PASCAL alfabēts

lielie un mazie latīņu alfabēta burti;
arābu cipari 0..9;
speciālie simboli +-*/=.,:;<>{}()[]^$#;
atstarpe;
sapārotie simboli := <> <= >= .. (. .) (* *)

PASCAL programmas struktūra

PROGRAM <programmas nosaukums>;
USES <pieslēdzamās bibliotēkas (Units)>;
LABEL <globālo iezīmju apraksts>;
CONST <globālo konstanšu apraksts>;
TYPE <globālo mainīgo tipu apraksts>;
VAR <globālo mainīgo apraksts>;
PROCEDURE <procedūru apraksts>;
FUNCTION <funkciju apraksts>;
BEGIN
    <galvenā programmas daļa>
END.

Veselo skaitļu tipi

Tips	Diapazons	Aizņemtais atmiņas apjoms
Shortint	-128..127	1 baits
Integer	-32768..32767	2 baiti
Longint	-2147483648..21477483647	4 baiti
Byte	0..256	1 baits
Word	0..65535	2 baiti

Darbības

+	saskaitīšana
-	atņemšana
*	reizināšana
/	dalīšana
DIV	dalīšana bez atlikuma
MOD	atlikums

Procedūras

DEC(x,[n]) – samazina x vērtību par n, ja nav noteikts, tad par 1
INC(x,[n]) – palielina x vērtību par n, ja nav noteikts, tad par 1

Reālo skaitļu tipi

Tips	Diapazons	Ciparu skaits mantisā	Aizņemtais atmiņas apjoms
Real	2.9E-39..1.7E38	11-12	6 baiti
Single	1.5E-45..3.4E38	7-8	4 baiti
Double	5.0E-324..1.7E308	15-16	8 baiti
Extended	3.4E-4932..1.1E4932	19-20	10 baiti
Comp	$-2^{63}+1..2^{63}-1$	19-20	8 baiti

Piezīme. Lai lietotu pēdējos 4 reālos tipus, iepriekš jābūt direktīvai {$N+}

Loģiskais tips

Tips	Aizņemtais atmiņas apjoms
Bolean	1 baits
ByteBool	1 baits
WordBool	2 baiti
LongBool	4 baiti

A	B	NOT A	A OR B	A AND B	A HOR B
TRUE	TRUE	FALSE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE	TRUE
FALSE	TRUE	TRUE	TRUE	FALSE	TRUE
FALSE	FALSE	TRUE	FALSE	FALSE	FALSE

Simbolu tips CHAR (1 baits)

Simbolu virknes ar mainīgu garumu STRING (līdz 255 simboliem)

Lietotāja definētie tipi

Intervāla tips
TYPE burts='A'..'Z';
VAR a:burts;
    b:10..1000;

Uzskaitāmais tips
TYPE diena=(Pirmdiena,Otrdiena,Tresdiena,Ceturtdiena,Piektdiena,Sestdiena,Svetdiena)
VAR a:diena;
    b:(sarkans,dzeltens,zals)

Uzskaitījuma tipam pieļaujamas tikai salīdzināšanas operācijas, tam pielietojamas standartfunkcijas:
succ(x) – nākamais elements aiz x
pred(x) – iepriekšējais elements pirms x
ord(x) – elementa x kārtas numurs (numurēšana sākas ar 0)

Mainīgie

Mainīgo nosaukumam jāatbilst šādām prasībām:

- tie var sastāvēt no latīņu alfabēta burtiem, cipariem vai pasvītrojuma zīmes (citi simboli nav pieļaujami);
- mainīgā nosaukums var sākties tikai ar burtu;
- mainīgā nosaukums nedrīkst sakrist ar kādu no rezervētajiem atslēgvārdiem (piemēram, begin ,for, if, end);
- nosaukuma garums var būt patvaļīgs, vērā tiks ņemti pirmie 63 simboli.

Izteiksmes

Izmantojot konstantes, mainīgos, operācijas un iekavas var izveidot izteiksmes, piemēram:

(1+2)*3

(NOT x) OR y

Uzdevumi

Uzrakstīt paskālā šādas izteiksmes:

1. $\frac{a+b}{a-b}$
2. $\frac{a^2+b^2-c^2}{2ab}$
3. $\frac{1+a}{2+a}.\frac{3+a}{4+a}$

Datu izvade

WRITE(<izvadāmā vērtība>[,<izvadāmā vērtība>…]);
WRITELN(<izvadāmā vērtība>[,<izvadāmā vērtība>…]);

Pirmajā gadījumā nākošā izvade turpināsies tajā pašā rindā, otrajā gadījumā – nākošajā rindā.

Piemēri:
WRITE(`Saule`);
WRITELN(`Saule`);
WRITE(2*2);
WRITE(a,b);
WRITE(`x=`,x);

Izvadāmajām vērtībām var norādīt atvēlēto zīmju skaitu, reāliem skaitļiem - arī zīmju skaitu aiz komata.

WRITE(<izvadāmā vērtība>:<M>:<N>[,<izvadāmā vērtība>:<M>:<N>…]);
WRITELN(<izvadāmā vērtība>:<M>:<N>[,<izvadāmā vērtība>:<M>:<N>…]);
kur M – atvēlēto pozīciju skaits, N – tai skaitā zīmju skaits aiz komata.

Datu ievade

READ(<mainīgais>[,<mainīgais>…]);
READLN(<mainīgais>[,<mainīgais>…]);

Piešķires operācija

<mainīgā nosaukums>:=<izteiksme vai konstante>

Piemēram,
a:=13.5
laukums:=a*b/2

Programmu piemēri

```
var a,b:integer;
begin
  readln(a,b);
  writeln(a+b);
end.
```

```
program Summa;
var a,b,c:integer;
begin
  write('Ievadiet pirmo saskaitāmo ===>');
  readln(a);
  write('Ievadiet otro saskaitāmo ===>');
  readln(b);
  c:=a+b;
  writeln;
  writeln('Skaitļu ',a,' un ',b,' summa ir ',c);
  writeln;
  writeln('Lai turpinātu, nospiediet ENTER taustiņu');
  readln;
end.
```

```
{Programma atrod naturāla skaitļa pēdējo ciparu}
Program Pedejais_cipars;
var a:longint;
begin
  write('Ievadiet naturālu skaitli ==>');
  readln(a);
  writeln;
  writeln('Pēdējais cipars ir ',a mod 10);
end.

{Programma noskaidro, vai ievadītais skaitlis ir pozitīvs.
Ja ir, tad atbilde TRUE, citādi atbilde FALSE}
program Pozitivs;
var a:integer;
    b:boolean;
begin
    readln(a); b:=a>0; writeln(b);
end.
```

Iebūvētās funkcijas

Funkcija	Ko tā atrod ?	Argumenta tips	Rezultāta tips	Piemērs	Piemēra rezultāts
ABS(x)	Absolūtā vērtība	INTEGER REAL	INTEGER REAL	ABS(5) ABS(-5.5)	5 5.5
SQR(x)	Kvadrāts	INTEGER REAL	INTEGER REAL	SQR(5) SQR(-1.5)	25 2.25
SQRT(x)	Kvadrātsakne	INTEGER REAL	REAL	SQRT(25)	5
Pi	π		REAL	Pi	3.1415926536
SIN(x)	Sinx	INTEGER REAL	REAL	SIN(Pi/2) SIN(Pi/6)	1 0.5
COS(x)	Cosx	INTEGER REAL	REAL	COS(Pi/2) COS(Pi/3)	0 0.5
ARCTAN(x)	Arctgx	INTEGER REAL	REAL	ARCTAN(0) ARCTAN(1)	0 Pi/4
EXP(x)	e^x	INTEGER REAL	REAL	EXP(5)	e^5
LN(x)	Lnx	INTEGER REAL	REAL	LN(1) LN(EXP(3))	0 3
TRUNC(x)	Veselā daļa	INTEGER REAL	INTEGER	TRUNC(5) TRUNC(5.7) TRUNC(-1.7)	5 5 -1
INT(x)	Veselā daļa	INTEGER REAL	REAL	INT(5) INT(5.7) INT(-1.7)	5 5 -1
FRAC(x)	Daļveida daļa	REAL	REAL	FRAC(5) FRAC(5.7) FRAC(-1.7)	0 0.7 -0.7
ROUND(x)	Noapaļo līdz tuvākajam veselam skaitlim	INTEGER REAL	INTEGER	ROUND(1) ROUND(1.4) ROUND(1.6)	1 1 2
ODD(x)	Paritāte	INTEGER	BOOLEAN	ODD(5) ODD(6)	TRUE FALSE
ORD(x)	ASCII kods	CHAR	INTEGER	ORD(`A`) ORD(`0`)	65 48
CHR(x)	Simbols, kura ASCII kods ir x	Veseli skaitļi no 0 līdz 255	CHAR	CHR(65) CHR(48)	A 0
RANDOM	Gadījuma skaitlis robežās no 0 līdz 1		REAL		
RANDOM(x)	Vesels gadījuma skaitlis robežās no 0 līdz x-1.	INTEGER	INTEGER	Piezīme: Lai gadījuma skaitļi netiktu sākti ģenerēt ar vienu un to pašu vērtību, tad tie „jāsamaisa" ar procedūru RANDOMIZE.	

Uzdevumi

1. Sastādīt programmu, kas aprēķina riņķa līnijas garumu, ja ievadīts tiek rādiuss.
2. Sastādīt programmu, kas aprēķina taisnstūra diagonāli, ja ievadīti tiek malu garumi.
3. Sastādīt programmu, kas aprēķina trijstūra laukumu, ja ievadīti tiek malu garumi.
4. Sastādīt programmu, kas aprēķina trijstūra augstumus, ja ievadīti tiek malu garumi.
5. Sastādīt programmu, kas naturāla skaitļa pēdējo ciparu aizstāj ar 0.
6. Sastādīt programmu, kas izvada naturāla skaitļa pēdējo divu ciparu veidoto skaitli.
7. Sastādīt programmu, kas atrod naturāla skaitļa priekšpēdējo ciparu.
8. Sastādīt programmu, kas atrod ievadītā reālā skaitļa pirmo ciparu aiz komata.
9. Sastādīt programmu, kas ievadīto skaitli noapaļo līdz desmitdaļām.
10. Sastādīt programmu, kas milimetrus pārvērš metros, centimetros un milimetros.
11. Sastādīt programmu, kas sekundes pārvērš stundās, minūtēs un sekundēs. Piemēram, 70s=0h 1min 10s.

Darbs ar simboliem un simbolu virknēm

Darbība

<1. simbolu virkne>+<2.simbolu virkne> - divu virkņu secīga saskaitīšana vienā veselā.

