Structuri de bază(liniară, alternativă şi repetitivă)

Programarea structurată este o manieră de concepere a programelor, potrivit unor reguli bine definite şi independent de limbajul de programare. Scopul programării structurate este elaborarea unor programe uşor de scris, de înţeles şi de modificat.

Programarea structurată are la bază **teorema de structură** care afirmă că orice algoritm cu o singură intrare şi o singură ieşire poate fi reprezentat ca o combinaţie de trei tipuri de structuri de control – **secvenţa, decizia** şi **ciclul cu test iniţial.** Se mai admite folosirea a încă trei tipuri de structuri de control – **selecţia, ciclul cu test final** şi **ciclul cu contor.**

Reprezentarea algoritmilor în pseudocod

1. Declararea variabilelor
   variabilă tip;
2. Atribuirea
   variabilă←expresie;
3. Operaţia de citire(intrare)
   citeşte variablia**1**, variabila**2**, ... , variabila**n**;
4. Operaţia de scriere(ieşire)
   Scrie expresie**1**, expresie**2**, ... , expresie**n**;
5. **Structura liniară**(secvenţa)

   Este o succesiune de operaţii ce realizează o prelucrare(transformare) a datelor. Operaţiile sunt executate una după alta, în ordinea scrierii.
6. ***Structura alternativă(decizia)***

   Permite alegerea unei operaţii/secvenţe de operaţii din două alternative posibile.

   a. Cu două ramuri

   dacă C **atunci** A;
   altfel B;

   Am notat cu **C** condiţia care este o propoziţie logică ce poate avea doar una din valorile 1 sau 0(adevărat sau fals). Dacă rezultatul propoziţiei logice **C** este 1(adevărat) atunci se execută secvenţa de operaţii **A**, altfel se execută secvenţa de operaţii **B**.

   b. Cu o ramură

   dacă C **atunci** A;

   Dacă condiţia **C** este adevărată, atunci se execută secvenţa **A.**

   c. Cu mai multe ramuri

   în cazul că selector
   cazul valoare**1**: secvenţa **A1**;
   cazul valoare**2**: secvenţa **A2**;
   .....
   cazul valoare**n**: secvenţa **An**;
   altfel secvenţa **An+1**;

   Se execută, în funcţie de valoarea selectorului, un singur caz.

7. ***Structura repetitivă(ciclu)***

Permite execuţia repetată a unei secvenţe de operaţii, în funcţie de o anumită condiţie. Poate fi reprezentată prin:

a. *Ciclu cu test iniţial*

Execută una sau mai multe operaţii, cât timp o condiţie este adevărată

Cât timp C **execută**
    A

Condiţia **C** este o expresie logică. Cât timp **C** este adevărată, secvenţa **A** este executată. Când **C** devine falsă, bucla se incheie. Condiţia **C** se reevaluează la fiecare intrare în buclă. Deoarece **C** poate fi falsă în momentul intrării în bucla, e posibil ca secvenţa **A** să nu se execute niciodată. Înainte de intrarea în buclă, trebuie să se realizeze o operaţie de iniţializare a valorii testate în **C.** În buclă trebuie să existe o operaţie care să creeze premisele ca **C** să devină falsă pentru a evita ciclarea infinită.

b. *Ciclu cu test final*

Execută repetat una sau mai multe operaţii, până când o condiţie **C** devine adevărată.

repetă
    A
până când C

Această structură poate fi simulată cu ajutorul structurii cu test iniţial:

A
cât timp **not C** execută
    A

c. *Ciclu cu test final*

Execută repetat una sau mai multe operaţii, până când condiţia **C** devine falsă.

execută
    A
cât timp C

Această structură poate fi simulată cu ajutorul structurii cu test iniţial:

A
cât timp **C** execută
    A

d. Ciclu cu contor

Execută repetat, de un număr prestabilit de ori, o secvenţă de operaţii.

