1. Operatii cu matrici 1

Cerinte:
Sa se realizeze functii pentru operatii cu matrici patratice (de dimensiune maxima 10x10). Operatiile cerute sunt: A+B (adunare), aA (inmultire cu scalar), A-B scadere), A^T (Transpusa), det(A) (determinantul numai in cazurile in care matricea e patratica de dimensiuni 2x2 sau 3x3). Sa se realizeze un program demonstrativ pentru operatii cu matrici, utilizand aceste functii. Utilizatorului i se va cere sa furnizeze de la tastatura o expresie cu matrici asemanatoare cu expresiile de mai sus. Programul va interpreta expresia si va cere calea inspre fisierul sau fisierele in care se gasesc matricile. Se va calcula rezultatul, care va fi scris intr-un fisier cu denumirea rezultat.txt.

Exemplul 1:

```
> Introduceti o operatie cu matrici:
> A+B
> A=
> a.txt
> B=
> b.txt
> Rezultat: rezultat.txt
```

2. Operatii cu matrici 2

Cerinte:
Sa se realizeze functii pentru operatii cu matrici patratice (de dimensiune maxima 10x10). Operatiile cerute sunt: A+B (adunare), AB (inmultire), aA (inmultire cu scalar), A-B (scadere). Sa se realizeze un program demonstrativ pentru operatii cu matrici, utilizand aceste functii. Utilizatorului i se va cere sa furnizeze de la tastatura o expresie cu matrici asemanatoare cu expresiile de mai sus. Programul va interpreta expresia si va cere calea inspre fisierul sau fisierele in care se gasesc matricile. Se va calcula rezultatul, care va fi scris intr-un fisier cu denumirea rezultat.txt.

Exemplul 1:

```
> Introduceti o operatie cu matrici:
> A+B
> A=
> a.txt
> B=
> b.txt
> Rezultat: rezultat.txt
```

3. Operatii cu multimi 1

Cerinte:

Sa se realizeze functii pentru operatii cu multimi de caractere ASCII. O multime contine maxim 100 de elemente. Operatiile cerunte sunt: verificare multime (elemente unice), apartenenta unui element la multime, reuniune, intersectie.

Sa se realizeze un program demonstrativ pentru operatii multimi, utilizand aceste functii. Utilizatorului i se va cere sa aleaga operatia apoi sa introduca multimile si/sau elementele cerute ca intrare pentru operatie. Pentru fiecare multime introdusa de la tastatura se va face verificarea (elemente unice). Rezultatul va fi afisat pe ecran. Erorile vor fi semnalate prin mesaje. Datele de intrare, operatia si rezultatul se vor scrie si intr-un fisier rezultat.

Exemplul 1:

```
> Introduceti o operatie cu multimi:
> 1. Verificare
> 2. Apartenenta
> 3. Reuniune
> 4. Intersectie
> 2
> e = 5
> Multime:
> 1 2 3 4
> Rezultat: nu apartine
> Fisier rezultat: rezultat.txt
```

4. Operatii cu multimi 2

Cerinte:

Sa se realizeze functii pentru operatii cu multimi de caractere ASCII. O multime contine maxim 100 de elemente. Operatiile cerunte sunt: verificare multime (elemente unice), apartenenta unui element la multime, diferenta, produs cartezian.

Sa se realizeze un program demonstrativ pentru operatii multimi, utilizand aceste functii. Utilizatorului i se va cere sa aleaga operatia apoi sa introduca multimile si/sau elementele cerute ca intrare pentru operatie. Pentru fiecare multime introdusa de la tastatura se va face verificarea (elemente unice). Rezultatul va fi afisat pe ecran. Erorile vor fi semnalate prin mesaje. Datele de intrare, operatia si rezultatul se vor scrie si intr-un fisier rezultat.

Exemplul 1:

```
> Introduceti o operatie cu multimi:
> 1. Verificare
> 2. Apartenenta
> 3. Diferenta
> 4. Produs cartezian
> 2
> e = 5
> Multime:
> 1 2 3 4
> Rezultat: nu apartine
> Fisier rezultat: rezultat.txt
```

5. Aplicatie calculator de buzunar in baza 10

Cerinte:

Sa se realizeze un program care executa functiile unui calculator de buzunar cu operatii aritmetice si logice (cu intregi pe 32 de biti) in zecimal. Operatiile trebuie sa fie implementate in functii. Operatii cerute: +, -, *, /.

Utilizatorul va introduce de la tastatura expresiile pe care doreste sa le calculeze. Semnul „=” inseamna ca expresia trebuie sa fie calculata si rezultatul afisat pe ecran. Daca expresia incepe cu un operator, primul operand este considerat ca fiind egal cu ultimul rezultat obtinut. Daca anterior nu s-a calculat nici un rezultat, operandul se considera a fi 0. Programul va recunoaste comanda exit (terminarea programului). Pana la primirea comenzii exit, programul va rula in continuu. O expresie contine oricati operanzi. Inmultirea si impartirea sunt prioritare.