Funkcijas

Funkcija	Ko tā atrod ?	Argumenta tips	Rezultāta tips	Piemērs	Piemēra rezultāts
ORD(x)	Simbola x kārtas numurs (ASCII kods)	CHAR	INTEGER	ORD(`A`) ORD(`0`)	65 48
CHR(x)	Simbols, kura kārtas numurs (ASCII kods) ir x	Veseli skaitļi no 0 līdz 255	CHAR	CHR(65) CHR(48)	A 0
PRED(x)	Simbols, kurš sarakstā ir pirms x	CHAR	CHAR	PRED(`B`)	A
SUCC(x)	Simbols, kurš sarakstā ir aiz x	CHAR	CHAR	SUCC(`B`)	C
LENGTH(x)	Simbolu virknes garums	STRING	INTEGER	LENGTH(`Kaķis un pele`	13
COPY(x,n,k)	Izkopē no simbolu virknes x tās fragmentu, sākot no n-tā simbola garumā k	x-STRING n,k-INTEGER	STRING	COPY(`logaritms`,2,3)	oga
POS(x,y)	Atrod simbolu virknē y pirmā elementa pozīcijas numuru, ar kuru sākot simbolu virkne x pilnībā ietilpst simbolu virknē y. Gadījumā, ja simbolu virkne x neietilpst simbolu virknē y, funkcija dod rezultātu 0.	x,y-STRING	INTEGER	POS(`un`,`suns`) POS(`ABD`,`ABCD`)	2 0
CONCAT(x1,x2,…)	Dara to pašu, ko x1+x2+…	STRING	STRING	CONCAT(`u`,`n`)	un

Vienu simbolu no simbolu virknes var izkopēt arī šādi:
<mainīgā nosaukums>[simbola numurs]

Piemērs

b:=a[4] mainīgajam b piešķir simbolu virknes a 4. simbolu.

Līdzīgi var nomainīt simbolu virknes vienu elementu:
<mainīgā nosaukums>[simbola numurs]:=<jaunais simbols>

Piemēram

a[4]:=`*` nomaina simbolu virknes a 4. simbolu pret zvaigznīti.

Procedūras

DELETE(x,n,k) – izmet no simbolu virknes x k simbolus, sākot ra n-to.
Piemērs
a:=`LOGARITMS`;
DELETE(a,3,5); {Mainīgā a vērtība ir `LOMS`}

INSERT(x,y,n) – simbolu virknē y, sākot ar n-to pozīciju, iesprauž simbolu virkni x.
Piemērs
a:=`LOMS`;
INSERT(`GARIT`,a,3); {Mainīgā a vērtība ir `LOGARITMS`}
STR(a,x) – skaitli a pārvērš par simbolu virkni x.
Piemēri
STR(5,x) x=`5`
a:=2.5
STR(a:5:2,x) x=` 2.50`

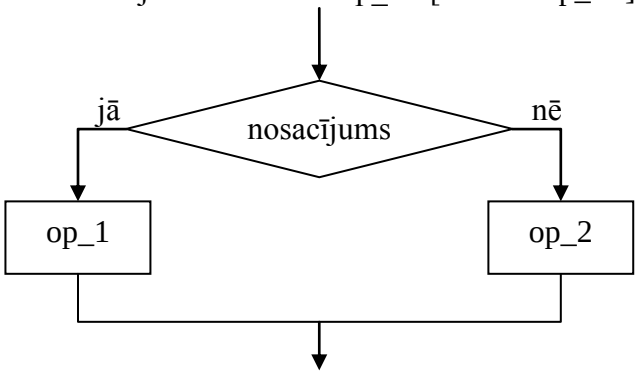
VAL(x,a,k) – simbolu virkni x pārvērš par skaitli a. k – kļūdu detektors – ja procedūra beigusies sekmīgi, tad k pieņem vērtību 0, bet pretējā gadījumā k norāda tās pozīcijas numuru, kurā atrasts pirmais skaitlim neraksturīgais simbols.
Piemēri
VAL(`12345`,a,k) a=12345 k=0
x:=`2.5E4`
VAL(x,a,k) ja a:INTEGER, tad a=2500 k=0

Uzdevumi

1. Sastādīt programmu, kas izvada ievadītā simbola kodu.
2. Sastādīt programmu, kas izvada ievadītajam kodam atbilstošu simbolu.
3. Sastādīt programmu, kas
   - pieprasa ievadīt jebkuru simbolu;
   - izvada ievadīto simbolu un tā kodu;
   - izvada simbolus, kas atrodas pirms un pēc ievadītā simbola, un to kodus.
4. Sastādīt programmu, kas izvada ievadītā teksta pirmo simbolu.
5. Sastādīt programmu, kas izvada ievadītā teksta pēdējo simbolu.
6. Sastādīt programmu, kas izvada ievadītā teksta vidējo simbolu, ja zināms, ka ievadītājā tekstā ir nepāra skaits simbolu.
7. Sastādīt programmu, kas izvada ievadītā teksta vidējos divus simbolus, ja zināms, ka ievadītājā tekstā ir pāra skaits simbolu.
8. Sastādīt programmu, kas ievadītā teksta pirmo simbolu nomaina pret `-`.
9. Sastādīt programmu, kas ievadītā teksta pēdējo simbolu nomaina pret `-`.
10. Sastādīt programmu, kas ievadītā teksta pēdējo simbolu nomaina pret `…`.
11. Sastādīt programmu, kas izdzēš ievadītā teksta vidējo simbolu, ja zināms, ka ievadītājā tekstā ir nepāra skaits simbolu.
12. Sastādīt programmu, kas izloka pirmās deklinācijas lietvārdu.

Sazarošanās operators

IF <nosacījums> THEN <op_1> [ELSE <op_2>]



Ja op_1 vai (un) op_2 vietā nepieciešams lietot vairākus operatorus, tad tie jāliek operatoru iekavās, t.i., šie operatori jāraksta starp BEGIN un END (aiz END pirms ELSE netiek likts semikols)

Vienkāršos nosacījumos izmanto salīdzināšanas operācijas: = < > <= >= <>.

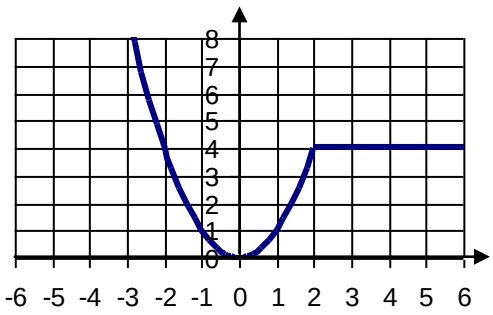
Saliktos nosacījumus veido no vienkāršajiem nosacījumiem, izmantojot darbības OR, AND, NOT, HOR (sk. 2. lpp.) un katru vienkāršo nosacījumu liekot iekavās.

Piemēri

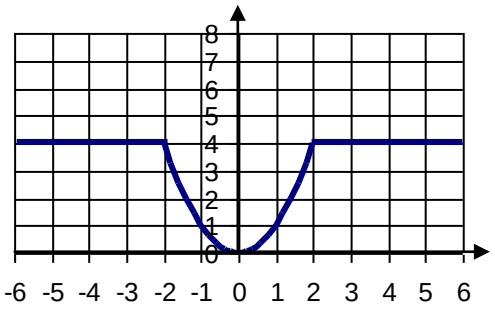
```
{Programma pārbauda, vai lietotājs prot 2*2}
program DivReizDivi;
var atb:integer;
begin
    writeln('Cik ir 2*2 ?');
    readln(atb);
    if atb=4 then writeln('Pareizi') else
writeln('Nepareizi');
end.

program Mikla;
var atb:string;
begin
    writeln('Atmini mīklu:');
    writeln('Mazs, mazs vīriņš, ass, ass cirvītis.');
    readln(atb);
    if (atb='Bite') or (atb='bite') or (atb='BITE') then
      writeln('Pareizi')
    else begin
      writeln('Nepareizi');
      writeln('Kā var neuzminēt tik vienkāršu mīklu!');
    end;
end.
```

Uzdevumi

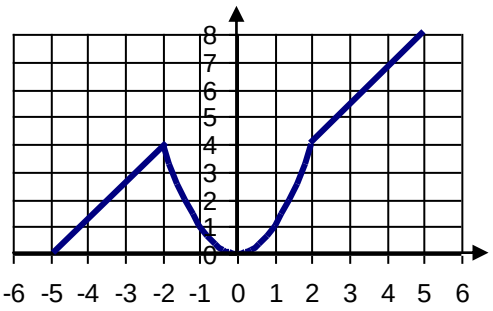
1. Sastādīt programmu, kas aprēķina funkcijas $f(x)=\begin{cases} x^3, \text{ja } x<1 \\ x, \text{ja } x\geq 1 \end{cases}$ vērtību jebkurai lietotāja ievadītai x vērtībai.
2. Sastādīt programmu, kas noskaidro, vai ievadītais skaitlis ir pāra vai nepāra.
3. Sastādīt programmu, kas noskaidro, vai ievadītais skaitlis dalās ar 3.
4. Sastādīt programmu, kas aprēķina funkcijas vērtību jebkurai lietotāja ievadītai x vērtībai, ja funkcija dota ar grafiku:



a)



b)



c)

5. Sastādīt programmu, kas lietotāju uzaicina ievadīt savu vārdu un tad ar viņu sasveicinās, piemēram, `Labdien, Juri!`.
6. Sastādīt programmu, kas par ievadīto skaitli pasaka, vai tas pozitīvs, negatīvs, vai nulle.
7. Dotas riņķa līnijas centra koordinātas un rādiuss. Dots arī punkts ar savām koordinātām. Sastādīt programmu, kas nosaka, vai punkts atrodas riņķa līnijas iekšpusē, ārpusē vai uz riņķa līnijas.
8. Dotas 3 punktu koordinātas. Sastādīt programmu, kas noskaidro, vai šie punkti atrodas uz vienas taisnes.
9. Dotas punktu A,B,C,D koordinātas. Sastādīt programmu, kas noskaidro, vai četrstūris ABCD ir paralelograms.
10. Sastādīt programmu, kas atrod mazāko no diviem (trim, četriem) skaitļiem.
11. Sastādīt programmu, kas sakārto dotos 3 skaitļus nedilstošā kārtībā.
12. Sastādīt programmu, kas atrisina kvadrātvienādojumu $ax^2+bx+c=0$ (a,b,c tiek ievadīti).
13. Sastādīt programmu, kas atrisina bikvadrātvienādojumu $ax^4+bx^2+c=0$ (a,b,c tiek ievadīti).
14. Sastādīt programmu, kas noskaidro, vai ievadītie 3 skaitļi var kalpot par trijstūra malu garumiem.
15. Par automorfu sauksim tādu naturālu skaitli n, kura kvadrāts beidzas ar n. Piemēram, 5 (25), 6 (36), 25 (625). Sastādīt programmu, kas noskaidro, vai dotais skaitlis ir automorfs.

