**Barem de notare pentru toate problemele:**

**Oficiu:**

* **1 punct (din oficiu)**
* **1 punct (fără erori de compilare)**
* **1 punct (definirea tuturor variabilelor)**
* **2 puncte (citirea corectă a datelor de intrare, validarea acestora în funcție de enunț)**
* **4 puncte (implementarea unui algoritm corect: prime, cifre, sortare, prelucrare caractere…)**
* **1 punct (afișarea rezultatelor)**

**Vectori:**

1. De la tastatură se citesc 2 numere a și b, (a<b), de maxim 9 cifre fiecare. Să se scrie în fișierul ***fibo.txt*** toate numerele din șirul lui Fibonacci care sunt în intervalul [a,b].
2. Fişierul text ***numere.txt*** conţine pe o singură linie, separate prin câte un spaţiu, cel mult 100 de numere întregi, fiecare număr având cel mult 4 cifre. Scrieţi un program care citeşte numerele din fişierul ***numere.txt*** şi afişează pe ecran, separate prin câte un spaţiu, în ordine crescătoare, toate numerele naturale nenule din fişier. Dacă nu există astfel de numere se va afişa pe ecran mesajul ***NU EXISTA***.
3. Se citeşte de la tastatură un număr natural n (n>=100). Să se determine dacă numărul dat este număr *„munte-vale”*. Un număr natural este „munte-vale” dacă cifrele din număr sunt în ordine crescătoare până la o anumită poziţie, iar cifrele care urmează sunt în ordine descrescătoare până la sfârşit. Exemple pentru numere *„munte-vale”*: 24521, 18942, 16432.
4. Se citesc de la tastatură două tablouri unidimensionale cu elementele în ordine crescătoare, a cu n elemente, respectiv b cu m elemente. Să se determine din cele două tablouri, un al treilea tablou unidimensional în mod optim, care să conţină toate elementele celor două tablouri tot în ordine crescătoare. Tabloul nou format va fi afișat în fișierul ***ordonat.txt***.
5. Se citesc de la tastatură două tablouri unidimensionale cu elementele în ordine crescătoare, a cu n elemente, respectiv b cu m elemente. Să se realizeze dacă sunt mulțimi, reuniunea și diferența lor. În cazul în care un tablou nu reprezintă o mulțime cu elemente distincte, se va afișa mesajul ”**X-nu e multime**”, unde X – este tabloul a sau b. Rezultatul va fi scris în fișierul ***multime.txt***.
6. Fişierul text ***interval.txt*** conţine pe prima linie un număr natural nenul n (1≤n≤1000), iar pe fiecare dintre următoarele n linii câte două numere întregi a şi b (1≤a≤b≤32000), fiecare pereche reprezentând un interval închis de forma [a,b]. Scrieţi un program care citeşte numerele din fişier şi determină un interval dintre cele citite care conţine cel mai mare număr de numere întregi şi afişează pe o linie a ecranului, separate printr-un spaţiu, numerele care reprezintă capetele intervalului determinat. În cazul în care sunt mai multe intervale care îndeplinesc această proprietate, se vor afişa informaţiile referitoare la acel interval la care numărul care reprezintă capătul din dreapta este minim.

Exemplu:

**4**

**17 24**

**-2 3**

**9 15**

**8 15**

Se va afișa **8 15.** [8,15] și [17,24] au același număr de elemente întregi, dar 8 e mai mic.

1. Fişierul text ***cifre.txt*** conţine pe prima linie un număr natural n (0<n<1000), iar pe a doua linie n numere naturale cu cel mult 9 cifre fiecare. Scrieţi un program care citeşte toate numerele din fişier şi afişează pe ecran, separate prin câte un spaţiu, numerele formate doar din cifre distincte şi care au exact trei cifre.

Exemplu: dacă fişierul ***cifre.txt*** are următorul conţinut:

**7**

**249 511 4329 2 4313 243 3562**

atunci pe ecran se vor afişa numerele 249 243.

1. Din fişierul ***vector.in*** se citeşte un vector de numere întregi, pozitive, cu cel puţin 2 cifre fiecare. Sa se determine cea mai lungă subsecvenţă de elemente prime, ale căror inverse sunt tot numere prime.

Exemplu: dacă fişierul ***vector.in*** are următorul conţinut:

**9**

11 971 44 19 **181 751 347** 33 929

atunci subsecvenţa cerută este: 181 751 347

1. Se dă un vector de numere întregi, pozitive, care se citeşte din fişierul ***vector.in***. Se cere să se afişeze subsecvenţa palindromică de lungime maximă.

Exemplu: dacă fişierul ***vector.in*** are următorul conţinut:

15

1 12 31 12 1 4 27 13 9 26 9 13 27 4 131

atunci subsecvenţa cerută este: 4 27 13 9 26 9 13 27 4

1. Se citeşte de la tastatură un număr natural n şi elementele unui tablou unidimensional cu n elemente numere întregi între 1 şi 10, apoi să se determine un tablou unidimensional, astfel încât pe fiecare poziţie i să avem numărul de apariţii a numărului din tabloul iniţial. Rezultatul trebuie scris într-un fişier ***aparitii.out***.