Exemplu:

```
> Introduceti o expresie:
> 2+3-2*2=1
> Introduceti o expresie:
```

```
> +25=26
> Introduceti o expresie:
> exit
```

6. Aplicatie calculator de buzunar in baza 16

Cerinte:

Sa se realizeze un program care executa functiile unui calculator de buzunar cu operatii aritmetice si logice (cu intregi pe 32 de biti) in hexazecimal. Operatiile trebuie sa fie implementate in functii. Operatii cerute: +, -, *, /.

Utilizatorul va introduce de la tastatura expresiile pe care doreste sa le calculeze. Semnul „=” inseamna ca expresia trebuie sa fie calculata si rezultatul afisat pe ecran. Daca expresia incepe cu un operator, primul operand este considerat ca fiind egal cu ultimul rezultat obtinut. Daca anterior nu s-a calculat nici un rezultat, operandul se considera a fi 0. Programul va recunoaste comanda exit (terminarea programului). Pana la primirea comenzii exit, programul va rula in continuu. O expresie contine oricati operanzi. Inmultirea si impartirea sunt prioritare.

Exemplu:

```
> Introduceti o expresie:
> 2+3-2*2=1
> Introduceti o expresie:
> +0A=0B
> Introduceti o expresie:
> exit
```

7. Aplicatie calculator de buzunar (numere reale)

Cerinte:

Sa se realizeze un program care executa functiile unui calculator de buzunar cu operatii aritmetice si functii pentru numere reale. Operatiile trebuie sa fie implementate in functii separate. Operatii cerute: +, -, *, /. Functii cerute: sin, log.

Utilizatorul va introduce de la tastatura expresiile pe care doreste sa le calculeze. Semnul „=” inseamna ca expresia trebuie sa fie calculata si rezultatul afisat pe ecran. Daca expresia incepe cu un operator, primul operand este considerat ca fiind egal cu ultimul rezultat obtinut. Daca anterior nu s-a calculat nici un rezultat, operandul se considera a fi 0. Programul va recunoaste comanda „exit” (terminarea programului) si valoarea „pi”. Pana la primirea comenzii exit, programul va rula in continuu. O expresie contine oricati operanzi. Inmultirea si impartirea sunt prioritare, nu se folosesc paranteze.

Exemplu:

```
> Introduceti o expresie:
> 3.5-2*2 = -0.5
> Introduceti o expresie:
> sin(pi/2)=1
> Introduceti o expresie:
> +2.2=3.2
> Introduceti o expresie:
> exit
```

8. Operatii pe lista simplu inlantuita de intregi

Cerinte:

Se va scrie un set de functii care implementeaza operatii pe liste simplu inlantuite de numere intregi (pe 32 de biti). Se vor scrie functii pentru:

- Creare lista
- Stergere lista
- Inserare element in lista – se aloca un element in memorie si se ataseaza la
- sfarsitul listei
- Stergere element din lista – se da indexul elementului care va fi sters, se refac
- legaturile si se dezaloca elementul din memorie
- Copierea valorilor din lista intr-un sir de intregi
- Salvare lista in fisier – se salveaza valorile listei in fisier, se sterge lista
- Incarcare lista din fisier – se creaza o lista noua si se copiaza valorile din fisier

Daca fisierul in care s-a salvat o lista va fi folosit pentru a crea o noua lista, valorile din lista salvata trebuie sa se regaseasca in lista noua in acceasi ordine. Se va scrie un program demonstrativ, interactiv, in care se va utiliza setul de functii pentru liste. Programul va recunoaste comenzile:

- Creare lista – creaza o lista
- In numar – introduce la sfarsitul listei elementul cu valoarea numarului
- Sterge index – sterge elementul care se afla la indexul precizat
- Afisare – afiseaza valorile din lista
- Save lista.txt – salveaza lista in fisierul precizat si o sterge din memorie
- Load lista.txt – incarca lista din fisierul precizat
- Exit – sterge lista si termina executia

Exemplu de functionare:

```
>creare lista
>in 23
>in 11
>in 22
>afisare
23 11 22
>sterge 1
>afisare
23 22
>save lista.txt
>afisare
>load lista.txt
>afisare
23 22
>exit
```

9. Operatii pe fisiere text

Cerinte:

Sa se realizeze un set de functii pentru operatii pe fisiere text. Operatiile cerute sunt: findc (numar de aparitii a unui caracter in text), find (gasire secventa de caractere in text), replace (inlocuirea unei secvente de caractere cu alta), toUpper (toate caracterele sunt transformate in caractere mari), toLower (toate caracterele sunt transformate in caractere mici), toSentece (textul este formatat astfel incat dupa semne de punctuatie care marcheaza sfarsitul unei fraze sa se inceapa cu un caracter mare si sa se continue cu caractere mici) in fisiere text, list (afisare la consola). Sa se scrie un demonstrativ care sa primeasca o cale catre un fisier text, o operatie si parametri (daca este cazul). Toate operatiile se vor efectua pe fisierul initial. Programul va rula in continuu pana cand va fi data ca operatie comanda exit.