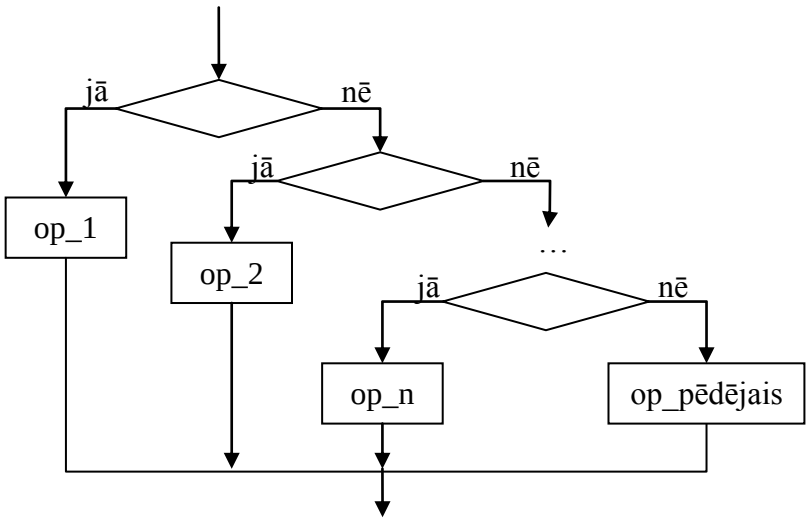
16. Sastādīt programmu, kas atrod vienu atrisinājumu veselos skaitļos sistēmai

$$\begin{cases} a < x < b \\ c < y < d \\ e < x + y < f \\ g < x - y < h \end{cases}$$

, ja a,b,c,d,e,f,g,h – veseli skaitļi, kas tiek ievadīti un, kas atrodas intervālā $[-10^8\text{-}10^8]$.

17. Sastādīt programmu, kas pēc ievadītā nedēļas dienas numura uzraksta dienas nosaukumu. Piemēram, ja ievadīts 3, jāuzraksta TREŠDIENA. Ja ievadīts neatbilstošs skaitlis, jāizvada: TĀDAS DIENAS NAV.

Varianta operators



CASE <izteiksme> OF
    <gadījums_1>:<op_1>;
    <gadījums_2>:<op_2>;
    …
    <gadījums_n>:<op_n>;
[ELSE <op_pēdējais>]
END;

Piemērs

```
program Simboli;
var Ch:char;
begin
    writeln('Ievadiet vienu simbolu');
    readln(Ch);
case Ch of
  'A'..'Z', 'a'..'z':  WriteLn('Burts');
  '0'..'9':            WriteLn('Cipars');
  '+', '-', '*', '/':  WriteLn('Darbība');
else
  WriteLn('Speciāls simbols');
end;
end.
```

Uzdevumi

1. Sastādīt programmu, kas pēc ievadītā nedēļas dienas numura uzraksta dienas nosaukumu. Piemēram, ja ievadīts 3, jāuzraksta TREŠDIENA. Ja ievadīts neatbilstošs skaitlis, jāizvada: TĀDAS DIENAS NAV.
2. Sastādīt programmu, kas uzaicina ievadīt divus skaitļus un darbību (+ - * /), un izvada rezultātu.
3. Sastādīt programmu, kas divciparu skaitli pārvērš romiešu pierakstā.
4. Sastādīt programmu, kas, divciparu skaitli no romiešu pieraksta pārvērš arābu pierakstā.

Operators GOTO

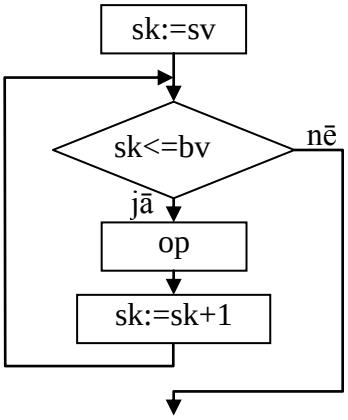
GOTO <iezīme> - notiek pāreja uz iezīmi. (Iezīmes veido no cipariem un burtiem un tās ir jāapraksta sadaļā LABEL)

Piemērs.

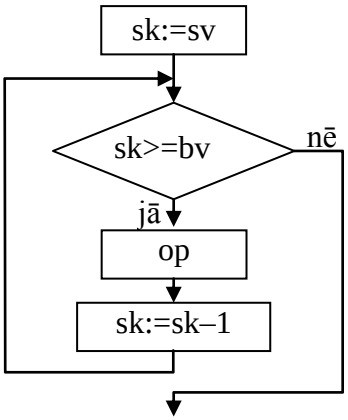
```
program Kvadratskane;
var a:real;
label b;
begin
b:writeln('Ievadiet nenegatīvu skaitli');
  readln(a);
  if a<0 then goto b;
  writeln('Kvadrātsakne no ',a:5:2,' ir',sqrt(a):5:2)
end.
```

Cikla operators ar skaitītāju

FOR <sk>:=<sv> TO <bv> DO <op>



FOR <sk>:=<sv> DOWNTO <bv> DO <op>



Piemēri.

```
program Saule;
var i:integer;
begin
  for i:=1 to 20 do writeln('saule');
end.
```

```
program Numurs;
var i:integer;
begin
  for i:=1 to 20 do writeln(i);
end.
```

```
program Klase;
var i:integer;
begin
  for i:=1 to 12 do writeln(i,'. klase');
end.
```

```
program Saule_lietus;
var i:integer;
begin
  for i:=1 to 10 do begin
    writeln('saule');
    writeln('lietus');
  end;
end.

program Alfabets;
var i:char;
begin
  for i:='A' to 'Z' do write(i:2);
end.
```

Uzdevumi

1. Sastādīt programmu, kas uz ekrāna izvada:

a)

1	1
2	4
3	9
…	…
20	400

b)

1	20
2	19
3	18
…	…
20	1

c)
1*1=1
2*2=4
3*3=9
…
9*9=81

d) Stabiņā skaitļus no 1 līdz 20, izņemot 10.

e)

1	n
2	p
3	n
…	…
20	p

f)
1
2
3
1
2
3
…
1
2
3
} 18 rindiņas

2. Izvadīt 20 gadījuma skaitļus intervālā [0,1].
3. Izvadīt 20 gadījuma skaitļus intervālā [0,10].
4. Izvadīt 20 gadījuma skaitļus intervālā [-5,5].
5. Izvadīt 20 veselus gadījuma skaitļus no 1 līdz 6.
6. Modulēt monētas mešanu 20 reizes, uz ekrāna izvadot rezultātu ģ vai c.
7. Izvadīt rindiņā n zvaigznītes (n tiek ievadīts).
8. Izvadīt r rindiņas ar n zvaigznītēm katrā (r un n tiek ievadīti).
9. Izvadīt šādu skaitļu trijstūri:
   1
   1 2
   1 2 3
   …
   1 2 …20

Piemēri

```
{Programma visus a nomaina pret b}
program Maina;
var i:byte;
    t:string;
begin
  writeln('Ievadiet tekstu');
  readln(t);
  for i:=1 to length(t) do if t[i]='a' then t[i]:='b';
  writeln(t);
end.
```

```
{Programma tekstu apgriež no otrādi}
program Apgriesana;
var i:byte;
    a,b:string;
begin
  writeln('Ievadiet tekstu');
  readln(a);
  b:='';
  for i:=1 to length(a) do b:=a[i]+b;
  writeln(b);
end.
```

10. Ievadītajā tekstā visus a nomainīt pret b un visus b – pret a.
11. Sastādīt programmu, kas uzaicina ievadīt vārdu un, izmantojot ievadīto vārdu, šādi izveido taisnstūri:

```
PĒTERIS
Ē     I
T     R
E     E
R     T
I     Ē
SIRETĒP
```

12. Atrast ievadītā skaitļa visus daītājus.

Skaitīšana un summēšana

Piemēri

$\{1^2+2^2+3^2+\ldots+n^2\}$

```
program Kvadratu_summa;
var n,i,S:integer;
begin
  write('Ievadiet n==>');
  read(n);
  S:=0;
  for i:=1 to n do S:=S+i*i;
  writeln(S);
end.
```

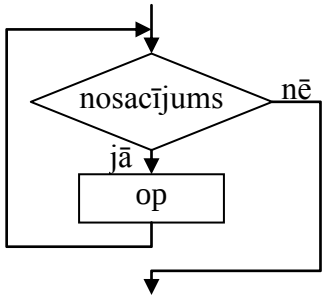
```
{Atrod burtu a skaitu dotajā tekstā}
program a_skaits;
var i,S:integer;
    t:string;
begin
  writeln('Ievadiet tekstu');
  read(t);
  S:=0;
  for i:=1 to length(t) do if t[i]='a' then inc(S);
  writeln(S);
end.
```

Uzdevumi

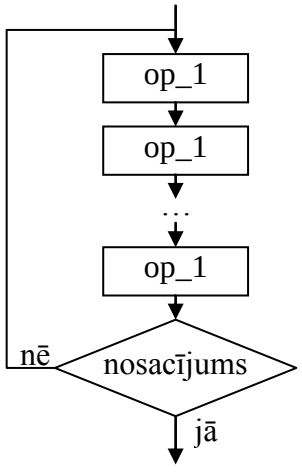
1. Saskaitīt vārdus dotajā teikumā, ja starp katriem diviem vārdiem ir tieši viena atstarpe un citu atstarpju nav.
2. Atrast divskaņu ie skaitu dotajā tekstā.
3. Aprēķināt ievadīto skaitļu vidējo aritmētisko.
4. Saskaitīt, cik no ievadītajiem skaitļiem ir pozitīvi un cik negatīvi.
5. Sastādīt programmu, kas aprēķina vidējo atzīmi. Vispirms kā simbolu virkne tiek ievadītas visas atzīmes un, nospiežot *Enter*, parādās rezultāts. Lai 10 varētu atšķirt no 1 un 0, tad atzīmes 10 vietā ievadam +.
6. Aprēķināt n! (n!=1·2·3·…·n).
7. Aprēķināt $1^2-2^2+3^2-4^2+\ldots+(-1)^{n+1}\cdot n^2$.
8. Aprēķināt $x-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\ldots+(-1)^{n+1}\frac{x^{2n-1}}{(2n-1)!}$.
9. Aprēķināt $\sqrt{1+\sqrt{2+\sqrt{3+\ldots+\sqrt{n}}}}$.
10. Aprēķināt $C_n^m=\frac{n!}{m!\cdot(n-m)!}=\frac{n(n-1)\ldots(n-m+1)}{m!}$

Atkārtijuma operatori WHILE un REPEAT

WHILE <nosacījums> DO <op>



REPEAT
    <op_1>;
    <op_2>;
    …
    <op_n>;
UNTIL <nosacījums>;



Piemēri

```
program LKD;
var a,b:integer;
begin
  writeln('Ievadiet divus naturālus skatļus');
  readln(a,b);
  while a<>b do if a>b then a:=a-b else b:=b-a;
  writeln('LKD=',a);
end.
```

```
program LKD1;
var a,b,r:integer;
begin
  writeln('Ievadiet divus naturālus skatļus');
  readln(a,b);
  repeat
    r:=a mod b;
    a:=b;
    b:=r;
  until r=0;
  writeln('LKD=',a);
end.
```

Ciklu un programmas pārtraukšana.

