INSTRUCŢIUNI

1. Conţinutul lucrării

În lucrare sunt prezentate principalele instrucţiuni simple şi structurate din limbajul C: instrucţiunea expresie, instrucţiunea vidă, instrucţiunea compusă, instrucţiunea **if**, instrucţiunea **switch** şi instrucţiunile repetitive.

2. Consideraţii teoretice

Programul structurat este un program care are o structură de control realizată numai cu:

- structura secvenţială;
- structura alternativă şi selectivă;
- structura repetitivă.

În limbajul C mai există instrucţiunile **return, break, continue** şi **goto**, care asigură o flexibilitate mare în scrierea de programe.

2.1. Instrucţiunea expresie

Instrucţiunea expresie are formatul:

expresie;

adică după expresie se scrie "**;**".

Ea se utilizează ca instrucţiune de atribuire sau ca instrucţiune de apel a unei funcţii.

Exemplu de utilizare:

```
/* Programul L4Ex1  */

/* Programul afiseaza maximul dintre 2 intregi */
#include <stdio.h>

int main()
{
  int a,b,c;
  printf("\nIntroduceti doi intregi a si b\n");
  scanf("%d %d",&a,&b);
  c=a>b?a:b;
  printf("\nMaximul dintre a=%d si b=%d este c=%d\n",a,b,c);
  return 0;
}
```

2.2. Instrucţiunea vidă

Instrucţiunea vidă se reduce la punct şi virgulă, fără a avea vreun efect. Ea se utilizează acolo unde se cere prezenţa unei instrucţiuni, dar de fapt nu trebuie să se execute ceva (de exemplu în instrucţiunile repetitive).

Exemplu de utilizare:

```
for (i = 0, s = 0; i < n; s = s + a[i], ++i);
```

2.3. Instrucţiunea compusă

Instrucţiunea compusă este o succesiune de instrucţiuni incluse între acolade, eventual precedate de declaraţii (valabile numai în acest loc):

```
{
   declaraţii;
   instrucţiuni;
}
```

Instrucţiunea compusă se utilizează acolo unde este nevoie conform sintaxei de o singură instrucţiune, dar procesul de calcul necesită mai multe instrucţiuni.

Exemplu de utilizare:

```
/*   Programul L4Ex2 */

/*  Calculul radacinilor ecuatiei a*x^2 +b*x +c =0   */
 #include <stdio.h>
 #include <math.h>
 int main()
 {
    float a,b,c,delta,x1,x2;
    printf("\nIntroduceti a,b,c\n");
    scanf("%f %f %f",&a,&b,&c);
    if (a!=0) {
          delta=b*b-4*a*c;
          if (delta >= 0) {
                    x1=(-b-sqrt(delta))/(2*a);
                    x2=(-b+sqrt(delta))/(2*a);
                    printf("\nEcuatia are radacinile x1=%g  si x2=%g\n", x1, x2);
               }
           else  {
                   x1=-b/(2*a);
                   x2=sqrt(-delta)/(2*a);
                   printf("\nEcuatia are radacinile complex conjugate:\
                            x1=%g - j*%g  si  x2= %g+ j*%g\n",x1,x2,x1,x2);

                  }
          }
     else
          printf("\nEcuatia nu este de ordinul 2 (a=0)\n");
      return 0;
}
```

2.4 Instrucţiunea if

Instrucţiunea if are două formate:

a) **if (expresie)**
instrucţiune

b) **if (expresie)**
instrucţiune_1
else
instrucţiune_2

Efectul ei este următorul:
Se evaluează logic expresia "expresie".
Dacă rezultatul expresiei este **1** se execută în cazul a) instrucţiunea "instrucţiune" şi în cazul b) "instrucţiune_1" şi apoi se trece la instrucţiunea imediat următoare instrucţiunii **if**.
Dacă rezultatul expresiei este **0** se trece în cazul a) la instrucţiunea imediat următoare instrucţiunii if, iar în cazul b) se trece la execuţia "instrucţiune_2" şi apoi se trece la instrucţiunea imediat următoare instrucţiunii structurate **if**.
Observaţii:
a) instrucţiunile "instrucţiune", "instrucţiune_1", "instrucţiune_2" pot conţine instrucţiuni de salt la alte instrucţiuni decât cea următoare instrucţiunii **if** ;
b) instrucţiunea if poate conţine alte instrucţiuni if. Atenţie la îmbinarea lui **else**, în sensul de a şti la care **if** aparţine.

Exemplu de utilizare: Programul L4Ex2 (a se vedea punctul 2.3).

