**Gimnaziul „ M. sadoveanu „ pleşeni, cantemir**

PROGRAMARE

PENTRU

ÎNCEPĂTORI

2009

 **Baze de numeraţie şi elemente de logică**

**Conspecte de reper pentru profesori şi elevi**

****

Gimnaziul “ M. sadoveanu „ Pleşeni , Cantemir

 **Baze de numeraţie şi elemente de logică**

**Pentru clasa VII**

### Sisteme de numeraţie în baza 2

Orice informaţie poate fi codificată folosind doar două simboluri: 0 şi 1. Să considerăm pentru început **sistemul de numeraţie în baza 2,** numit şi **sistemul de numeraţie binar** .

**Trecerea unui număr din baza 10 în baza 2 :**

Luând spre ex. nr.57 şi folosind teorema împărţirii cu rest , obţinem :

 57 2 .

56 28 2 .

 1 28 14 2 . **57( 10) = 111001( 2 )**

 0 14 7 2 .

 0 6 3 2 .

 1 2 . 1

 1

Obs. Cifrele obţinute pentru scrierea în baza 2 sunt luate începând cu ultima împărţire .

**Regulă generală :***Pentru a trece un număr din baza 10 în baza 2 se împarte numărul la 2 şi se reţine restul, apoi câtul obţinut se împarte la 2 şi se reţine restul; se continuă procedeul până se obţine câtul 0, după care numărul scris în baza 2 se obţine scriind toate resturile de la ultimul la primul .*

**Exemple :** 2 = 2(10) =10(2) ;

62(10) = 111110(2) ; 1995(10) = 11111001011(2) ; 1024(10) = 10000000000(2) ;

**Trecerea unui număr din baza 2 în baza 10 :**

 Pentru început să luăm numărul 101011(2) , care în baza 10 devine :

1x25 + 0x24 + 1x23 + 0x22 + 1x21 + 1x20 = 43, deci 101011(2) = 43 (10) = 43 .

**Regulă generală :***Unui număr în baza 2 îi corespunde în baza 10 un număr egal cu suma produselor dintre fiecare cifră şi 2 la exponentul egal cu unu mai puţin decât poziţia pe care se află cifra în numărul scris în baza 2 ( poziţia este calculată de la stânga la dreapta ) .*

 **Exemple :**

1010011(2) = 83(10) ; 1100011(2) = 227(10) ; 1000000000(2) = 512(10) ; 11001(2) =25(10)

### Sisteme de numeraţie în baza 16

Să considerăm **sistemul de numeraţie în baza 16,** numit şi **sistemul de numeraţie hexazecimal** .

**Trecerea unui număr din baza 10 în baza 16 :**

**Regulă generală :***Pentru a trece un număr din baza 10 în baza 16 se procedează astfel : se împarte succesiv la 16 numărul ( apoi câturile ş.a.m.d. ) , reţinându-se resturile , după care , dacă acestea sunt 10,11,12,13,14,15 se înlocuiesc respectiv cu A,B,C,D,E,F ; resturile obţinute formează cifrele numărului scris în baza 16 ( de la ultimul rest la primul ) .*

**Exemplu :** 2156(10)=...(16)

2156 = 134 x 16 +12 ⇒ 134 = 8 x 16 + 6 ⇒ 8 = 0 x 16 + 8 ⇒ X = 86C(16)

**Trecerea unui număr din baza 16 în baza 10 :**

**Regulă generală** : Pentru a trece un număr din baza 16 în baza 10 se procedează la fel ca la trecerea din baza 2 în baza 10, avându-se în vedere înlocuirea literelor A, B, C, D, E, F cu 10, 11, 12, 13, 14, 15.

 **Exemplu :**

 86C(16) = 8 x 162 + 6 x 16 + 12 = 2156(10)

 **Algoritmi , schemele logice**

Pentru clasa VIII

**Noţiunea de algoritm**

 Dacă în matematică noţiunea cea mai importantă este numărul, în informatică cea mai importantă noţiune cu care se lucrează este **algoritmul**.

