Laboratorul 4 – Instrucțiuni

```
/// 3.2. Se citesc 4 perechi de numere reale, care reprezintă în coordonatele vârfurilor unui patrulater. Să se stabilească natura acestui patrulater.

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

typedef struct { float x, y; } Punct;

float panta(Punct *A, Punct *B){ return (B->y-A->y)/(B->x-A->y); }

float unghi(Punct *A, Punct *B, Punct *C, Punct *D){
    float u1, u2;

    u1=atan(panta(A, B))*180/M_PI;
    u2=atan(panta(C, D))*180/M_PI;

    return u2-u1;
}

int trapez(Punct *A, Punct *B, Punct *C, Punct *D){
    if(panta(A, B)==panta(C, D) && panta(B, C)!=panta(D, A) || panta(B, C)==panta(D, A) && panta(A, B)!=panta(C, D)) return 1;
    return 0;
}

int paralelogram(Punct *A, Punct *B, Punct *C, Punct *D){
    if(panta(A, B)==panta(A, D) && panta(B, C)==panta(D, A)) return 1;
    return 0;
}

int dreptunghi(Punct *A, Punct *B, Punct *C, Punct *D){
    if(abs(unghi(D, A, A, B))==90) return 1;
    return 0;
}

int romb(Punct *A, Punct *B, Punct *C, Punct *D){
    if(panta(A, C)*panta(B,D)==-1) return 1;
    return 0;
}

int patrat(Punct *A, Punct *B, Punct *C, Punct *D)
{
    if(dreptunghi(A, B, C, D) && romb(A, B, C, D)) return 1;
    return 0;
}


int main()
{
    Punct *A, *B, *C, *D;
```

```
    printf("Cititi varful lui A (x, y): ");
    scanf("(%lf, %lf)%*c", &A->x, &A->y);

    printf("Cititi varful lui B (x, y): ");
    scanf("(%lf, %lf)%*c", &B->x, &B->y);

    printf("Cititi varful lui C (x, y): ");
    scanf("(%f, %f)%*c", &C->x, &C->y);

    printf("Cititi varful lui D (x, y): ");
    scanf("(%f, %f)", &D->x, &D->y);

    if(patrat(A, B, C, D)==1) printf("ABCD este patrat");
    else if(trapez(A, B, C, D)==1) printf("ABCD este trapez");
    else if(dreptunghi(A, B, C, D)==1) printf("ABCD este dreptunghi");
    else if(romb(A, B, C, D)==1) printf("ABCD este romb");
    else if(paralelogram(A, B, C, D)) printf("ABCD este paralelogram");

    return 0;
}
```

```
/// 3.3. De la intrarea standard sunt citite elementele reale ale unui şir de dimensiune
n. Să se găsească valoarea minimă şi valoarea maximă dintre elementele şirului şi poziţia
lor.

#include <stdio.h>
#include <stdlib.h>
#define N 20

int main()
{
    float a[N];

    int n, i, imin=0, imax=0;

    for(n=-1; n<1 || n>N; scanf("%d", &n)){
        printf("Introduceti numarul elementelor (<=%d): ", N);
        fflush(stdin);
    }

    for(i=0; i<n; i++){
        printf("a[%d] = ", i);
        scanf("%f", &a[i]);
    }

    for(i=1; i<n; i++){
        if(a[i]<a[imin]) imin=i;
        if(a[i]>a[imax]) imax=i;
    }

    printf("Minimul %.2f e pe pozitia %d\n", a[imin], imin);

    return 0;
}
```

```c
/// 3.4. Să se scrie un program pentru generarea tuturor numerelor prime mai mici sau egale cu un număr natural n (Ciurul lui Eratostenes).

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define N 100000

int main()
{
    char a[N]={0};
    int n, i, sqrtn, j;

    printf("Introduceti valoarea lui n: ");
    scanf("%d", &n);

    sqrtn=sqrt(n);

    for(i=2; i<=sqrtn; i++){
        if(a[i]==0)
            for(j=i+i; j<=n; j=j+i) a[j]=1;
    }

    for(i=2; i<=n; i++)
        if(a[i]==0) printf("%d ", i);

    return 0;
}
```

```c
/// 3.5. Se citeşte un număr natural n. Să se găsească cel mai mare pătrat perfect mai mic sau egal cu n. Aceeaşi problemă, dar să se indice numărul prim cel mai mic, dar mai mare sau egal cu numărul citit.