CONTINUE – sākt no jauna kārtējo ciklu
BREAK – iziet no cikla
} FOR, WHILE, REPEAT
HALT – beigt programmas izpildi

Uzdevumi

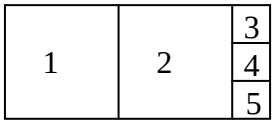
1. Sastādīt programmu, kas aprēķina pozitīvu skaitļu vidējo aritmētisko. Skaitļu skaits netiek ievadīts, bet beigās tiek ievadīta 0 (tā ir ievades beigu pazīme).
2. Sastādīt programmu, kas uzdod jautājumu "Cik ir 2*2 ?" un ļauj minēt līdz uzminēts. Kad uzminēts, pasaka "Pareizi".
3. Papildināt iepriekšējo uzdevumu, lai neļauj minēt vairāk par 3 reizēm un, ja neuzmin, lai pasaka pareizo atbildi.
4. Sastādīt programmu, kas noskaidro, vai skaitlis pieder Fibonači virknei. Ja pieder, tad noteikt tā numuru, ja nē – paziņot par nepiederību.
5. Tiek ievadīta lappušu numuru summa. Jāaprēķina lappušu skaitu grāmatā. Ja tāda numuru summa nav iespējama, par to jāpaziņo.
6. Atrast naturāla skaitļa pirmo ciparu.
7. Atrast naturāla skaitļa ciparu skaitu.
8. Atrast naturāla skaitļa ciparu summu.
9. Noskaidrot, cik piecniekus satur ievadītais skaitlis.
10. Noskaidrot, vai ievadītais skaitlis satur ciparu 5?
11. Aprēķināt summu $x-\frac{x^3}{3!}+\frac{x^5}{5!}-\frac{x^7}{7!}+\ldots$ Summēšana jābeidz, kad kārtējā saskaitāmā modulis kļuvis mazāks par doto skaitli.
12. Parādīt virkni $a_{n+1}=\begin{cases}\frac{a_n}{2}, \text{ja } a_n - \text{pāra skaitlis}\\ 3a_n+1, \text{ja } a_n - \text{nepāra skaitlis}\end{cases}$ līdz pirmajam vieniniekam.

Pilnās pārlases uzdevumi par skaitļiem

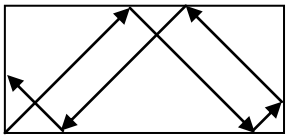
1. Cik ir tādu divciparu skaitļu, kuru ciparu kvadrātu summa dalās ar 13?
2. Atrast divciparu skaitli, kurš vienāds ar savu ciparu summas kvadrātu ?
3. Atrast visus divciparu skaitļus ar šādu īpašību: ja skaitļa ciparu summai pieskaita šīs summas kvadrātu, tad iegūst pašu skaitli.
4. Atrast visus trīsciparu skaitļus ar šādu īpašību: skaitļa kvadrāts beidzas ar 3 cipariem, kuri veido šo skaitli.
5. Atrast četrciparu skaitli, kurš dalot ar 133, dod atlikumu 125, bet, dalot ar 134, dod atlikumu 111.
6. Skaitlim 523*** pierakstīt galā 3 tādus ciparus, lai šis skaitlis dalītos ar 7, ar 8 un ar 9.
7. Skaitlim ***999 pierakstīt priekšā 3 tādus ciparus, lai iegūtais 6-ciparu skaitlis dalītos ar 13, ar 17 un ar 19.
8. Ilgdzīvotājs (t.i. cilvēks, kurš nodzīvojis vairāk par 100 gadiem) konstatēja, ka, ja viņa vecuma ciparu kvadrātu summai pieskaita viņa dzimšanas datumu (t.i., kādu skaitli no 1 līdz 31), tad rezultātā tiek iegūts viņa vecums. Cik gadu ilgdzīvotājam?
9. Atrast trīsciparu skaitli, kura kvadrāts beidzas ar 3 vienādiem nenulles cipariem.
10. Zvaigznīšu vietā ielieciet ciparus, lai piecciparu skaitlis 42*4* dalītos ar 72.
11. Atrast četrciparu pilnu kvadrātu, kura pirmie divi cipari vienādi un otrie divi cipari arī vienādi.
12. Veikalā ir smērviela 16 kg, 17 kg un 21 kg kārbās. Kā iespējams pārdot 185 kg, neatverot nevienu kārbu?
13. Gan četrciparu skaitlis gan skaitlis, kurš sastāv no šiem pašiem cipariem tikai apgrieztā kārtībā ir pilni kvadrāti. Atrast visus šādus skaitļus.
14. Atrast visus trīsciparu skaitļus, kas vienādi ar savu ciparu kubu summu.
15. Trīsciparu skaitlim nosvītroja pirmo ciparu, iegūto divciparu skaitli pareizināja ar 7 un ieguva sākotnējo trīsciparu skaitli. Atrast to.
16. Trīsciparu skaitlim, kura visi cipari nepāra, izsvītroja vidējo ciparu. Izrādījās, ka iegūtais divciparu skaitlis ir sākotnējā skaitļa dalītājs. Atrast visus šādus trīsciparu skaitļus.
17. Atrast visus trīsciparu skaitļus, kuri dalās ar 7 un kuru ciparu summa dalās ar 7.
18. Atrisināt rēbusu: KTO+KOT=TOK.
19. Atrisināt rēbusu: $ABBA=AA^2+BB^2$.
20. Atrisināt rēbusu: КИО·ИО=ТОКИО.
21. Atrast lielāko summas M2+N2 vērtību, ja M un N ir naturāli skaitļi robežās no 1 līdz 100 un tie apmierina vienādību |N2-MN-M2|=1.
22. Cik ir "laimīgo" 6-ciparu biļešu, t.i., kuru numuru pirmo 3 ciparu summa vienāda ar otro 3 ciparu summu?
23. Cik saskaitāmo summā 1+2+3+… jāņem, lai iegūtu trīsciparu skaitli, kura visi cipari vienādi.
24. Cik ir tādu veselu skaitļu starp 1 un 1988, kuri nedalās ne ar vienu no skaitļiem 6, 10, 15?
25. Atrast mazāko naturālo skaitli M, kuram piemīt īpašība: 11 pēc kārtas ņemtu naturālu skaitļu, sākot ar M, kvadrātu summa ir vesels kvadrāts.
26. Cik dažādos veidos latu var izmainīt 1, 2 un 5 santīmu monētās?
27. Virknē 19886138… katrs cipars, sākot ar piekto, ir vienāds ar iepriekšējo četru ciparu summas pēdējo ciparu. Pēc cik cipariem atkal būs 1998, t.i. cik garš periods?
28. Atrast visus divciparu skaitļus, kuru ciparu summa nemainās, ja tos reizina ar 2, 3, 4, 5, 6, 7, 8, 9.
29. Atrast lielāko un mazāko trīsciparu skaitļa attiecību pret šī skaitļa ciparu summu. Kādiem skaitļiem ir šīs mazākā un lielākā attiecības?
30. Atrast četrciparu skaitli, kurš ir četras reizes mazāks par skaitli, kurš uzrakstīts ar tiem pašiem cipariem tikai apgrieztā kārtībā.

Uzdevumi par cikliem

1. Noskaidrot, vai ievadītais skaitlis ir pirmskaitlis.
2. Noskaidrot, vai divi dotie skaitļi ir savstarpēji pirmskaitļi.
3. Sadalīt pirmreizinātājos doto skaitli.
4. Atrast lielāko no ievadītajiem skaitļiem.
5. Atrast lielāko negatīvo skaitli.
6. Atrast divus mazākos no ievadītajiem skaitļiem.
7. Cik piecnieku ir starp ievadītajiem skaitļiem?
8. Tiek ievadīti n skaitļi. Vai kādā brīdī pēc kārtas tika ievadīti 2 vienādi skaitļi?
9. Tiek ievadīta skaitļu virkne. Cik reizes tajā mainās zīme?
10. Vai ievadītā skaitļu virkne ir augoša?
11. Vai ievadītā skaitļu virkne ir maiņzīmju (katram nākošajam loceklim zīme mainās uz pretējo)?
12. Izvadīt uz ekrāna $S = \begin{cases} 2*4*...*n, \text{ ja n - pāra skaitlis} \\ 1*3*...*n, \text{ ja n - nepāra skaitlis} \end{cases}$.
13. Naturālu skaitli sauksim par ideālu, ja tas vienāds ar savu dalītāju (atskaitot pašu skaitli) summu. Piemēram, 6=1+2+3. Sastādīt programmu, kas atrod visus ideālos skaitļus līdz 1000000.
14. Simbolu virknē vairākus blakus esošos vienādos simbolus aizstāt ar vienu.
15. Dota simbolu virkne. Nošifrēt tajā visus mazos latīņu burtus šādi a=>b=>c=>…=>y=>z=>a.
16. Saskaitīt vārdus dotajā teikumā, ja starp katriem diviem vārdiem ir viena vai vairākas atstarpes un citu atstarpju nav.
17. Noteikt īsākā un garākā vārda garumus, ja vārdus atdala tieši viena atstarpe.
18. Noteikt īsākā un garākā vārda garumus, ja vārdus atdala vismaz viena atstarpe.
19. Noteikt, cik vāri sākas un beidzas ar vienu un to pašu burtu, ja vārdus atdala atstarpju grupa.
20. Noteikt, cik vārdi satur kaut vienu e, ja vārdus atdala atstarpju grupa.
21. Noteikt, cik vārdi satur tieši 3 e, ja vārdus atdala atstarpju grupa.
22. Izdzēst no simbolu virknes vērdus, kuros ir nepāra skaits burtu.
23. Simbolu virknē atrast garumu pašai garākajai apakšvirknei, kas sastāv tikai no a burtiem.
24. Dota simbolu virkne. Izvadīt tās fragmentu starp pirmo un pēdējo punktu.
25. No matemātiskas izteiksmes izdzēsti citi simboli, palikušas tikai apaļās iekavas. Noskaidrot, vai iekavas bijušas saliktas korekti. Piemēram, iekavu izteiksme (()()) ir korekta, bet)(nav korekta.
26. Iepriekšējais uzdevums papildināts ar kvadrātiekavām un figūriekavām.
27. Tiek ievadīts lauciņš, uz kura atrodas dāma. Programmai jāizvada šaha galdiņš 8×8 simbolu veidā. Lauciņos, uz kuriem dāma var nokļūt vienā gājienā, jāraksta 1, citos lauciņos – 0.
28. Rindā pēc kārtas uzrakstīti visi naturālie skaitļi 123456789101112… . Noskaidrot, kurš cipars šajā virknē atrodas n-tajā pozīcijā.
29. Taisnstūra garums un platums ir veseli skaitļi. No taisnstūra nogriež lielāko iespējamo kvadrātu, no atlikušās daļas atkal nogriež lielāko iespējamo kvadrātu utt. Cik pavisam kvadrātu iegūs?