 Vom prezenta câteva exemple care ne vor conduce la această noţiune :

**Algoritmul de aflare a sumei dintre două numere :**

*Pas 0* : START

*Pas 1* : Scriem numerele a,b

*Pas 2* : S : = a + b

*Pas 3* : Afişăm rezultatul : S

*Pas 4* : STOP .

  **Maximul şi minimum a două numere naturale**

Fie a , b є N . Vom nota cel mai mare număr cu *max* şi cel mai mic număr cu *min* . Etapele de rezolvare a problemei sunt următoarele :

*Pas 0 :* START

*Pas 1 :* Se stabilesc valorile lui *a* şi *b* .

*Pas 2 :*  Se compară a cu b ; dacă *a >= b* atunci *max* va fi egal cu *a* şi *min* cu *b* ; iar în caz contrar ( adică *a < b* ) , *max = b* şi *min = a* .

*Pas 3 :*  Se scrie rezultatul , adică *max , min* .

*Pas 4 :*  STOP .

**DEF *.*** *Prin* **algoritm** *vom înţelege o metodă de rezolvare a problemelor de un anumit tip .* **Algoritmul** *este o succesiune finită de operaţii cunoscute , care se execută într-o ordine stabilită astfel încât plecând de la un* set de date *( numite* **datele problemei** *sau* **date de intrare** *=* **D.I.** *) ce îmdeplinesc anumite condiţii , să obţinem într-un interval de timp finit un* set de valori *( numite* **soluţiile problemei** *sau* **date de ieşire** *=* **D.E.** *) .*

Nu orice descriere pas cu pas a etapelor rezolvării unei probleme reprezintă un algoritm . Pentru a fi algoritm , o metodă de rezolvare a unei probleme trebuie să aibă ***trei proprietăţi*** :

* **generalitate** ( universalitatea ) - un algoritm oferă o metodă generală de rezolvare a unui anumit tip de probleme ;
* **finitudine** – orice algoritm se încheie într-un timp după un număr finit de paşi ;
* **unicitatea** - pentru aceleaşi date de intrare se obţin totdeauna aceleaşi date de ieşere.

Reprezentarea Algoritmilor

Una din cele mai răspândite metode de reprezentare a algoritmilor este o metodă de reprezentare grafică : metoda **SCHEMELOR LOGICE** . Simbolurile grafice folosite sunt :

1. **Blocurile terminale** - pun în evidenţă începutul respectiv sfârşitul unui algoritm .
2. **Bloc de intrare/ieşire**- pune în evidenţă D.I. , respectiv D.E. ale algoritmului .
3. **Bloc de calcul** - pune în evidenţă operaţiile aritmetice şi transferul de valori între diferite date ale algoritmului .

 **Bloc de procedură**

1. **Bloc de decizie** – pune în evidenţă puncte de ramificaţie ale algoritmului . În interiorul blocului se scrie o condiţie şi în funcţie de îndeplinirea sau neîndeplinirea acesteia se va continua pe ramura cu ’’da,, sau pe ramura cu ’’nu,,.

 C

 da nu

1. **Conector**- arată punctele de intersecţie dintre ramurile unui algoritm .
2. **Săgeata** – indică un drum posibil de urmat în schema logică .

**Exemplu :** Algoritmul şi schema logică pentru suma a două numere şi aflarea minimului şi maximului a două numere .

D.I. a , b

D.E. S

 Algoritm :

Pas0 : START SCRIE a,b

Pas1 : Se scriu a , b .

Pas2 : Calculăm S : = a + b .

S = a + b

Pas3 : Afişăm rezultatul , S .

Pas4 : STOP .

 Afişează S

**Pentru clasa IX**

**Pseudocodul**

 **Pseudocodul**  este un limbaj simbolic situat între limbajul natural şi limbajul de programare obţinut prin introducerea unor reguli specifice limbajului de programare în limbajul natural. *Pseudocodul* conţine un set de cuvinte cheie cum ar fi: if, read, write, repeat, for etc. Există o corespondenţă între blocurile unei scheme logice şi cuvintele cheie ale pseudocodului :

 Un algoritm scris în pseudocod este de fapt o succesiune de instrucţiuni scrise în pseudocod .