#include <stdio.h>
#include <stdlib.h>

int prim(int n){
    if(n==0 || n==1) return 0;
    if(n==2) return 1;
    if(n%2==0) return 0;
    for(int d=3; d*d<=n; d=d+2)
        if(n%d==0) return 0;
}
int main()
{
    int n, n_aux;

    printf("Introduceti valoarea pentru n: ");
    scanf("%d", &n); n_aux=n;

    printf("Cel mai mare patrat perfect mai mic <=%d: %d\n", n,
(int)sqrt(n)*(int)sqrt(n));

    while(!prim(n_aux)) n_aux++;
    printf("Cel mai mic numar prim >=%d: %d", n, n_aux);
    return 0;
}
```

```c
/// 3.6. Se citeşte un număr natural n. Să se verifice dacă numărul respectiv este palindrom.

#include <stdio.h>
#include <stdlib.h>

int main()
{
    int n, n_aux, n_inv=0;

    printf("Introduceti valoarea pentru n: ");
    scanf("%d", &n);

    n_aux=n;

    while(n_aux){
        n_inv=n_inv*10+n_aux%10;
        n_aux/=10;
    }

    if(n_inv==n) printf("%d este palindrom", n);
    else printf("%d nu este palindrom", n);

    return 0;
}
```

```c
/// 3.7. Se citesc cifrele hexazecimale ale unui număr întreg în baza 16. Să se calculeze
şi să se afişeze reprezentarea numărului în baza 10.
/// Observaţie: Problema este rezolvată pentru orice bază!

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#define BMAX 36

int main()
{
    char nr[20];

    int b, i, val=0, cifra;

    printf("Baza (intre 2 si %d): ", BMAX);

    scanf("%d%*c", &b);
    printf("Numarul in baza %d: ", b);

    scanf("%s", &nr);

    for(i=0; nr[i]; i++){
        if(nr[i]>='0' && nr[i]<='9') cifra=nr[i]-'0';
        else {
            nr[i]=toupper(nr[i]);
            if(nr[i]>='A' && nr[i]<='Z') cifra=nr[i]-'A'+10;
            else{
                printf("%c: caracter gresit\n", nr[i]);
                exit(1);
            }
        }

        if(cifra>=b){
            printf("%d: cifra gresita\n", cifra);
            exit(1);
        }

        val=val*b+cifra;
    }

    printf("Numarul %s din baza %d are valoarea %d\n", nr, b, val);

    return 0;
}
```

```c
/// 3.8. Se citesc gradul şi coeficienţii polinomului p(x)=a0+a1*x^1+...+an*x^n. Să se calculeze valoarea polinomului în punctul x=x0 (valoarea x0 se citeşte).

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define N 20

int main()
{
    int a[N], n, x0, val_pol, i;

    printf("Introduceti gradul polinomului (maxim %d): ", N);
    scanf("%d", &n);

    printf("Introduceti valoarea pentru x0: ");
    scanf("%d", &x0);

    printf("Introduceti coeficientii polinomului: \n");
    for(i=0; i<=n; i++){
        printf("a[%d] = ", i);
        scanf("%d", &a[i]);
        val_pol+=a[i]*pow(x0, i);
    }

    printf("f(%d) = %d", x0, val_pol);

    return 0;
}
```

```c
/// 3.9. Să se scrie un program pentru efectuarea operaţiilor de adunare, scădere,
înmulţire şi împărţire între două polinoame.

#include <stdio.h>
#include <conio.h>
#define GRADMAX 20

void produs(int n, float a[], int m, float b[], int *p, float c[]){
    int i,j;
    *p=n+m;
    for(i=0; i<=n+m; i++) c[i]=0;
    for(i=0; i<=n; i++)
        for(j=0; j<=m; j++)
            c[i+j]+=a[i]*b[j];
}

void impartire(int n, float a[], int m, float b[], int *grad_cat, float cat[], int
*grad_rest, float rest[]){
    int i,j,k;
    if(n<m){
        *grad_cat=0;
        cat[0]=0;
        *grad_rest=n;
        for(i=0; i<=n; i++) rest[i]=a[i];
    }
    else {
        *grad_cat=n-m;
        *grad_rest=m-1;
        for(i=n-m, j=n; i>=0; i--, j--){
            cat[i]=a[j]/b[m];
            for(k=m; k>=0; k--) a[i+k]=a[i+k]-cat[i]*b[k];
            a[j]=0;
        }
        for(i=0; i<=m-1; i++) rest[i]=a[i];
    }
}

void citire_polinom(int *n, float a[]){
    int i;
    printf("\nIntroduceti gradul polinomului: ");
    scanf("%d", n);

    for(i=0; i<=*n; i++){
        printf("\ta[%d]=", i);
        scanf("%f", &a[i]);
    }
}

void afis_polinom(int n, float a[], char c){
    int i;
    printf("\t%c[x]=%g", c, a[0]);
    for(i=1; i<=n; i++) printf("+%g*x^%d",a[i],i);
    printf("\n");
}

int main()
{
    int n, m, grad_r, grad_cat, grad_rest;
```

```
    float x, v, p[GRADMAX+1], q[GRADMAX+1], r[GRADMAX+1], cat[GRADMAX+1],
rest[GRADMAX+1];

    citire_polinom(&n, p);
    afis_polinom(n, p, 'P');

    citire_polinom(&m, q);
    afis_polinom(m, q, 'Q');

    produs(n, p, m, q, &grad_r, r);
    printf("\n\nR[x]=P[x]*Q[x]\n\n");
    afis_polinom(grad_r, r, 'R');

    printf("\n\nRezultatul impartirii P[x]/Q[x]=>catul C[x] si restul R[x]\n\n");
    impartire(n, p, m, q, &grad_cat, cat, &grad_rest, rest);
    afis_polinom(grad_cat, cat, 'C');
    afis_polinom(grad_rest, rest, 'R');

    return 0;
}
```

/// 3.10. Se dă un sistem de n ecuaţii liniare cu n necunoscute. Să se scrie un program de rezolvare a sistemului, folosind o metodă numerică.

