoi să încărcați tema.