Pentru contor ← v_i, v_f [pas v] execută
    A

unde v_i = valoarea iniţială a contorului
v_f = valoarea finală a contorului
v = pasul cu care creşte contorul

Implementarea structurilor de control

1. Implementarea structurii secvenţiale

Structura secventiala este o insiruire de secvente de prelucrare (instructiuni), plasate una dupa alta, in ordinea in care se doreste executia acestora.

a. Instructiunea vida

Sintaxa: **;**

Instructiunea vida nu are nici un efect. Se utilizeaza in constructii in care se cere prezenta unei instructiuni, dar nu se executa nimic (de obicei, in instructiunile repetitive).

Exemple:

```
int a;
for (;;) {…}
```

b. Instructiunea expresie

Sintaxa: expresie;

Exemple:

```
int b, a=9;
double c;
b=a+9;
cout<<a;
```

Instructiunea compusa (instructiunea bloc)

Sintaxa:
```
{  declaratii de variabile;
   instr1;
   instr2;
   . . . . }
```

Intr-un bloc se pot declara si variabile care pot fi accesate doar in corpul blocului. Instructiunea bloc este utilizata in locurile in care este necesara prezenta unei singure instructiuni, insa procesul de calcul este mai complex, deci trebuie descris in mai multe secvente.

2. Implementarea structurii decizionale

a. Instructiunea if

Sintaxa:

```
if (expresie)
    instructiune1;
 [ else
     instructiune2;]
```

Ramura “else” este optionala.

La intalnirea instructiunii “if”, se evalueaza expresie (care reprezinta o conditie) din paranteze. Daca valoarea expresiei este 1, sau diferita de 0 (conditia este indeplinita) se executa instructiune1; daca valoarea expresiei este 0 (conditia nu este indeplinita), se executa instructiune2. Deci, la un moment dat, se executa doar una dintre cele doua instructiuni: fie instructiune1, fie instructiune2. Dupa executia instructiunii “if” se trece la executia instructiunii care urmeaza acesteia.

Observaţii:

1. Instructiune1 si instructiune2 pot fi instructiuni compuse (blocuri), sau chiar alte instructiuni if (if-uri imbricate).
2. Deoarece instructiunea if testeaza valoarea numerica a expresiei (conditiei), este posibila prescurtarea: if (expresie), in loc de if (expresie != 0).
3. Deoarece ramura else a instructiunii if este optionala, in cazul in care aceasta este omisa din secventele if-else imbricate, se produce o ambiguitate. De obicei, ramura else se asociaza ultimei instructiuni if.

b. Instructiunea switch

In unele cazuri este necesara o decizie multipla speciala. Instructiunea switch permite acest lucru.

Se testeaza daca valoarea pentru expresie este una dintre constantele specificate (expr_const_1, expr_const_2, etc.) si se executa instructiunea de pe ramura corespunzatoare. In schema logica test_expresie este una din conditiile: expresie=expr_const_1, expresie=expr_const_2, etc.

Sintaxa:

```
switch (expresie)
{
case expresie_const_1:  instructiune_1;
    [break;]
case expresie_const_2:  instructiune_2;
    [break;]
. . . . . .
case expresie_const_n-1:  instructiune_n-1;
    [break;]
[ default: instructiune_n; ]
}
```

Este evaluata expresie (expresie aritmetica), iar valoarea ei este comparata cu valoarea expresiilor constante 1, 2, etc. (expresii constante=expresii care nu contin variabile). In situatia in care valoarea expresie este egala cu valoarea expr_const_k, se executa instructiunea corespunzatoare acelei ramuri (instructiune_k). Daca se intalneste instructiunea break, parcurgerea este intrerupta, deci se va trece la executia primei instructiuni de dupa switch. Daca nu este intalnita instructiunea break, parcurgerea continua. Break-ul cauzeaza deci, iesirea imediata din switch.

In cazul in care valoarea expresiei nu este gasita printre valorile expresiilor constante, se executa cazul marcat cu eticheta default (cand acesta exista). Expresiile expresie, expresie_const_1, expresie_const_2,etc., trebuie sa fie intregi.