/// 3.11. Să se calculeze polinoamele P(x) şi Q(x) din relaţia dată.

```c
/// 3.12. Se citeşte un şir de n elemente reale ordonate crescător. Să se verifice dacă o valoare citită x se găseşte în şir şi să se indice poziţia sa.

#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a[100], n, x, i, mij, ls=0, ld;

    printf("Introduceti numarul de elemente: ");
    scanf("%d", &n);

    printf("Introduceti valoarea cautata: ");
    scanf("%d", &x);

    ld=n;

    for(i=0; i<n; i++){
        printf("a[%d] = ", i);
        scanf("%d", &a[i]);
    }

    while(ls<=ld){
        mij=(ld+ls)/2;
        if(a[mij]==x){
            printf("S-a gasit %d pe pozitia %d", x, mij);
            exit(1);
        }
        if(a[mij]>x) ld=mij-1;
        else ls=mij+1;
    }

    printf("Nu s-a gasit nicaieri valoarea %d", x);

    return 0;
}
```

```
/// 3.13. Se citește un șir de n numere întregi. Să se extragă subșirul de dimensiune
maximă, ordonat crescător.

#include <stdio.h>
#include <stdlib.h>

void afisare(int k, int a[], int p[]){
    while(k){
        printf("%d ", a[k]);
        k=p[k];
    }
}

int main()
{
    int lis[1000]={1}, n, i, j, max;

    printf("Introduceti valoarea pentru n: ");
    scanf("%d", &n);

    for(i=0; i<n; i++){
        printf("a[%d] = ", i);
        scanf("%d", &a[i]);
    }

    for(i=0; i<n-1; i++){
        max=0;
        for(j=i+1; i<n; i++)
            if(max<lis[i] && x[i]<=x[j]) max=lis[j];

        lis[i]=max+1;
    }

    for(i=2; i<n; i++){
        if(max<lis[i]) max=lis[i];
    },
    return 0;
}
```

/// 3.14. Pentru elaborarea unui test de aptitudini se dispune de un set de n întrebări, fiecare întrebare i fiind cotată cu un număr de pi puncte. Să se elaboreze toate chestionarele având q întrebări, fiecare chestionar totalizând între a și b puncte. Întrebările sunt date prin număr și punctaj.

```c
/// 3.15. Se dau 2 şiruri de n si respectiv m elemente de tip întreg. Să se calculeze:

#include <stdio.h>
#include <stdlib.h>

int apartine(int x, int a[], int n){
    int i;
    for(i=0; i<n; i++)
        if(x==a[i]) return 1;
    return 0;
}

int main()
{
    int nA, nB, a[100], b[100], i, j, c[100], k=0;

    printf("Introduceti numarul de elemente pentru sirul A: ");
    scanf("%d", &nA);

    for(i=0; i<nA; i++){
        printf("A[%d] = ", i);
        scanf("%d", &a[i]);
    }

    printf("Introduceti numarul de elemente pentru sirul B: ");
    scanf("%d", &nB);

    for(i=0; i<nB; i++){
        printf("B[%d] = ", i);
        scanf("%d", &b[i]);
    }

    /// a) şirul ce conţine elementele comune ale celor două şiruri;
    printf("\nIntersectie: ");
    k=0;
    for(i=0; i<nB; i++)
        if(apartine(b[i], a, nA)) c[k++]=b[i];
    for(i=0; i<k; i++) printf("%d ", c[i]);

    /// b) şirul ce conţine toate elementele celor două şiruri luate o singura dată;
    printf("\nReuniune: ");
    k=nA;
    for(i=0; i<nA; i++) c[i]=a[i];
    for(i=0; i<nB; i++)
        if(!apartine(b[i], a, nA)) c[k++]=b[i];
    for(i=0; i<k; i++) printf("%d ", c[i]);

    /// c) şirul ce conţine elementele primului şir din care au fost eliminate elementele
comune.