30. Dots taisnstūrveida biljarda galds ar malām a un b, kur a,b – naturāli skaitļi. No stūra izlido bumbiņa 45^0 leņķī pret malu, atsitas pret bortu, nākošo bortu utt., līdz nonāk kādā no stūra kulēm. Cik reizes bumbiņa atsitīsies pret bortiem?



31. Uzrakstīt programmu, kas izvada 2 doto naturālo skaitļu reizināšanas pierakstu stabiņā.
    Piemērs:

```
 123
 201
 ---
 123
246
-----
24723
```

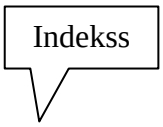
32. Saskaitīt divus dotos skaitļus, katram no kuriem var būt līdz 100 cipariem.

Masīvi

Uzdevums
Noskaidrot, vai ievadītais teksts ir simetrisks, t.i., vienādi lasāms no abiem galiem.

```
program Simetrija;
var sakums,beigas:integer;
    teksts:string;
    simetrisks:boolean;
begin
  writeln('Ievadiet tekstu');
  readln(teksts);
  sakums:=1;beigas:=length(teksts);
  simetrisks:=true;
  while (sakums<beigas) and simetrisks do begin
    simetrisks:=teksts[sakums]=teksts[beigas];
    inc(sakums);dec(beigas);
  end;
  if simetrisks then writeln('Simetrisks') else writeln('Nav
simetrisks');
end.
```

Šeit teksts ir masīvs un burti ir masīva elementi.



Masīvu apraksts:

VAR a:ARRAY[-7..13] of integer;

TYPE mas1=ARRAY[A..Z] of real;
mas2=ARRAY[CHER] of real;

VAR x:mas1;
y:mas1;

Elementi: a[-7], a[-6],…,a[13]
x[‘A’],x[‘B’],…x[‘Z’]

Darbības: a[1]:=5;
a[1]:=a[7]+3*a[5];
x:=y;

Masīva ievade un izvade

```
var a:array[1..100] of integer;
    i,n,S:integer;
begin
  {IEVADE}
  write('Ievadiet skaitļu skaitu==>');
  readln(n);
  writeln('Ievdiet skaitļus');
  for i:=1 to n do readln(a[i]);
  {IZVADE}
  for i:=1 to n do write(a[i]:4);
  writeln;
  {APRĒĶINA VIDĒJO ARITMĒTISKO}
  s:=0;
  for i:=1 to n do s:=s+a[i];
  writeln('Vid. aritm. ir ',s/n:5:2);
end.
```

Uzdevumi

1. Atrast pāra un nepāra skaitļu summas.
2. Izvadīt ievadīto masīvu rindā un apakšā vēlreiz rindā tikai no otra gala.
3. Izvadīt ievadīto masīvu un apakšā vēlreiz tikai pretējām zīmēm.
4. Izvadīt ievadīto masīvu un apakšā visu masīva elementu pēdējos ciparus.
5. Izveidot un izvadīt masīvu, kura elementi ir veseli gadījuma skaitļi no 1 līdz 10.
6. Saskaitīt, cik no masīva elementiem ir piecnieki.
7. Saskaitīt, cik piecnieku satur masīva elementi.
8. Apmainīt vietām izvēlētos divus masīva elementus.
9. Saskaitīt, cik vietās nākošais elements ir lielāks par iepriekšējo.
10. Noskaidrot vai masīva elementi veido aritmētisko progresiju.
11. Atrast lielāko masīva elementu.
12. Atrast lielāko negatīvo masīva elementu.
13. Atrast lielākā masīva elementa indeksu.
14. Atrast lielāko elementu skaitu.
15. Ievietot augošā masīvā doto skaitli tā, lai masīvs joprojām būtu augošs.
16. Doti 2 augoši masīvi. Apvienot tos vienā masīvā tā, lai tas arī būtu augošs.
17. Doti divi masīvi a un b. Cik reizes masīvs a satur masīvu b?
18. Doti vairāki skaitļi. Vai starp tiem eksistē divi, kuru summa ir 100?
19. Rindā uzrakstīti n veseli skaitļi. Virs katriem diviem blakus skaitļiem uzraksta to summu. Virs katriem diviem jaunajiem blakus skaitļiem atkal uzraksta to summu. Tā turpina līdz iegūst vienu skaitli (skatīt piemēru). Sastādīt programmu, kas dotajiem n skaitļiem atrod skaitli trijstūra virsotnē.

```
      19
    7  12
  3  4  8
2  1  3  5
```

Darbs ar failiem

ASSIGN(<mainīgais>,<faila nosaukums>); - sasaista mainīgo ar failu
RESET(<mainīgais>); - atver esošu datu failu lasīšanai
REWRITE(<mainīgais>); - izveido jaunu failu un atver to rakstīšanai
APPEND(<mainīgais>); - atver failu un ļauj to papildināt ar jaunu informāciju
CLOSE(<mainīgais>); - aizver atvērtu failu;
ERASE(<mainīgais>); - nodzēš failu no diska;
RENAME(<mainīgais1>,<mainīgais2>); - maina faila nosaukumu no pirmā uz otro
Funkciju EOF(<mainīgais>) lieto lai noteiktu, vai ir sasniegtas faila beigas

$$EOF = \begin{cases} \text{true, ja sasniegtas faila beigas} \\ \text{false, ja nav vēl faila beigas} \end{cases}$$

Funkciju EOLN(<mainīgais>) lieto, notiktu, vai sasniegtas rindiņas beigas

$$EOLN = \begin{cases} \text{true, ja sasniegtas rindiņas beigas} \\ \text{false, ja nav vēl rindiņas beigas} \end{cases}$$

Ja faila nosaukums ir CON vai tukšais vārds, notiks ievade no tastatūras vai izvade uz ekrāna. Ja faila nosaukums ir LPT1, notiks izvade uz printeri.

Piemēri

```
{Programma nolasa no faila laukums.dat kvadrāta malas garumu
un ieraksta failā kvadrāts.rez kvadrāta laukumu}
program Laukums;
var a:integer;
    f:text;
begin
  assign(f,'laukums.dat');
  reset(f);
  readln(f,a);
  close(f);
  assign(f,'laukums.rez');
  rewrite(f);
  writeln(f,sqr(a));
  close(f);
end.

{Faila vaardi.txt pirmajā rindā ir vārdu skaits, katrā
nākošajā rindā – viens vārds. Programma atrod un parāda uz
ekrāna garākā vārda garumu}
program MaxGarums;
var max,skaits,i:integer;
    vards:string;
    f:text;
begin
  assign(f,'vaardi.txt');
  reset(f);
  readln(f,skaits);
  max:=0;
  for i:=1 to skaits do begin
    readln(f,vards);
    if length(vards)>max then max:=length(vards);
  end;
  close(f);
  writeln('Garākā vārda garums ir ',max);
end.

{Programma nolasa un parāda uz ekrāna lietotāja izvēlētā faila
saturu (ja rindiņā nav vairāk par 255 simboliem)}
program Fails;
var nosaukums,a:string;
    f:text;
begin
  writeln('Ievadiet faila nosaukumu');
  readln(nosaukums);
  assign(f,nosaukums);
  reset(f);
  while not eof(f) do begin
    readln(f,a);
    writeln(a);
  end;
end.
```

Uzdevumi

1. Sastādīt programmu, kas lietotāju uzaicina ievadīt skaitļu skaitu n un intervāla galapunktus, un izveido failu gad.txt ar n veseliem gadījuma skaitļiem no izvēlētā intervāla.
2. Failā gad.txt viens zem otra sarakstīti veseli skaitļi. Sastādīt programmu, kas noskaidro, vai starp šiem skaitļiem ir 13.
3. Sastādīt programmu, kas kodē vai atkodē doto teksta failu. Kodējot visi simboli, kuru kodi lielāki par 32, jāaizstāj ar nākošo simbolu kodu tabulā. Simbols ar kodu 255 jāaizstāj ar simbolu, kura kods ir 32. Kodētais teksts jāsaglabā citā failā.
4. Pārkopēt tekstu no dotā faila uz citu, lielos latīņu burtus aizstājot ar mazajiem.
5. Pārkopēt no dotā teksta faila uz citu katru otro simbolu.

Vairākdimensiju masīvi

```
{Ievade, izvade, summa}
var a:array[1..10,1..10] of integer;
    i,j,r,k,S:integer;
begin
  write('Ievadiet rindiņu skaitu==>');
  readln(r);
  write('Ievadiet kolonu skaitu==>');
  readln(k);
  for i:=1 to r do begin
    writeln('Ievadiet ',i,'. rindiņu');
    for j:=1 to k do read(a[i,j]);
  end;
  for i:=1 to r do begin
    for j:=1 to k do write(a[i,j]:4);
    writeln;
  end;
  S:=0;
  for i:=1 to r do for j:=1 to k do inc(S,a[i,j]);
  writeln('Summa ir ',S)
end.
```

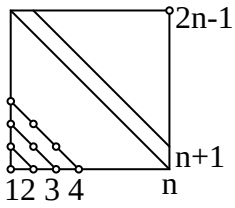
Ievade

Izvade

Summēšana

Uzdevumi

1. Palielināt visus masīva elementus par 1.
2. Palielināt izvēlētās rindiņas elementus par 1.
3. Atrast izvēlētās kolonas elementu summu.
4. Kvadrātveida tabulā galvenās diagonāles elementus nomainīt pret 0.
5. Kvadrātveida tabulā blakusdiagonāles elementus nomainīt pret 0.
6. Kvadrātveida tabulā elementus virs galvenās diagonāles nomainīt pret 0.
7. Pagriezt kvadrātveida masīvu par 90^0 pulksteņa rādītāja kustības virzienā.
8. Katrai rindiņai beigās un katrai kolonai apakšā izvadīt tās elementu summu.
9. Atrast mazāko elementu un tā atrašanās vietu. Ja tādi ir vairāki, tad visiem atrašanās vietu.
10. Masīvā glabājas burti. Katru rindiņu sakārtot alfabēta kārtībā.
11. Pārkārtot rindiņas lai pirmā kolona būtu nedilstošā kārtībā.
12. Noskaidrot, cik vietās blakus elementi ir vienādi. (Blakus elementi ir tādi, kuriem viens vai otrs indekss atšķiras tieši par 1)
13. Aprēķināt kvadrātveida masīva k-tās diagonāles summu, ja diagonāles sanumurētas kā parādīts zīmējumā.