STRUCTURA LINIARA –Aplicatii Rezolvate

Exemplu : interschimbarea a doua numere intregi a si b.

PSEUDOCOD	// **Interschimbare in C++**

algortim Interschimbare
a, b, c intregi;
citeste a, b;
scrie a, b;
c = a;
a = b;
b = c;
scrie a, b;
sfarsit algoritm
	#include <iostream>
	using namespace std;
	int main ()
	{
	int a,b,c;
	cin>>a>>b; // aici citim a si b
	cout << endl<<a<<b;
	c=a;
	a=b;
	b=c;
	cout << endl<<a<<b;
	}

STRUCTURA ALTERNATIVA –Aplicatii rezolvate

Exemplu :maximul dintre doua numere intregi

algoritm maxim
a, b, maxi intregi;
citeste a, b;
daca (a>b)
maxi=a
altfel
maxi=b;
scrie maxi;
sfarsit algoritm
	// **Maxim intre 2 numere**
	#include <iostream>
	using namespace std;
	int main ()
	{
	int a,b,maxi;
	cin>>a>>b; // aici citim a si b
	if (a>b)
	maxi=a;
	else
	maxi=b;
	cout << endl<<maxi;
	}

STRUCTURA REPETITIVA

Exista trei tipuri de structuri repetitive:

1) Structura cu numar cunoscut de repetitii (**FOR**)
2) Structura cu numar necunoscut de repetitii si cu test initial (**WHILE**)
3) Structura cu numar necunoscut de repetitii si cu test final (**DO-WHILE**)

STRUCTURA REPETITIVA CU NUMAR CUNOSCUT DE PASI - FOR

Exemplu :Calculul sumei primelor n numere naturale

algoritm suma
n,i,s intregi;
citeste n;
s=0;
pentru i=1,n executa
s=s+i;
scrie s;
sfarsit algoritm
	// suma primelor n numere naturale
	#include <iostream>
	using namespace std;
	Int main ()
	{
	int n,i,s=0;
	cin>>n; // citim n
	for (i=1; i<=n; i++)
	s=s+i;
	cout << endl<<s;
	}

STRUCTURA REPETITIVA CU TEST INITIAL WHILE –Aplicatii rezolvate

Exemplu : Calculul celui mai mare divizor comun a doua numere naturale date

Cmmdc(a,b)
a, b, rest intregi;
citeste a, b;
rest=a %b
cat timp rest !=0 executa
  a=b
  b=rest
  rest=a % b;
■
scrie b;
sfarsit algoritm
	// **cmmdc(a,b)**
	#include <iostream>
	using namespace std;
	int main ()
	{
	int a,b,rest;
	cin>>a>>b; // aici citim a si b
	rest=a%b;
	while (rest!=0)
	{ a=b; b=rest;
	rest=a%b;
	}
	cout << endl<<b;
	}

STRUCTURA REPETITIVA CU TEST FINAL DO - WHILE

Exemplu : Determinarea cmmdc(a,b)

Cmmdc(a,b)
a, b, rest intregi;
citeste a, b;
repeta
  rest=a %b
  a=b
  b=rest
cat timp rest !=0
■
scrie a;
sfarsit algoritm
	// **cmmdc(a,b)**
	#include <iostream>
	using namespace std;
	int main ()
	{
	int a,b,rest;
	cin>>a>>b; // aici citim a si b
	do
	{ rest=a%b;
	a=b;
	b=rest;
	}
	while (rest!=0)
	cout << endl<<a;
	}

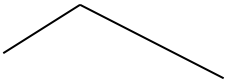
Structura liniara si alternativa-probleme propuse