Exemplu:

```
> Introduceti calea spre fisier:
> file.txt
> Operatia:
> find ana
> 2 aparitii la index: 5, 35
> Operatia:
```

```
> relpace ana mana
> Operatia:
> toUpper
> Operatia:
>exit
```

10. Operatii pe siruri de intregi

Cerinte:

Sa se realizeze un set de functii pentru operatii pe siruri de numere intregi citite dintr-un fisier text. Numerele se vor citi din fisier se vor stoca in memorie, apoi se va determina valoarea minima si maxima (care vor fi de asemenea stocate in memorie). Operatiile pe siruri cerute sunt:

1. Histograma valorilor.
2. Calculul mediei bazat pe histograma.
3. Calculul deviatiei standard bazat pe histograma $\sigma = \sqrt{\frac{\sum_1^n |x-\bar{x}|^2}{n-1}}$
4. Eliminarea din sir a valorilor care sunt in afara intervalului $[-2\sigma, 2\sigma]$

Utilizatorul va alege o operatie dintr-un meniu de optiuni rezultatul va fi pastrat intr-un fisier. Programul va permite realizarea unei serii de operatii pe fisierul initial si pe rezultate intermediare.

Exemplu:

```
> Incarcare fisier:
> numere.txt
> min=12, max=260
> salvat in minmax.txt
> Selecteaza operatie
> 1.Histograma
> 2.Calculul mediei bazat pe histograma
> 3.Calculul deviatiei standard
> 4.Eliminare valori
> 1 numere.txt
> salvat in histograma.txt
> Selecteaza operatie
> 1.Histograma
> 2.Calculul mediei bazat pe histograma
> 3.Calculul deviatiei standard
> 4.Eliminare valori
> 2 histograma.txt
> salvat in media.txt
```

11. Criptare/decriptare fisiere text

Cerinte:

Sa se realizeze functii pentru criptarea si decriptarea fisierelor text. Sa se implementeze aceste functii pentru 2 algoritmi de criptare.

Sa se scrie un program demonstrativ care realizeaza criptarea/decriptarea unui fisier text. Utilizatorului i se va cere sa furnizeze de la tastatura urmatoarele (in aceasta ordine):

1. Calea absoluta catre fisier.
2. Operatia.
3. Cheia de criptare.

Algoritmul 1

1. Se va lua fiecare caracter ASCII si se va face operatia de complement fata de 2 (pe biti). Exemplu: '0'=30H = 00110000B =>(C1) 11001111+1=>C2
2. La valoarea obtinuta la pasul 1 se va face rotire la dreapta (ROR) cu C , unde C e cheia de criptare (o valoare de la 0 la 7).
3. Pentru decriptare se vor realiza operatiile inverse.

Algoritmul 2

1. Se va lua cate un bloc de 10 octeti si se va face operatia de complement fata de 1 (pe biti).
2. La valoarea obtinuta la pasul 1 se va aplica XOR cu o cheie de criptare pe 64 de biti.
3. Pentru decriptare se vor realiza operatiile inverse.

12. Comprimare/ decomprimare fisiere text

Cerinte:

Sa se realizeze functii pentru comprimarea si decomprimarea fisierelor text. Sa se implementeze aceste functii pentru 1 algoritm de comprimare. Algoritmul de comprimare trebuie sa functioneze pentru mai mult de 20 de cuvinte.

Sa se scrie un program demonstrativ care realizeaza comprimarea/decomprimarea unui fisier text. Utilizatorului i se va cere sa furnizeze de la tastatura calea absoluta catre fisier.

Algoritm de comprimare bazat pe dictionar:

1. Se analizeaza fisierul si se face o lista cu toate cuvintele din text fara a se tine cont de literele mari si de aparitii multiple (se contruieste multimea cuvintelor care alcatuiesc textul). Se atribuie fiecarui cuvant un caracter ASCII (cifra, litera mare/mica) sau o combinatie de 2 caractere ASCII, astfel construindu-se un dictionar.
2. In fisierul text se vor inlocui cuvintele cu corespondentul lor (caracterul ASCII) din dictionar. Se va salva pe disc noul fisier si un alt fisier care contine dictionarul.

Exemplu:

Text: Ana are mere. George are mere.

Dictionar: ana-0, are-1, mere-2, george-3

Text criptat: 0 1 2 . 3 1 2 .