14. Izvadīt tos skaitļus, kas sastopami visās divdimensiju masīva rindiņās. Ja tādu nav, izdot attiecīgu paziņojumu. Katru kopīgo skaitli izvadīt tikai vienu reizi.
15. Figūra „ēzelītis” lēkā pa šaha galdiņu. Dots tās maršruts. Saskaitīt, uz cik lauciņiem pabijis ēzelītis. Piemēram, ja maršruts ir a1 c7 d3 c7 d3, tad ēzelītis bijis uz 3 lauciņiem.

16. Datnes *pilsetas.txt* pirmajā rindā ierakstīts pilsētu skaits, katrā nākošajā rindiņā – vienas pilsētas nosaukums. Datnes *attalumi.txt* 1. rindiņā doti attālumi no 1. pilsētas līdz visām pilsētām; 2. rindiņā – attālumi no 2. pilsētas līdz visām pilsētām utt. (Katri divi vienas rindiņas skaitļi atdalīti ar vienu atstarpi.)
Sastādīt programmu, kas izvēlētajai pilsētai atrod tuvāko kaimiņpilsētu.

Piemērs.

Pilsetas.txt

4
Cēsis
Limbaži
Rīga
Sigulda

Attalumi.txt

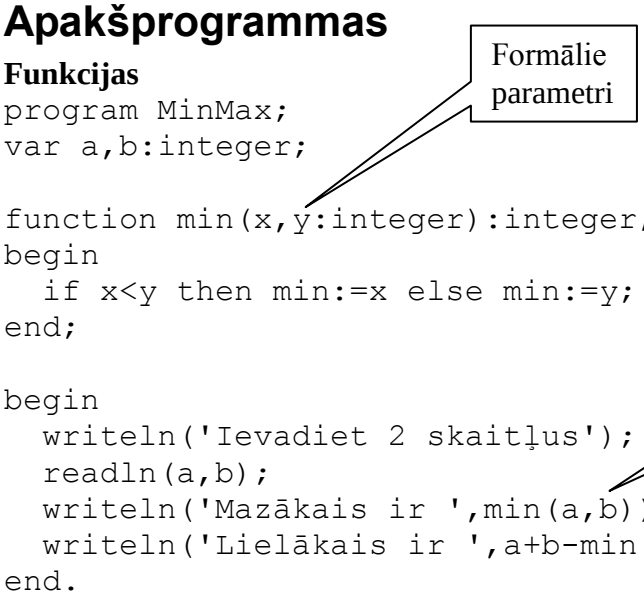
0 39 80 31
39 0 71 40
80 71 0 50
31 40 50 0

Attālums no Cēsīm līdz Rīgai

Ja ievadīta *Sigulda* atbilde ir *Cēsis*.

Apakšprogrammas

Funkcijas



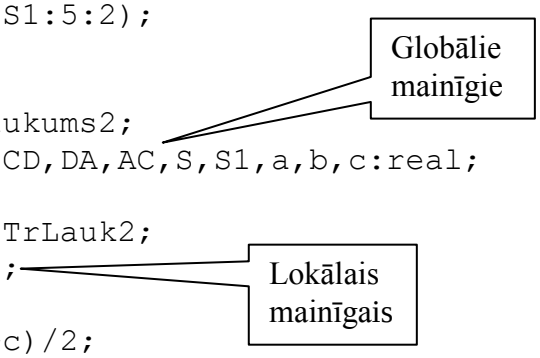

```
program MinMax;
var a,b:integer;

function min(x,y:integer):integer;
begin
  if x<y then min:=x else min:=y;
end;

begin
  writeln('Ievadiet 2 skaitļus');
  readln(a,b);
  writeln('Mazākais ir ',min(a,b));
  writeln('Lielākais ir ',a+b-min(a,b));
end.
```

Uzdevumi

1. Sastādīt programmu, kas , izmantojot funkciju min, atrod mazāko no 3 ievadītajiem skaitļiem (min atrod mazāko no diviem skaitļiem).
2. Programmēt funkciju LKD, kas atrod divu naturālu skaitļu lielāko kopīgo dalītāju.
3. Izmantojot funkciju LKD atrast 3 skaitļu lielāko kopīgo dalītāju.
4. Izmantojot funkciju LKD atrast 2 skaitļu mazāko kopīgo dalāmo.
5. Programmēt funkciju left(a,n), kuras rezultāts ir teksta a pirmie n simboli.
6. Programmēt funkciju right(a,n), kuras rezultāts ir teksta a pēdējie n simboli.
7. Sastādīt programmu, kas aprēķina izteiksmi $\frac{1+\sqrt{a}}{1+\sqrt{7}}+1+\sqrt{a+1}$, izmantojot funkciju $f(x)=1+\sqrt{x}$.
8. Programmēt funkciju KOS(a,b,c), kas aprēķina trijstūra leņķa A kosinusu, ja dotas visas trijstūra malas.
9. Sastādīt programmu, kas, izmantojot funkciju KOS, aprēķina trijstūra visu leņķu kosinusus.
10. Programmēt funkciju, kas atrod skaitļa ciparu summu.
11. Sastādīt programmu, kas aprēķina trijstūra perimetru, ja dotas trijstūra virsotņu koordinātas. Definēt funkciju, kas rēķina nogriežņa garumu.
12. Aprēķināt četrstūra laukumu, ja dotas virsotņu koordinātas. Definēt funkcijas nogriežņa garumam un trijstūra laukumam.
13. Programmēt funkciju UZ10, kas pārveido doto skaitli no dotās skaitīšanas sistēmas uz decimālo skaitīšanas sistēmu.
14. Programmēt funkciju NO10, kas pārveido doto skaitli no decimālās skaitīšanas sistēmas uz doto skaitīšanas sistēmu.
15. Sastādīt programmu, kas, izmantojot funkcijas NO10 un UZ10, pārveido doto skaitli no dotās skaitīšanas sistēmas uz citu doto skaitīšanas sistēmu.

Procedūras
Doti četrstūra malu un vienas diagonāles garumi. Sastādīt programmu, kas aprēķina četrstūra laukumu.

```
program Laukums1;
var AB,BC,CD,DA,AC,S,S1,a,b,c,p:real;

procedure TrLauk1;
begin
  p:=(a+b+c)/2;
  S:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
  readln(AB,BC,CD,DA,AC);
  a:=AB;b:=BC;c:=AC;
  TrLauk1;
  S1:=S;
  a:=CD;b:=DA;
  TrLauk1;
  S1:=S1+S;
  writeln(S1:5:2);
end.

program Laukums2;
var AB,BC,CD,DA,AC,S,S1,a,b,c:real;

procedure TrLauk2;
var p:real;
begin
  p:=(a+b+c)/2;
  S:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
  readln(AB,BC,CD,DA,AC);
  a:=AB;b:=BC;c:=AC;
  TrLauk2;
  S1:=S;
  a:=CD;b:=DA;
  TrLauk2;
  S1:=S1+S;
  writeln(S1:5:2);
end.
```



```
program Laukums3;
var AB,BC,CD,DA,AC,S1,S2:real;

procedure TrLauk3(var a,b,c,S:real);
var p:real;
begin
  p:=(a+b+c)/2;
  S:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
  readln(AB,BC,CD,DA,AC);
  TrLauk3(AB,BC,AC,S1);
  TrLauk3(CD,DA,AC,S2);
  writeln(S1+S2:5:2);
end.

program Laukums4;
var AB,BC,CD,DA,AC,S1,S2:real;

procedure TrLauk4(a,b,c:real; var S:real);
var p:real;
begin
  p:=(a+b+c)/2;
  S:=sqrt(p*(p-a)*(p-b)*(p-c));
end;

begin
  readln(AB,BC,CD,DA,AC);
  TrLauk4(AB,BC,AC,S1);
  TrLauk4(CD,DA,AC,S2);
  writeln(S1+S2:5:2);
end.
```

Apakšprogrammu parametri var būt arī masīvi. Tikai tad gan formālais parametrs, gan reālā vērtība jāapraksta vienā vietā (Ja lietotāja definēti mainīgie tiek aprakstīti 2 reizes pilnīgi vienādi, paskāls tomēr tos neuzskata par vienādiem). Piemērs:

```
type mas=array[1..10] of byte;
var a:mas;
procedure Proc(b:mas;c:integer);
...
Proc(a,3);
...
```

Uzdevumi

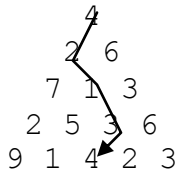
1. Zināms, ka riņķa līnijas centrs ir (0,0). Dots tās rādiuss un patvaļīga punkta koordinātas. Uzrakstīt procedūru, kas noskaidro koordinātas punktam, kurā stars, kas vilkts no riņķa līnijas centra caur doto punktu, krusto riņķa līniju.
2. Uzrakstīt procedūru, kas saskaita divas parastās daļas. Ievadīti tiek 4 skaitļi: abu daļu skaitītāji un saucēji. Izvadīti tiek 2 skaitļi: rezultāta skaitītājs un saucējs. Daļai jābūt saīsinātai.

Rekursija
Algoritmu sauksim par rekursīvu, ja tas izsauc sevi kā apakš algoritmu.

```
program Faktorials;
var n:longint;

function fakt(n:longint):longint;
begin
  if n=1 then fakt:=1 else fakt:=n*fakt(n-1);
end;

begin
  readln(n);
  writeln(fakt(n));
end.
```

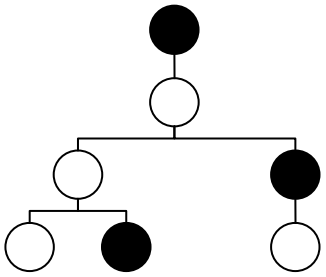
Uzdevumi

1. Fibonači virkne tiek definēta šādi: tās pirmie divi locekļi ir vieninieki, bet katru nākošo aprēķina saskaitot divus iepriekšējos. Sastādīt programmu, kas aprēķina n-to Fibonači virknes locekli.
2. Sastādīt programmu, kas aprēķina pakāpi ar veselu kāpinātāju pēc šādas rekursīvas definīcijas: $a^n = \begin{cases} a^{n-1} \cdot a, ja n > 0 \\ a^{n+1} : a, ja n < 0 \\ 1, ja n = 0 \end{cases}$
3. Sastādīt programmu, kas atrod dotā naturālā skaitļa ciparu summu; pēc tam – iegūtā skaitļa ciparu summu utt. Līdz palicis viens cipars.
4. Trijstūrī sarakstīti naturāli skaitļi: 1. rindā – viens skaitlis; 2. rindā – 2 skaitļi; ...; n-tajā rindā - n skaitļi. No jebkura skaitļa drīkst pārvietoties uz jebkuru no diviem tam tieši apakšā esošajiem skaitļiem. Sastādīt programmu, kas atrod lielāko summu, ko var savākt pārvietojoties no virsotnes līdz pamatam. Piemēram, zīmējumā redzamā maršruta summa ir 14.