Exemplu: Pentru n=9, v=(1, 5, 2, 1, 5, 7, 2, 1, 5) se obţine w=(3, 3, 2, 3, 3, 1, 2, 3, 3).

**Matrici:**

1. De la tastatură de citește un număr natural n (n<=10) apoi un vector cu n\*n elemente numere întregi. Să se completeze matricea circular cu elementele vectorului în ordinea citirii. Matricea formată va fi afișată în fișierul ***matrice.txt***.

Exemplu: n=4, vectorul este: 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16.

Matricea afișată va fi:

1. Din fișierul ***vecini.txt*** se citește un număr n, urmat de o matrice pătratică cu elemente numere întregi. Să se afișeze pe linii diferite ale ecranului coordonatele elementelor care au ca vecini (8 direcții) numai numere impare.
2. Scrieţi programul care citeşte de la tastatură un număr natural n (n≤50) şi construieşte în memorie o matrice cu n linii şi n coloane, ale cărei elemente sunt numere întregi citite de la tastatură. Pentru fiecare coloană a matricei, în ordine, programul afişează în fișierul ***minim.txt*** cel mai mic număr de pe respectiva coloană. Numerele afişate vor fi separate prin câte un spaţiu.
3. Fie fișierul ***maxim.txt*** care conține pe prima linia două numere naturale *n* și *m*. Începând cu linia a doua a fișierului găsim cele *n\*m* elemente ale unei matrici. Să se determine prima valoare maximă din matrice. Se va afișa pe ecran alături de coordonatele sale. Să se elimine din matrice linia și coloana unde s-a găsit această valoare maximă. Matricea va fi afișată după ștergere.
4. Din fişierul ***matrice***.***in*** se citeşte de pe primul rând două valori *n* şi *m* ce reprezintă numărul de linii şi de coloane ale unei matrici şi apoi *n\*m* valori întregi. Să se rotească matricea cu 90 de grade în sens trigonometric şi să se afişeze noua matrice.
5. Se citește de la tastatură un număr natural n – impar. Să se genereze în fișierul ***matrice.txt*** o matrice de forma:

|  |  |  |
| --- | --- | --- |
| N=3 | N=5 | N=7 |
| **1 1 1**  **1 3 1**  **1 1 1** | **1 1 1 1 1**  **1 3 3 3 1**  **1 3 5 3 1**  **1 3 3 3 1**  **1 1 1 1 1** | **1 1 1 1 1 1 1**  **1 3 3 3 3 3 1**  **1 3 5 5 5 3 1**  **1 3 5 7 5 3 1**  **1 3 5 5 5 3 1**  **1 3 3 3 3 3 1**  **1 1 1 1 1 1 1** |

1. Scrieţi un program care citeşte de la tastatură un număr natural nenul. Fie x- numărul dn cifre al numărului citit. Construiți în memorie şi afişsți apoi pe ecran o matrice având x linii şi x coloane, completată astfel: elementele de pe coloană x a matricei vor fi toate egale cu cifra unităţilor numărului dat, elementele de pe coloana x-1 a matricei vor fi toate egale cu cifra zecilor... Prima coloană va conține prima cifră a numărului dat.

|  |  |
| --- | --- |
| N=123 | N=10038 |
| **1 2 3**  **1 2 3**  **1 2 3** | **1 0 0 3 8**  **1 0 0 3 8**  **1 0 0 3 8**  **1 0 0 3 8**  **1 0 0 3 8** |

1. Din fişierul ***matrice.in*** se citeşte de pe primul rând două valori n şi m ce reprezintă numărul de linii şi de coloane ale unei matrici şi apoi n\*m valori întregi. Să se afişeze toate elementele de tip „șa”. Numim elemente de tip „șa” acele elemente care sunt maxime pe linia de care apartin şi minime pe coloană sau invers.
2. Fişierul ***matrice.in*** conţine pe primul rând două valori *n* şi *m* ce reprezintă numărul de linii şi de coloane ale unei matrici şi apoi *n\*m* valori întregi. Să se determine cea mai mare valoare aflată pe rama matricii şi de cîte ori apare această valoare.  
   Numim *ramă a matricii* cadrul format din prima linie, ultima coloană, ultima linie şi prima coloană.
3. Se dă o matrice pătratică A cu n linii si n coloane, cu valori pozitive. Să se scrie un program care citește matricea, înlocuiește fiecare valoare de pe diagonalele principală si secundară cu suma elementelor de pe linia respectivă și apoi afișează matricea într-un fișier text ***rez.txt*** (pe fiecare rând, o linie a matricii).