2.5 Instrucţiunea switch

Instrucţiunea **switch** are următoarea sintaxă:

```
switch ( expresie )
{
 case C1:  sir_instrucţiuni_1;
           break;
 case C2:  sir_instrucţiuni_2;
           break;
 ….....................................
 case Cn:  sir_instrucţiuni_n;
           break;
 default:  sir_instrucţiuni
}
```

Efectul instrucţiunii **switch** este următorul:
a) se evaluează "expresie";
b) se compară pe rând rezultatul evaluării cu constantele C1, C2, …, Cn. Dacă rezultatul evaluării coincide cu constanta Ci se vor executa instrucţiunile "sir_instrucţiuni_i" şi apoi se trece la instrucţiunea imediat următoare **switch**-ului. Dacă rezultatul evaluării nu coincide cu nici una din constantele C1, C2, …, Cn se execută instrucţiunile "sir_instrucţiuni" aflate după "**default**".

Observaţii:
a) expresia “expresie” trebuie să se evalueze la o valoare întreagă. Constantele C1, C2, …, Cn trebuie şi ele să reprezinte valori întregi;
b) alternativa **default** este opţională. Dacă nu este prezentă, în cazul în care rezultatul expresiei “expresie” nu coincide cu nici o constantă Ci, instrucţiunea **switch** nu are nici un efect;
c) dacă **break** nu este prezentă, atunci se execută şi şirurile de instrucţiuni imediat următoare, până la întâlnirea unei instrucţiuni **break** sau până la terminarea instrucţiunii **switch**;
d) instrucţiunea structurata **switch** poate fi înlocuită prin instrucţiuni **if** imbricate.

Exemplu de utilizare:

```
/*  Programul L4Ex3  */

/* Operatii cu numere intregi de forma OPERAND1operatorOPERAND2 */

 #include <stdio.h>
 #include <stdlib.h>
 #define INFINIT  0x7fffffff

 int main()
 {
        int operand1,operand2,rezultat;
        char operatie;
        printf("\nScrieti expresia fara spatii intre operanzi si operator\n");
        scanf("%d%c%d",&operand1,&operatie,&operand2);
        switch(operatie)
                {
                 case '+': rezultat=operand1+operand2;
                           break;
                 case '-': rezultat=operand1-operand2;
                           break;
                 case '*': rezultat=operand1*operand2;
                           break;
                 case '/':  if (operand2!=0)
                                 rezultat = operand1/operand2;
                            else if (operand1>0) rezultat=INFINIT;
                                 else rezultat=-INFINIT;
                            break;
                 default:
                            exit(1);
                };

        printf("\n%d %c %d = %d\n", operand1, operatie, operand2, rezultat);
        return 0;
   }
```

2.6 Instrucţiunea while

Formatul instrucţiunii **while** este următorul:

```
while ( expresie )
    instrucţiune
```

Efectul instrucţiunii **while** este următorul:

a) se evaluează logic expresia "expresie";
b) dacă rezultatul este **1** se execută corpul său ("instrucţiune") şi se revine la pasul a); dacă rezultatul este **0** se trece la execuţia instrucţiunii imediat următoare instrucţiunii **while**.

Observaţii:

a) în cazul în care expresie este **0** de la început, atunci "instrucţiune" nu se execută niciodată;
b) în cadrul corpului instrucţiunii **while** este nevoie de existenţa unor instrucţiuni de modificare a variabilelor care intră în "expresie".

Exemplu de utilizare:

```
/*  Programul L4Ex4  */

/* Calculul c.m.m.d.c. si a c.m.m.m.c. a doua numere naturale a si b   */
#include <stdio.h>
int main()
{
    int a,b,a1,b1,cmmdc,cmmmc,rest;
    printf("Introduceti a=");
    scanf("%d",&a);
    printf("Introduceti b=");
    scanf("%d",&b);

    /* Aflarea c.m.m.d.c. */
    a1=a;b1=b;
    while ((rest=a1%b1)!=0)
       {
           a1=b1;
           b1=rest;
       }

     cmmdc=b1;
     cmmmc=a*b/cmmdc;

     printf("a=%d  b=%d  cmmdc(a,b)=%d  cmmmc=%d", a, b, cmmdc, cmmmc);
     return 0;
  }
```

2.7 Instrucţiunea for

Formatul instrucţiunii **for** este următorul:

```
for ( expr1; expr2; expr3 )
    instrucţiune
```

unde:
- expr1, expr2, expr3 sunt expresii;
- instrucţiune este corpul instrucţiunii.

Descrierea efectului instrucţiunii **for**, cu ajutorul instrucţiunii **while** este următorul:

```
expr1;
while ( expr2 ) {
                  instrucţiune;
                  expr3;
                }
```

Observaţie: expr1, expr2, expr3 pot fi vide, însă caracterele **";"** nu pot lipsi.