5. Iepriekšējā uzdevumā parādīt arī ceļu.
6. Tabulā sarakstīti n×m naturāli skaitļi. No kādas rūtiņas drīkst pāriet vai nu pa labi vai uz leju. Sastādīt programmu, kas atrod lielāko summu, ko var savākt pārvietojoties no kreisā augšējā stūra līdz labajam apakšējam stūrim.
7. Iepriekšējā uzdevumā parādīt arī ceļu.
8. Tabulā sarakstīti n×m naturāli skaitļi. No kādas rūtiņas drīkst pāriet vai nu pa labi vai uz eju, ja rūtiņā, uz kuru gribam iet ir lielāks skaitlis nekā tajā, kur atrodamies. Atrast ceļu no kreisā augšējā uz labo apakšējo stūri.
9. Sastādīt programmu, kas ģenerē visas iespējamās pirmo n alfabēta burtu virknītes.
10. Hanojas torņi. Doti 3 stieņi A, B un C. Uz stieņa A uzvērtas n dažāda diametra ripas, pie tam diametra dilšanas kārtībā, skaitot no apakšas. Ar vienu gājienu atļauts no jebkura stieņa pārlikt augšējo ripu uz jebkuru citu stieni, ja šīs ripas diametrs ir mazāks par tās ripas diametru, kurai liekam virsū. Uz tukša stieņa drīkst likt jebkura diametra ripu. Sastādīt programmu, kas izdod gājienu secību, kā visas ripas no stieņa A pārvietot uz stieni B.

11. Par augošas naturālu skaitļu virknes kodu sauksim skaitli, ko iegūst pēc kārtas uzrakstot visus virknes locekļus. Piemēram, virknes 1, 7, 23, 98 kods ir 172398. Tādu pašu kodu iegūst kodējot arī virknes 172,398; 17,23,98; 172398. Ievadītajam virknes kodam noteikt, cik dažādas virknes kodējot var iegūt šo kodu.
12. Pirmajā līmenī ir viena melna bumbiņa. Katra melnā bumbiņa rada vienu baltu bumbiņu. Katra balta bumbiņa rada vienu baltu un vienu melnu bumbiņu. Pirmie četri līmeņi redzami zīmējumā. Ievadītajam līmeņa numuram noteikt, cik melno un cik balto bumbiņu ir šajā līmenī. Piemēram, 4. līmenī ir 1 melna un 2 baltas bumbiņas.



13. Sastādīt programmu, kas atrod dotās skaitļu virknes garākās nedilstošās apakšvirknes garumu. Piemēram, virknē 1;7;2;3;2;5 viena no nedilstošajām apakšvirknēm ir pasvītrota. Tas garums ir 4.

Grafiskās procedūras un funkcijas

```
Program Demo;
USES Graph;                          {deklarē bibliotēku}
Var x,y:integer;
begin
  x:=detect;                         {draiveris pats noskaidros, kāda grafiskā karte ir
                                      datorā}
  initgraph(x,y,'C:\BP\BGI');        {instalē grafisko draiveri}

  <programma>

  readln;
  CloseGraph;                        {aizver visas grafiskās operācijas}
end.
```

Punkts

PutPixel(x,y,<krāsa>) iekrāso doto punktu
MoveTo(x,y) pārvieto aktīvo punktu uz (x,y)
MovieRel(dx,dy) pārvieto aktīvo punktu par vektoru (dx,dy)

Nogriežņi

Line(x1,y1,x2,y2) zīmē nogriezni no (x1,y1) līdz (x2,y2)
LineTo(x,y) zīmē nogriezni no aktīvā punkta līdz (x2,y2)
LineRel(dx,dy) zīmē vektoru (dx,dy) no aktīvā punkta
SetLineStyle(<stils>,<ornaments>,<biezums>) uzstāda līnijas veidu
stils=0..4
ornaments tiek ņemts tikai, ja stils=4
biezums=1 vai 3

Taisnstūri

Rectangle(x1,y1,x2,y2) zīmē taisnstūri, kura pretējās virsotnes ir (x1,y1) un (x2,y2)
Bar(x1,y1,x2,y2) zīmē aizkrāsotu taisnstūri
Bar3D(x1,y1,x2,y2,<biezums>,<boolean parametrs>) zīmē paralēlskaldni ar augšējo skaldni, ja boolean parametrs=TRUE, vai bez, ja boolean parametrs=FALSE un aizkrāso priekšējo skaldni

Loki

Circle(x,y,R) zīmē riņķa līniju ar centru (x,y) un rādiusu R
Arc(x,y,l1,l2,R) zīmē riņķa līnijas loku ar doto centru un rādiusu
l1 - loka sākums (grādos)
l2 - loka beigas
Ellipse(x,y,l1,l2,xR,yR) zīmē elipses loku
xR - rādiuss pa x asi
yR - rādiuss pa y asi

FillEllipse(x,y,xR,yR) zīmē aizkrāsotu elipsi
PieSlice(x,y,l1,l2,R) zīmē aizkrāsotu riņķa sektoru
Sector(x,y,l1,l2,xR,yR) zīmē aizkrāsotu elipses sektoru

Krāsojumi

SetColor(<krāsa>) uzstāda aktīvo krāsu

SetBkColor(<krāsa>) uzstāda fona krāsu

SetFillStyle(<ornaments>,<krāsa>) uzstāda aizkrāsojuma ornamentu un krāsu
ornaments=0..12:
0 – fona krāsa
1 – bez ornamenta
12 – lietotāja šablons

SetFillPattern(<ornaments>,<krāsa>) uzstāda lietotāja definētu aizkrāsojuma ornamentu un krāsu

FloodFill(x,y,<robežas krāsa>) aizkrāso slēgtu apgabalu ap doto punktu

Teksts

OutText(<teksts>) izvada tekstu uz ekrāna

OutTextXY(x,y,<teksts>) izvada tekstu uz ekrāna norādītajā vietā

SetTextJustify(h,v) uzdod teksta izlīdzināšanas režīmu
h: 0 – aktīvajā punktā kreisā mala
1 - aktīvajā punktā centrs
2 - aktīvajā punktā labā mala
v: 0 - aktīvajā punktā apakšējā mala
2 – aktīvajā punktā augšējā mala

SetTextStyle(<fonts>,<virziens>,<izmērs>) uzstāda teksta stilu
fonts=0..10
virziens=0 vai 1

Ekrāns un logs

ClearDevice dzēst ekrānu

GetMaxX lielākā x koordināta

GetMaxY lielākā y koordināta

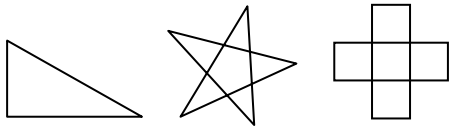
SetViewPort(x1,y1,x2,y2,<boolean parametrs>) definē logu, ja boolean parametrs=TRUE, tad zīmējums aiz loga būs nogriezts, ja parametrs=FALSE, tad nē.

ClearViewPort notīra uzstādīto logu

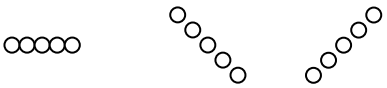
```
Program VeidotsRaksts;
uses graph,crt;
var  c,d,i:integer;
     f:FillPatternType;
begin
  f[1]:=15;f[2]:=200;f[3]:=123;f[4]:=64;
  f[5]:=80;f[6]:=95;f[7]:=132;f[8]:=7;
  DetectGraph(c,d);
  InitGraph(c,d,'c:\BP\Bgi');
  SetColor(2);
  Circle(320,240,200);
  SetFillPattern(f,4);
  FloodFill(320,200,2);
  while not KeyPressed do;
End.
```

Uzdevumi

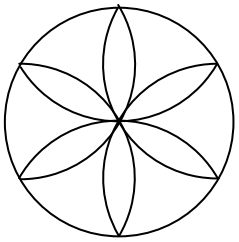
1. Sastādīt programmu, kas uzzīmē
   - taisnleņķa trijstūri;
   - zvaigzni;
   - krustu.



2. Sastādīt programmu, kas zīmē riņķīšu virkni
   - horizontāli;
   - pa galveno diagonāli;
   - pa blakus diagonāli.



3. Sastādīt programmu, kas zīmē krustiņu virkni. × × × × × ×
4. Sastādīt programmu, kas zīmē daudzas horizontālas krustiņu virknes (pilnu ekrānu).
5. Sastādīt programmu, kas ekrānu izkrāso kā šaha galdiņu. Rūtiņas mala 5 punkti.
6. Sastādīt programmu, kas saliek ekrānā 1000 gadījuma punktus gadījuma krāsās.
7. Sastādīt programmu, kas zīmē lauztu līniju ar gadījuma virsotnēm.
8. Sastādīt programmu, kas uzzīmē 100 riņķus ar gadījuma centriem un gadījuma rādiusu un tos aizkrāso gadījuma krāsās.
9. Sastādīt programmu, kas uzzīmē 100 gadījuma taisnstūrus.
10. Sastādīt programmu, kas uzzīmē zīmējumā redzamo ziediņu un izkrāso tā ziedlapiņas.



Rekursīvi zīmējumi

Algoritmu sauksim par rekursīvu, ja tas izsauc sevi kā apakšalgoritmu.

Aplūkosim rekursīvu grafisku algoritmu piemērus

1. piemērs.

Uzzīmēt koncentriskas riņķa līnijas.

Risinājums.

Liekam izpildīt procedūru **aplis**, ja r = 200.

Programma **aplis**.

Ja r > 0, tad

1. Zīmēt riņķa līniju, kuras centrs ir ekrāna centrā un rādiuss ir r.
2. Izpildīt programmu **aplis**, ja r =r–10.

PASCAL programma.