 Instrucţiunile pseudocodului sunt :

* 1) Instrucţiunea de citire : **read** (a,b,c,…) ;
* 2) Instrucţiunea de scriere ( afişare ) : **write** (a,b,c,…) ;
* 3) Instrucţiunea de atribuire : x **: =** y + 2 ;
* 4) Instrucţiunea alternativă :

 **If**  ( cond. ) **then**

 ……instr. …

**else**

 ……instr. … ;

* 5) Instrucţiunea repetitivă repeat-until :

**repeat**

bloc de instrucţiuni

 **until** ( condiţie ) ;

* 6) Instrucţiunea repetitivă while-do :

**while** ( condiţie ) **do**

… bloc de instrucţiuni … ;

* 7) Instrucţiunea repetitivă FOR :

**for** v = vi , vf , p **do**

… bloc de instrucţiuni …;

!!! *OBSERVAŢII*

*La terminarea unei instrucţiuni se pune " ; " .*

*După " end . " de la sfârşitul algoritmului se pune " . " .*

*Înainte de " else " nu se pune punct şi virgulă .*

*După " Begin " nu se pune nici un semn .*

 **STRUCTURI FUNDAMENTALE DE CONTROL**

Orice schemă logică poate fi descompusă în trei structuri de bază şi anume:: **LINIARĂ, ALTERNATIVĂ ŞI REPETITIVĂ.**

**a) Structura liniară ( secvenţială ) :**

- este o succesiune de operaţii în care nu apar decizii .

Schema logică :

 În pseudocod : a1 ;

 a1

 a2;

 .

 .

 a2

 an;

 .

 an

 Mărimile cu care se lucrează în soluţionarea problemelor sunt variabilele şi constantele .Variabilele sunt mărimi care în timpul execuţiei unui program se modifică iar constantele rămân neschimbate .

**Variabila** este un ansamblu de patru elemente :

* ***numele variabilei*** - format din unul sau mai multe caractere (litere şi cifre , dar primul caracter este literă ) . Ex. : a , b , x , min , max , x1 , b19 , MAX
* ***tipul variabilei***

 -- indică mulţimea de valori posibile :

* întreg - integer
* real
* string sau char (şir de caractere , de ex : "A" , "+¤a" , "DORU" , "abc" )
* boolean ( true , false )

 -- indică operaţiile ce se pot efectua cu variabila , precum şi modul de reprezentare ale acesteia în memoria calculatorului .

* ***valoarea variabilei*** este valoarea efectivă pe care o are aceasta la un moment dat . O variabilă are în orice moment o singură valoare , care rămâne neschimbată până când aceasta se modifică într-o operaţie de atribuire sau citire . Operaţia de atribuire a unei valori pt o variabilă înseamnă modificarea valorii variabilei , vechea valoare a acesteia pierzându-se .
* ***adresa***din memorie unde este păstrată este locaţia fizică la care se află valoarea variabilei .

**Regula interschimbării scaunelor** : Trebuie să ne imaginăm că , în memoria calculatorului fiecare informaţie îşi are locul bine stabilit ,care se numeşte locaş (zonă ) de memorie .

 Dacă aş schimba un număr aflat într-un asemenea locaş cu un altul , atunci primul număr s-ar pierde , deoarece ar fi înlocuit . De aceea , pentru a efectua această schimbare am nevoie de un al treilea locaş de memorie în care să mut primul număr , apoi transfer al doilea număr în locul primului şi pe primul în locul celui de-al doilea . Această trecere poartă numele de **" regula celor trei scaune** " .

Rezolvarea este deci următoarea ( t - se numeşte variabilă auxiliară) :