Ex: n=4

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 6 | 1 | 2 |  | 12 | 6 | 1 | 12 |
| 1 | 9 | 4 | 7 |  | 1 | 21 | 21 | 7 |
| 4 | 2 | 6 | 3 | Devine: | 4 | 15 | 15 | 3 |
| 2 | 5 | 1 | 8 |  | 16 | 5 | 1 | 16 |

**Șiruri de caractere:**

1. Din fișierul ***cuvinte.txt*** se citesc mai multe cuvinte, fiecare de maxim 20 caractere, scrise pe linii diferite. Să se afișeze aceste cuvinte în ordine crescătoare, ținând cont de numărul de vocale din fiecare.

Exemplu:

|  |  |
| --- | --- |
| ***cuvinte.txt*** | ***Pe ecran*** |
| **Atestat**  **informatica**  **Bac**  **programator**  **Pascal**  **Cpp**  **Competente** | **Cpp**  **Bac**  **Pascal**  **Atestat**  **Competente**  **programator**  **informatica** |

1. Scrieţi un program care citeşte de la tastatură un cuvânt format din cel mult 20 de litere mici ale alfabetului englez şi afişează pe ecran, toate “clonele” (*s2 este “clona” şirului de caractere s1 dacă se poate obţine din s1 prin eliminarea tuturor apariţiilor unei singure vocale*) acestui cuvânt, fiecare pe câte o linie a ecranului.

Exemplu: pentru cuvântul ***informatica*** se afişează, nu neapărat în această ordine, “clonele” scrise alăturat: ***nformatca, infrmatica, informtic***.

1. Scrieţi un program care citeşte de la tastatură două caractere c1 şi c2 şi un text având cel mult 250 caractere (spaţii şi litere ale alfabetului englez), pe care îl modifică înlocuind toate apariţiile caracterului memorat în c1 cu cel memorat în c2 şi toate apariţiile caracterului memorat în c2 cu cel memorat în c1. Programul afişează pe linii separate ale ecranului atât textul iniţial cât şi textul obţinut după efectuarea înlocuirilor.
2. Se citește un text de la tastatură. Să se insereze după fiecare vocală mică, vocala mare corespunzătoare. Noul șir format se va scrie în fișierul ***vocale.txt***.

Ex. ”informatica” va deveni ”**iInfoOrmaAtiIcaA**”.

1. Din fișierul ***sufix\_prefix.txt*** se citesc 2 texte de maxim 50 caractere situate pe linii diferite. Pentru primul cuvânt să se afișeze toate sufixele sale pornind de la 1 caracter. Pentru cel de al doilea text să se afișeze prefixele sale pornind de la șirul inițial.

Exemplu: în fișierul de intrare avem ”mate”, ”info”, atunci pe ecran vom obține:

e

te

ate

mate

info

inf

in

i

1. Un şir cu maximum 255 de caractere conţine cuvinte separate prin unul sau mai multe spaţii. Cuvintele sunt formate numai din litere mici ale alfabetului englez. Scrieţi un program care citeşte un astfel de şir şi îl afişează în fișierul ***litere.txt*** modificat, prima şi ultima literă a fiecărui cuvânt fiind afişată ca literă mare.

Ex: ”atestat la informatica” va fi ”AtestaT LA InformaticA”.

1. Din fişierul ***text.in*** se citeşte o valoare naturală *n* şi apoi de pe următoarele *n* rânduri câte un cuvânt. Să se numere şi să se afişeze câte dintre cuvintele citite încep şi se încheie cu o vocală.
2. Din fişierul ***text.in*** se citeşte o valoare naturală *n* şi apoi de pe următoarele ***n*** rânduri câte un cuvânt. Să se numere şi să se afişeze câte dintre cuvintele citite au număr egal de vocale şi de consoane.
3. Fiind dat un şir de maximum 100 de caractere care conţine cuvinte formate din literele alfabetului englez, separate prin câte un spaţiu. Să se genereze un nou şir în care vocalele să se multiplice de un număr egal de ori cu poziţiile lor din şirul iniţial.

Exemplu:

Şirul dat: Cerul este alb.

Şirul rezultat: Ceeruuuul eeeeeeesteeeeeeeeee aaaaaaaaaaaalb.

1. Fie fişierul text ***in.txt***, care conţine un text de maximum 200 de caractere. Să se scrie un program care codifică textul dat în felul următor: după fiecare vocală inserează litera “p” şi vocala. Textul codificat va fi afişat pe ecran.

Exemplu: dacă conţinutul fişerului ***in.txt*** este „Mere pere banana”,

atunci pe ecran se va afişa: „Meperepe peperepe bapanapanapa”

1. Se citește un șir de caractere de forma: cifră-literă, cifră literă …etc. Să se construiască şi să se tipărească în fişierul ***sir.out*** decodificarea acestuiun şir, în care fiecare literă apare de atâtea ori cât este cifra care o precede.

Exemplu: pentru şirul de intrare **2a4b5c**, în fişierul de ieşire ***sir.out*** se afişa **aabbbbccccc**