Exemplu de utilizare:

```
/*  Programul L4Ex5  */

/* Calculul mediei aritmetice a n numere reale */
#include <stdio.h>

int main()
{
 float a[100], media, suma;
 int i,n;
 printf("\nIntroduceti nr.de elemente n=");
 scanf("%d",&n);
 printf("\nIntroduceti elementele sirului\n");
 for(i=0, suma=0; i<n; ++i)
    {
        printf("a[%2d]=",i);
        scanf( "%f",&a[i]);
        suma+=a[i];
    }
 media=suma/n;
 printf("\nMedia aritmetica=%g\n",media);
 return 0;
}
```

2.8 Instrucţiunea do-while

Instrucţiunea **do-while** este instrucţiunea ciclică cu test final. Formatul ei este următorul:

```
do
    instrucţiune
while ( expresie );
```

Efectul ei este descris cu instrucţiunea **while** astfel:

```
instrucţiune;
while( expresie )
     instrucţiune;
```

Se observă că corpul ciclului se execută cel puţin o dată.

Exemplu de utilizare:

```
/*  Programul L4Ex6  */

/* Calculul c.m.m.d.c. si a c.m.m.m.c a doua numere naturale a si b   */
#include <stdio.h>
int main()
{
    int a,b,a1,b1,cmmdc,cmmmc,rest;
    printf("Introduceti a=");
    scanf("%d",&a);
    printf("Introduceti b=");
    scanf("%d",&b);

    /* Aflarea c.m.m.d.c. */
    a1=a;b1=b;
    do{
            rest=a1%b1;
            a1=b1;
            b1=rest;
       }
     while (rest!=0);

     cmmdc=a1;
     cmmmc=a*b/cmmdc;

     printf("a=%d  b=%d  cmmdc(a,b)=%d cmmmc=%d", a, b, cmmdc, cmmmc);
     return 0;
 }
```

2.9 Instrucţiunile continue şi break

Instrucţiunile **continue** şi **break** se pot utiliza numai în corpul unui ciclu.

Instrucţiunea **continue** abandonează iteraţia curentă şi se trece la execuţia pasului de actualizare şi apoi la revalidarea expresiei care stabileşte continuarea sau terminarea ciclului în cazul instrucţiunii **for,** respectiv la revalidarea directă a expresiei care stabileşte continuarea sau terminarea ciclului în cazul instrucţiunilor **while** şi **do-while**.

Instrucţiunea **break** termină ciclul şi se trece la instrucţiunea imediat următoare celei repetitive (**for, while, do-while**).

2.10. Instrucţiunea goto

Instrucţiunea **goto** este utilizată pentru saltul dintr-un punct al unei funcţii (chiar şi dintr-un ciclu) în alt punct al aceleaşi funcţii, respectiv la o instrucţiune etichetată.

Eticheta este un nume urmat de caracterul ":"

nume:

Formatul instrucţiunii **goto** este:

goto eticheta;

Exemplu:

```
…
goto alfa;
…
alfa: if (  ) …
…
```

2.11. Funcţia standard exit

Prototipul funcţiei standard **exit** este descrisă în fişierele **stdlib.h** şi este:

void exit(int cod);

Funcţia **exit** are ca scop terminarea forţată a programului. Codul de ieşire folosit este zero pentru terminare normală şi alte valori pentru terminare datorată unor erori.

3. Mersul lucrării

3.1. Se vor analiza şi executa programele date ca exemplu în lucrare şi în materialul de curs.

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

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.

3.4. Să se scrie un program pentru generarea tuturor numerelor prime mai mici sau egale cu un număr natural n.

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.

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

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.

3.8. Se citesc gradul şi coeficienţii polinomului $p(x)=a_0+a_1x^1+...+a_nx^n$ Să se calculeze valoarea polinomului în punctul $x=x_0$ (valoarea x_0 se citeşte).

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

$A(x)=a_0+a_1x^1+.....+a_nx^n$

$B(x)=b_0+b_1x^1+.....+b_mx^m$

Gradele şi coeficienţii reali ai polinoamelor se citesc de la intrarea standard.

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:

$$\frac{Q(X)}{P(X)}=\sum_{i=1}^{n}\frac{a_i}{b_ix+c_i}$$

Valorile n, a_i , b_i , c_i se citesc de la intrarea standard.

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.

3.13. Se citeşte un şir de n numere întregi. Să se extragă subşirul de dimensiune maximă, ordonat crescător.

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 p_i puncte. Să se elaboreze toate chestionarele având q întrebări, fiecare chestionar totalizând între a şi b puncte. Întrebările sunt date prin număr şi punctaj.

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

a) şirul ce conţine elementele comune ale celor două şiruri;

b) şirul ce conţine toate elementele celor două şiruri luate o singura dată;

c) şirul ce conţine elementele primului şir din care au fost eliminate elementele comune.

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

3.17. Se citește un număr natural n.

a) Să se găsească numărul obţinut prin eliminarea cifrelor care apar de mai multe ori în număr.
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.
c) Să se găsească cel mai mare număr ce se poate obţine din cifrele sale.

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

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