    k=0;
    printf("\nA fara B: ");
    for(i=0; i<nA; i++)
        if(!apartine(a[i], b, nB)) c[k++]=a[i];
    for(i=0; i<k; i++) printf("%d ", c[i]);

    return 0;
}
```

```c
/// 3.16. Se citește un număr real în baza 10. Să se scrie programul de conversie a lui în baza B, B <= 16.

#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#define BMAX 16

int main()
{
    char s[20];

    int b, nr, i=-1, r, cn;

    printf("Baza (intre 2 si %d): ", BMAX);

    scanf("%d%*c", &b);
    printf("Numarul in baza 10: ");
    scanf("%d", &nr);

    cn=nr;

    do{
        r=cn%b;
        cn/=b;
        i++;
        if(r<10) s[i]=r+'0';
        else s[i]=r-10+'A';
    } while(cn>0);

    printf("Numarul %d transformat in baza %d: \n", nr, b);

    printf("%s", s);


    return 0;
}
```

```c
/// 3.17. Se citește un număr natural n.

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int nrCifre(int n){ return log10(n)+1; }

int elimCifra(int n, int c){
    int p=1, rez=0;
    while(n){
        if(n%10!=c){
            rez=rez+(n%10)*p;
            p*=10;
        }
        n/=10;
    }
    return rez;
}

int main()
{
    int cifre[10], i, nr, n, n_aux, j, aux, fr[10]={0}, cifMaxima=-1, nrAparMax=-1;

    printf("Introduceti numarul n: ");
    scanf("%d", &nr);

    n=nrCifre(nr);
    n_aux=nr;

    /// Să se găsească numărul obţinut prin eliminarea cifrelor care apar de mai multe
ori în număr.
    for(i=n-1; i>=0; i--){
        fr[nr%10]++;
        cifre[i]=nr%10;
        nr/=10;
    }

    for(i=0; i<=9; i++)
        if(fr[i]>nrAparMax){
            nrAparMax=fr[i];
            cifMaxima=i;
        }

    printf("Numarul %d fara cifra %d: %d\n", nr, cifMaxima, elimCifra(n_aux, cifMaxima));

    /// b) Să se găsească numărul obţinut prin interschimbarea între ele a primei cifre
cu ultima, a celei de a doua cu penultima ş.a.m.d.
    for(i=0; i<n/2; i++){
        aux=cifre[i];
        cifre[i]=cifre[n-i-1];
        cifre[n-i-1]=aux;
    }

    printf("Numarul obtinut prin interschimbari: ");
    for(i=0; i<n; i++) printf("%d", cifre[i]);
    printf("\nCel mai mare numar obtinut din cifrele lui %d: ", nr);

    /// c) Să se găsească cel mai mare număr ce se poate obţine din cifrele sale.â
```

```
    for(i=9; i>=0; i--)
        for(j=1; j<=fr[i]; j++)
            printf("%d", i);

    return 0;
}
```

```
/// 3.18. Se citește o matrice de dimensiune nxn cu elemente 0 și 1. Să se stabilească
dacă matricea respectivă este simetrică.

#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a[100][100], n, i, j;

    printf("Introduceti valoarea pentru n: ");
    scanf("%d", &n);

    for(i=0; i<n; i++)
        for(j=0; j<n; j++){
            printf("a[%d][%d] = ", i, j);
            scanf("%d", &a[i][j]);
        }

    for(i=0; i<n; i++)
        for(j=0; j<n; j++)
            if(a[i][j]!=a[j][i]){
                printf("Matricea nu este simetrica");
                exit(1);
            }
1
    printf("Matricea este simetrica");

    return 0;
}
```

```c
/// 3.19. Se citeşte o propoziţie de la intrarea standard. Să se indice numărul
cuvintelor şi cuvântul cel mai lung din propoziţie.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <conio.h>

int main()
{
    int k=0;
    char s[1024], *cuvMax, *p;

    printf("Introduceti o propozitie: ");
    gets(s);

    p=strtok(s, " ");
    cuvMax=p;

    while(p!=NULL){
        if(strlen(p)>strlen(cuvMax)) strcpy(cuvMax, p);
        p=strtok(NULL, " ");
        k++;
    }

    printf("Sunt %d cuvinte, iar cel mai lung este %s", k, cuvMax);

    return 0;
}
```