 **1.Algoritmul 2. Schema logică**

p0) START ( **Begin** ) START

p1) citim datele problemei

 READ a,b

 (a , b )

p2) interschimbăm valorile t: = a

 a şi b a:= b

 b:= t

p3) scriem rezultatul

 WRITE a,b

 (a,b )

p4) STOP ( **End .** )

 stop

**☺PROBLEME PROPUSE**

1. *Algoritmi ce folosesc numai structuri liniare:*
* Să se calculeze suma şi produsul a trei nr. .
* Să se calculeze perimetrul şi aria unui triunghi echilateral .
* Să se calculeze perimetrul şi aria unui pătrat .
* Să se calculeze perimetrul unui triunghi isoscel .
* Să se calculeze media aritmetică a trei numere .
* Să se calculeze max. şi min. dintre două numere .
* Să se afişeze în ordine cresc. şi descresc. două numere .
* Să se calculeze perimetrul şi aria unui dreptunghi dat prin lăţime şi lungime .
* Să se calculeze lungimea şi aria unui cerc dat prin raza sa .
* Să se calculeze suma: 1+2+3+…+n , când se ştie n .
* Să se calculeze produsul primelor n nr naturale:1\*2\*3…\*n=n!
* Să se calculeze suma: k+(k+1)+…+n ,unde n şi k se cunosc (k n) .
* Să se afle suma de bani pe care o deţine un copil ştiind că ,dacă i se mai dau x lei ,cu jumătate din total poate cumpăra o culegere ce costă y lei .
* Să se calculeze media aritmetică a trei nr. .
* Dacă se dau valorile lui a,b,c,d să se calculeze :

S1=a+b+c+d

S2=a\*b+a\*c+a\*d+b\*c+b\*d+c\*d .

* Să se transforme lungimea unui obiect din m în dm. , cm. şi mm .
* Care este nr maxim de cuburi de muchie 1 dm pe care le putem ţine într-o ladă sub formă de cub cu latura de lungime L

**b) Structura alternativă IF-THEN-ELSE:**

 ***În pseudocod :***

 **C**

 **nu** **da** **if C then a1**

 **a2**

 **a1**

 **else a2 ;**

***Modul de execuţie*** *:* se testează condiţia C . Dacă aceasta este adevărată se execută blocul de operaţie a1 şi se continuă algoritmul . Dacă condiţia C este falsă se execută blocul a2 şi se continuă algoritmul .

***Obs.*** Există structuri alternative de forma : if C then

 Instructiuni;

***Exemple:***

**A1) Scrieţi trei numere sub forma unei sume de fracţii .**

if a <> 0 then

if b <>0 then

if c <>0 then v :=1/a+1/b+1/c;

**sau**

if a<>0 and b<>0 and c<>0 then

 v:=1/a+1/b+1/c;

**A2) Citindu-se trei valori reale , să se precizeze dacă sunt sau nu distincte .**

If a<>b then

 If b<>c then

 If a<>c then Write(a,b,c- sunt distincte)

else Write(a,c - nu sunt distincte)

 else Write(b,c - nu sunt distincte)

elseWrite(a,b - nu sunt distincte) ;

**A3) Să se afişeze în ordine crescătoare trei numere a,b,c є R .**

If a>b then

 If b>c then Write(a,b,c)

else Write(a,c,b)

 else If a>c then write(b,a,c)

 else If b>c then Write(b,c,a) ;

**A3) Fie numerele reale x, y şi n întreg. Dacă n<>0, să se calculeze suma nr., iar în caz contrar produsul lor.**

start

 citeşte x , y , n ;

 dacă n=0 atunci p:=x\*y ;

 scrie p

 altfel s:=x+y ;

 scrie s;

stop .

**A5)** **Să se afişeze maximul dintre 3 numere .**

if a<b then

 If b<c then Max:=c

else Max:**=**b

 else

 If a<c then Max:=c

 Else Max:=a ;

Write(max);

Structura alternativă CASE n OF

Instrucţiunea IF are posibilitatea de a selecta spre execuţie o instrucţiune sau alta . Deşi se pot face selecţii dintre mai multe instrucţiuni ( instrucţiunea IF subordonează alte instrucţiuni IF ) fondul problemei rămâne acelaşi : *se selectează o instrucţiune sau alta ( se alege dintre două variante ) .* Este necesar să existe o instrucţiune care să permită selecţia multiplă , adică se va selecta o instrucţiune din **n** instrucţiuni şi nu din două . Aceasta este instrucţiunea CASE cu forma generală :

 ***În pseudocod :***

 **Case** condiţie **OF**

C1 : instrucţiune1 ;

 C2 : instrucţiune2 ;

 …………………………

 Cn : instrucţiunen

 **ELSE**

instrucţiune ;