1) Dandu-se doua numere naturale cu maxim patru cifre in variabilele a si b, se cere sa se afiseze cate numere impare sunt mai mici sau egale decat b si mai mari sau egale decat a.
2) Sa se determine cate numere din intervalul [a,b] sunt divizibile cu un numar dat k (a,b,k se vor citi din fisierul de intrare divizibile.in , iar rezultatul se va afisa in divizibile.out).
3) Determinati numarul numerelor naturale mai mici sau egale decat n care sunt divizibile cu 2 sau 3.
4) Sa se determine ultima cifra a lui 2^n.
5) O broscuta face in fiecare minut cate un salt.Lungimea primului salt este p (*valoare citita din fisierul de intrare broscuta.in*) dupa care, fiecare salt are lungimea dubla fata de lungimea saltului anterior facut.Realizati un program care afiseaza in fisierul de iesire broscuta.out, distanta totala parcursa de broscuta in m salturi (*m citit din fisierul broscuta.in*)

Ex: p=2,m=5, se va afisa 62(2+4+8+16+32)

6) Ordonati crescator trei numere naturale date.

7) Se citeste o valoare x, numar natural.Sa se realizeze un program care va afisa cele mai apropiate doua numere naturale pare de numarul x.
   Ex: x=14, se va afisa 12 si 16
   x=15, se va afisa 14 si 16
8) Realizati un program care citeste din fisierul de intrare o valoare x*(x<365*).Afisati pe ecran in ce luna a anului se afla ziua cu numarul x. Ex: x=33, se va afisa "februarie"
9) ***Problema "Cadoul***"
   Irina vrea sa cumpere doua carti pentru ziua de nastere a prietenei, Ana.Dar Irina trebuie sa aleaga doua carti dintre cele trei carti pe care si le doreste Ana. Stiind ca Irina dipune de s lei si ca doreste sa-i ramana cat mai multi lei dupa cumpararea celor doua carti, astfel incat sa poata cumpara si niste flori frumoase, scrieti un program prin care sa o ajutati sa aleaga cele doua carti. Suma de care dispune ea, precum si preturile celor trei carti se vor introduce de la tastatura.Programul va afisa numerele de ordine ale cartilor din care se compune cadoul.Restrictii : toate valorile sunt numere naturale mai mici sau egale cu 1000000.
10) Se introduc de la tastatura trei numere naturale mai mici decat 10000.Sa se rearanjeze cele trei numere pe ecran astfel incat sa se formeze urmatoarele forme de relief

PODIS ———— **MUNTE** 

11) In fiecare zi lucratoare din saptamana, Pinochio spune cate o minciuna datorita careia ii creste nasul cu cate p cm pe zi.Sambata si duminica, cand vine bunicul Gepeto acasa, pentru a nu-l supara prea tare, Pinochio reuseste sa nu spuna nici o minciuna, ba chiar uitandu-se in oglina observa ca in fiecare din aceste zile lungimea nasului sau scade cu cate 1 centimetru pe zi.Cand incepe o noua saptamana, ramanand singur acasa, Pinochio continua sirul minciunilor.
    Care este dimensiunea nasului lui Pinochio dupa k zile, stiind ca initial nasul sau masura n centimetri? (n,p,k se citesc de la tastatura)
12) Sa se determine cele mai mari doua numere dintre trei numere naturale introduse de la tastatura.

13)http://liis.ro/~infosuport -fisa structura liniara si fisa structura alternativa

Tema

http://varena.ro problemele: cifra, balaur, parola
http://campion.edu.ro problemele: xyz,bancomat,psp

Trimiteţi soluţiile pe adresa maftei03@gmail.com sub forma unei arhive denumită cu numele vostru.
Creaţi arhiva urmând paşii:

1. Creaţi un folder cu **numelevostru_tema2**
2. Copiati una câte una sursele **main.cpp** în acest folder şi redenumiţi-le cu numele problemei
3. Arhivaţi acest folder pastrand numele arhivei identic cu al folderului
4. Ataşaţi arhiva la email-ul pe care îl trimiteţi la adresa maftei03@gmail.com

Termen: 22 nov 2015, ora 21
SUCCES!