 **END ;**

***Modul de execuţie*** *:* Se testează condiţia . Se caută alternativa adevărată din cele n instrucţiuni în a cărei listă de constante case se regăseşte valoarea condiţiei . Dacă se regăseşte o astfel de alternativă se execută instrucţiunea respectivă . În caz contrar se execută instrucţinea din else şi se iese din structura case *.*

***Exemple:***

**C1) Tastând un caracter, verificaţi dacă aţi tastat o consoană sau vocală .**

Read(c);

Case c of

 'a' , 'e' , 'i' , 'o' , 'u' : write('vocala') ;

 'b'..'d' , 'f'..'h' , 'j'..'n' , 'p'..'t' , 'v'..'z' : write('consoana') ;

end;

**C2) Tastând o cifră , verificaţi dacă aţi tastat un nr.par sau nu .**

case i of

 0 , 2 , 4 , 6 , 8 : write('cifra para');

 1 , 3 , 5 , 7 , 9 : write('cifra impara')

else writeln ('nu ai tastat o cifra');

end;

**☺PROBLEME PROPUSE**

2**.***Algoritmi ce folosesc structuri liniare şi alternative*

* *Se cunosc numele şi mediile a trei elevi.Să se afişeze numele acestora în ordinea descrescătoare a mediilor.*
* *Să se calculeze valoarea expresiei E= 2\*x , dacă x* ∈ {0, 1, 2 } şi E=5 dacă x >=3,cunoscând valoarea lui x natural.
* *Să se calculeze suma a cinci numere din mulţimea { 1,2,3,4,10,11,12,13,14}*
* *Să se determine maximul a patru numere .*
* *Care dintre numerele ab şi ba este mai mare ?*
* *Care dintre numerele abc şi cba este mai mare ?*
* *Se dau două mulţimi A şi B fiecare având trei elemente.Care este intersecţia lor?*
* *Un elev a scris pe hârtie 5 cifre. Câte cifre distincte a scris?*
* *Să se verifice dacă două nr naturale sunt consecutive.*

**c) Structura repetitivă**

Permite executia unui bloc de operatii de un anumit numar de ori . Structurile repetitive se impart dupa :

* locul si modul de conditionare a executiei :
* structura repetitiva conditionata anterior
* structura repetitiva conditionata posterior
* dupa numarul de pasi :
* structura repetitiva cu numar cunoscut de pasi
* structura repetitiva cu numar necunoscut de pasi

a)Structura repetitivă WHILE-DO

(structură repetitivă condiţionată anterior sau cu test iniţial )

 ***În pseudocod :***

 **while C do** ( cât timp C execută

 **C**

 **DA** **a1;** a1; )

 **a1**

 **NU**

***Modul de execuţie*** *:* se evaluează condiţia C . Dacă este adevărată se executa blocul de operaţii a1 şi se reia testarea condiţiei . Dacă C este falsă se continuă algoritmul . Deci blocul a1 se execută cât timp condiţia C este adevărată ; când condiţia devine falsă se continuă algoritmul.

***Obs.*** 1) Există posibilitatea ca blocul a1 să nu se execute niciodată (când condiţia C este falsă de la început).

 2) În blocul a1 trebuie să existe o operaţie prin care să se modifice condiţia C . În caz contrar algoritmul poate intra într-un ciclu infinit .

***Exemple :***

**W1) Se citeşte nєN. Să se calculeze suma primelor n numere naturale.**S=1+2+3+…+n

Valoarea iniţială a variabilei i este 1 . Atât timp cât i este egal cu n citit , se execută o instrucţiune compusă (un bloc de instrucţiuni subordonat instrucţiunii WHILE). Aceasta cuprinde instrucţiunea prin care valoarea lui i este adunată valorii reţinute de s şi instrucţiunea prin care se adună 1 valorii reţinute de i. Vom face observaţia că blocul de instrucţiuni subordonat instrucţiunii WHILE se execută numai dacă condiţia este îndeplinită .

W2) Se citeşte un număr natural n, diferit de zero. Să se tipărească suma cifrelor sale .

 Ex. Se citeşte nr. 248 .Cum vom proceda : izolăm ultima cifră ( pe care o adunăm la o variabilă s , cu valoarea iniţială 0 ) , obţinem numărul fără ultima cifră . Aceste operaţii se repetă repetitiv numai dacă numărul este diferit de zero .

**W3) Se citeşte un număr natural n . Se cere să se decidă dacă este prim sau nu .**

Se generează primul posibil divizor 2 . Spre a executa instrucţiunea subordonată WHILE acesta trebuie să îndeplinească simultan două condiţii : să fie mai mic decât jumătatea nr. şi să nu dividă numărul . Dacă s-au testat toate numerele posibile înseamnă că nr. este prim . Altfel , înseamnă că a fost găsit un divizor , caz în care numărul nu este prim .

b)Structura repetitivă REPEAT-UNTIL

(structură repetitivă condiţionată posterior sau cu test final )

 ***În pseudocod :***

 **a1**

 **repeat** ( repetă

 **nu a1** a1

 **c**

 **until C ;** până când C ; )

 **da**

***Modul de execuţie*** *:* se execută blocul a1 şi se testează condiţia C . Dacă aceasta este falsă se reia execuţia blocului a1 , iar dacă C este adevărată se continuă algoritmul .

***Obs.*** 1) Blocul de operaţii a1 se execută sigur cel puţin o dată .

2) În blocul a1 trebuie să existe o operaţie prin care să se modifice condiţia C .

***Exemple :***

R1) Se citesc pe rând de la tastatură un şir de numere naturale care se termină cu zero. Să se calculeze suma lor.

Observăm că nu cunoaştem de la început câte nr. se citesc, dar ştim că se citeşte cel puţin un nr.Deci este indicat să folosim instr. REPEAT. Condiţia din REPEAT este nr:=0(la citirea acestui nr., se termină execuţia instrucţiunii REPEAT şi se afişează suma).

 c)Structura repetitivă FOR

(structură cu variabilă contor )

 ***În pseudocod :***

 **i:=vi**

 **For i:= vi , vf , p do**

 **a1 ;**

 **i<=vf**

 **DA**

 **a1**

 **NU**

 **i:=i+p**

***Modul de execuţie :*** se atribuie variabilei contor i valoarea iniţială vI . Dacă i ≤ vf ( valoarea finală ) atunci se execută blocul a1 se modificăvariabila contor adăugând la valoarea anterioară pasul p şi se reia testarea condiţiei i ≤ vf . Când i depăşeşte valoarea finală vf se continuă algoritmul .

 ***Obs.*** Structura repetitivă FOR este o structură cu număr cunoscut de paşi .

***Exemple :***

**F1) Să se listeze numerele de la 1 la un n dat.**

 For i:= 1 to n do Write ( i:5 ) ;

**F2) Să se listeze alfabetul de la " z " la " a " .**

 For i:= 'z' down to 'a' do Write( I ) ;

**F3) Să se facă steluţe pe prima linie.**

 For i:= 1 to 79 do begin Gotoxy( i,1); Write ( ‘ \* ’ ) ; end;

**F4) Să se facă steluţe pe prima coloana.**

 For i:= 1 to 24 do begin Gotoxy( 1,i); Write ( ‘ \* ’ ) ; end;

**F5) Să se facă steluţe pe tot ecranul.**

 For i:= 1 to 79 do

 For j:=1 to 24 do

 begin

 Gotoxy( i,j); Write ( ‘ \* ’ ) ;

 end;

**☺PROBLEME PROPUSE**

3**.***Algoritmi ce folosesc structuri liniare , alternative , repetitive:*

* *Se citesc n numere .Să se calculeze suma lor.( produsul)*
* *Să se calculeze numărul de numere mai mari ca* ***a*** *dintr-o mulţime cu* n *numere.*
* *Să se determine maximul(minimul) a n numere ce se citesc de la tastatură.*
* *Să se determine numărul de nr nenule pentru mai multe numere citite,ultimul fiind 10.*
* *Se citesc numere naturale până la întâlnirea nr200. Să se determine suma numerelor care sunt mai mici ca b şi mai mari ca a.*
* *Să se calculeze media aritmetică a numerelor citite până la întâlnirea numărului 0.*
* *Se citesc numere până la întâlnirea a două numere consecutive egale.Să se calculeze suma numerelor mai mari ca a.*
* *Se citeşte la început un număr a, apoi mai multe numere, ultimul fiind 0 .să se determine câte dintre acestea sunt egale cu a.*
* *Se citesc n numere.Să se verifice dacă este mai mare suma primelor k numere sau suma următoarelor n-k numere ?*
* *Să se verifice dacă un număr natural este prim.*
* *Să se afişeze toţi divizorii unui număr natural .*
* *Să se genereze toate numerele prime mai mici decât b şi mai mari decât a.*

-- ***STRUCTURA UNUI PROGRAM TP***

 Vom învăţa să introducem , să rulăm şi să concepem programe scrise în TP. Pentru a uşura munca programatorilor Pascal firma Borland a pus la dispoziţia acestora **mediul integrat de programare TP**. Acesta cuprinde: editor de texte, compilator şi meniuri. Toate concură la realizarea unui program şi se găsesc la un loc în cadrul programului *turbo.exe .* COMPONENTELE TP :

 **Editorul de texte** este acea componentă software care permite introducerea textelor diverselor programe .

 **Compilatorul**  este acea componentă software care traduce textul programului introdus de noi în limbaj maşină ( succesiune de 0 şi 1 ) .

 Prin **meniu** se înţelege o succesiune de cuvinte , fiecare cuvânt având semnificaţia de comandă dată calculatorului , pentru ca acesta să efectueze o anumită operaţie .

 După apel ( al programului turbo.exe ) ecranul devine albastru , iar deasupra apar:

* **un nume** ( NONAMExx - fără nume , iar xx este un număr cu două cifre ) ;
* **meniul** - mai multe cuvinte scrise orizontal în partea de sus ( FILE , EDIT…);
* **cursorul** aflat sub meniu - o liniuţă orizontală care clipeşte şi indică în permanenţă poziţia de pe ecran în care va apare primul caracter tastat .

După apelul de mai sus se poate introduce textul programului pe care vrem să-l rulăm ( să fie executat de către calculator ) .

**STRUCTURA UNUI PROGRAM Pascal :**

PROGRAM nume\_program ;

USES listă\_de\_unituri ;

CONST listă\_de\_constante ;

TYPE definiţie de tipuri ;

VAR listă de variabile ;

Procedure nume\_procedură ;

Function nume\_funcţie ;

Begin { program principal }

…set de instrucţiuni

END.

*Obs :* Numele unui program , al unei variabile , al unui tip de date , al unei proceduri , funcţii **nu** conţine caracterul spaţiu (" space") - poate conţine numai semnul \_ .

***INSTRUCŢIUNI PASCAL CORESPUNZĂTOARE STRUCTURII LINIARE***

 a) instrucţiuni de citire şi scriere a datelor :

☻ READ - cursorul rămâne pe linia unde s-a citit data

☻ Readln - cursorul trece la linia următoare .

La fel şi pentru write ( ln ) .

Ex.

 program adunare ;

 Var a , b , S : integer ;

 begin

write ('a=') ;readln ( a) ;

write ('b=') ;readln ( b) ;

S : = a+b ;

write ( a,’ + ’,b,’ = ’,S ) ;

 end.

 b) instrucţiunea de calcul :=

 c) instrucţiunea compusă (când am cel puţin două operaţii)

 begin

 set de instrucţiuni

 end;

Pentru toate problemele de la lecţiile anterioare se fac programele respective .

***INSTRUCŢIUNI PASCAL CORESPUNZĂTOARE STRUCTURII ALTERNATIVE***

 a) If condiţia then secv1

 else secv2

 b) **Case** condiţie **OF**

C1 : instrucţiune1 ;

 C2 : instrucţiune2 ;

 …………………………

 Cn : instrucţiunen

 **ELSE** instrucţiune

 **End;**

***INSTRUCŢIUNI PASCAL CORESPUNZĂTOARE STRUCTURII REPETITIVE***

 a) **while C do** **a1;**

 b) **repeat**

 **a1** **until C ;**  c) **For i:= vi , vf , p do a1 ;**