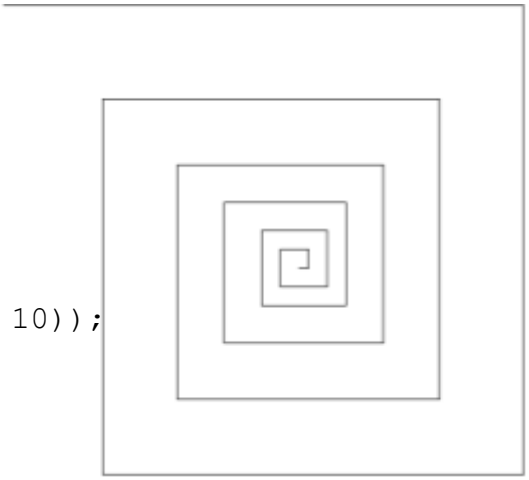
```
Program Apli;
uses graph;
var a,b:integer;

procedure aplis(r:integer);
begin
     if r>1 then begin
        circle(GetMaxX div 2,GetMaxY div 2,r);
        aplis(r-10);
     end;
end;

begin
  detectgraph(a,b);
  initgraph(a,b,'c:\BP\bgi');
  aplis(200);
  readln;
end.
```

2. piemērs

Zīmēt horizontālu nogriezni, tam galā vēl vienu, bet divreiz īsāku un citā krāsā, tam galā vēl vienu – vēl divreiz īsāku un citā krāsā utt.

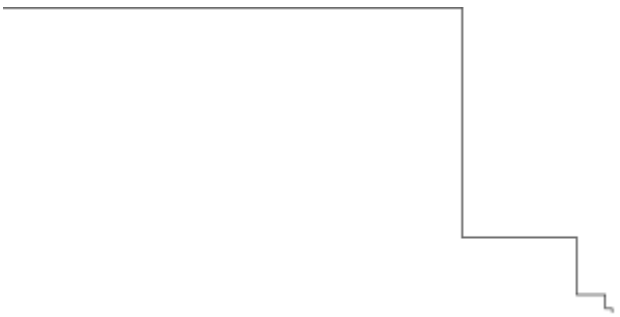
```
Program Taisne;
uses graph;
var a,b:integer;
procedure
nogrieznis(d,k:integer);
begin
  if d>1 then begin
    SetColor(k);
    LineRel(d,0);
    nogrieznis(d div 2,k+1);
  end;
end;

begin
  detectgraph(a,b);
  nitgraph(a,b,'c:\BP\bgi');
  MoveTo(0,100);
  nogrieznis(256,1);
  readln;
end.
```

3.piemērs. Uzzīmēt spirāli (katrs nākošais posms ir 9/10 no iepriekšējā).

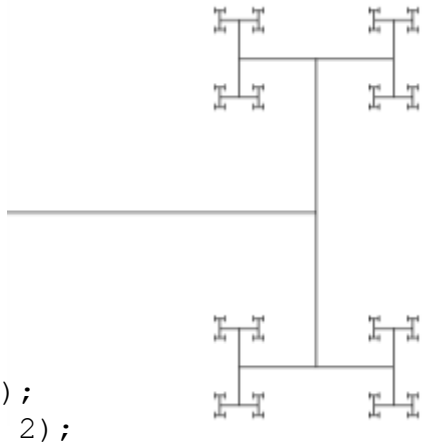


```
Program Spirale;
uses graph;
var a,b:integer;

procedure nogrieznis(x,y:integer);
begin
  if abs(x+y)>1 then begin
    LineRel(x,y);
    nogrieznis(9*(-y div 10),9*(x div 10));
  end;
end;

begin
  detectgraph(a,b);
  initgraph(a,b,'c:\BP\bgi');
  MoveTo(50,1);
  nogrieznis(500,0);
  readln;
end.
```

1.uzdevums. Uzzīmēt attēlā redzamo (katrs nākošais posms 2 reizes īsāks par iepriekšējo).



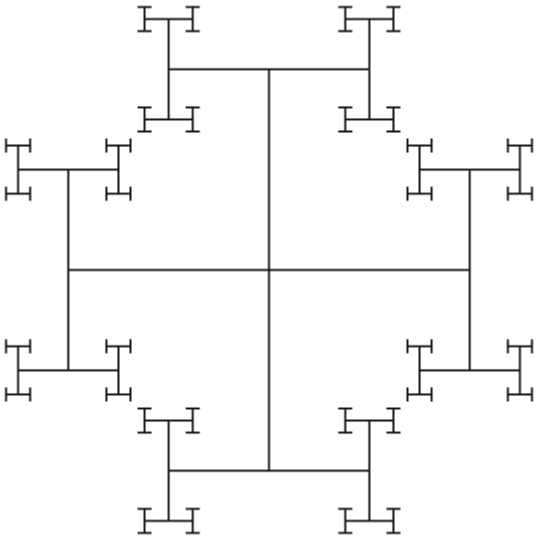
4.piemērs. Uzzīmēt horizontālu nogriezni, tam galā perpendikulāri uz abām pusēm – pa divreiz īsākam nogrieznim, katram no tiem galā perpendikulāri uz abām pusēm – pa divreiz īsākam nogrieznim utt.
Atšķirībā no iepriekšējā piemēra, šajā bildītē nākošais nogrieznis vienmēr netiks zīmēts iepriekšējam galā. Tāpēc kā parametrus nodosim x,y – nogriežņa sākuma koordinātas, dx,dy – beigu relatīvās koordinātas (salīdzinoši ar sākumu)



```
Program Fraktals;
uses graph;
var a,b:integer;

procedure nogrieznis(x,y,dx,dy:integer);
begin
  if abs(dx+dy)>1 then begin
    MoveTo(x,y);
    LineRel(dx,dy);
    nogrieznis(x+dx,y+dy,dy div 2,dx div 2);
    nogrieznis(x+dx,y+dy,-dy div 2,-dx div 2);
  end;
end;

begin
  detectgraph(a,b);
  initgraph(a,b,'c:\BP\bgi');
  nogrieznis(0,200,256,0);
  readln;
end.
```

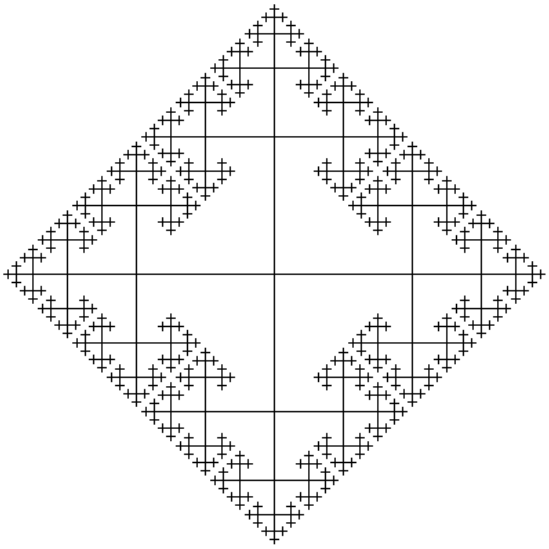
2. uzdevums. Uzzīmēt divus vienādus horizontālus nogriežņus katru uz savu pusi no ekrāna centra, katram no tiem galā perpendikulāri uz abām pusēm – pa divreiz īsākam nogrieznim, katram no tiem galā perpendikulāri uz abām pusēm – pa divreiz īsākam nogrieznim utt.

3. uzdevums. Līdzīgi kā iepriekšējā uzdevumā, tikai no ekrāna centra iziet 4 nogriežņi.



4. uzdevums. Kā iepriekš. Tikai katram nogrieznim galā jāzīmē trīs nogriežņi.

5. uzdevums. Kā iepriekš. Tikai nogriežņi nav perpendikulāri, bet veido 120^0 leņķi un no katra krustojuma iziet 3 nogriežņi.



Kopas

TYPE <tipa vārds>=SET OF <bāzes tips>;
VAR <kopas vārds>:<tās tips>;
CONST <kopas vārds>[:<tās tips>]=[<elementu uzskaitījums>]; - elementu uzskaitījums jāliek kvadrātiekavās.

Piemēri

```
type burti=set of 'A'..'Z';
     diena=set of (pir,otr,tre,cet,pie,ses,sve);
var a:set of 0..9;
    b:burti;
const c=[1..6,9];
```

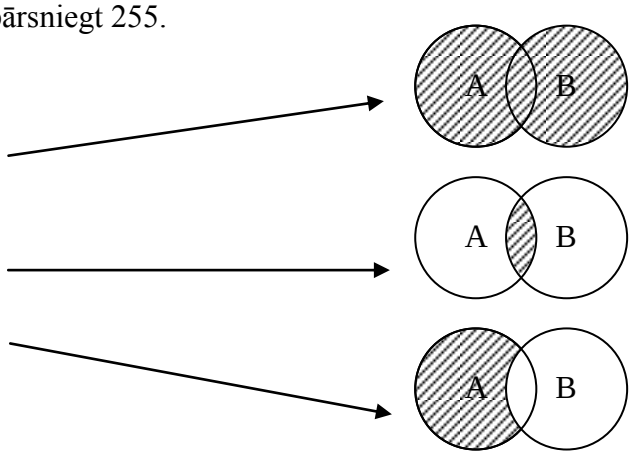
Kopas elementu skaits nedrīkst pārsniegt 255.

Darbības ar kopām

Kopu apvienojums R:=A+B

Kopu šķēlums R:=A*B

Kopu starpība R:=A-B



Kopu salīdzināšana

A=B

A<>B

A<=B (A ir B apakškopa)

A>=B (B ir A apakškopa)

A<B

A>B

Elementa piederība kopai

<elements> IN <kopa>

Piemēri
Noskaidrot, vai ievadītais simbols ir latīņu alfabēta burts.

```
var a:char;
begin
  readln(a);
  if a in ['A'..'Z','a'..'z'] then writeln('Ievadītais simbols ir burts')
                              else writeln('Ievadītais simbols nav
                              burts')
end.
```

Atrast dotajā frāzē unikālus simbolus, t.i., tādus, kuri šajā tekstā sastopami 1 reizi. Frāzes beigu pazīme ir punkts.

```
uses crt;
var k1,k2: set of char;
    c:char;
begin
  k1:=[];
  k2:=[];
  repeat
    c:=readkey;
    if c in k1 then k2:=k2+[c] else k1:=k1+[c];
  until c='.';
  k1:=k1-k2;
  writeln('Unikālo simbolu kopa ir ');
  for c:=' ' to chr(126) do if c in k1 then write(c);
end.
```

Te būs visi „sastaptie" simboli

Te būs simboli, kuri jau ir kopā k1

Uzdevumi

1. Sastādīt programmu, kas tekstā saskaita patskaņus (izmantot patskaņu kopu).
2. Atrast visus pirmskaitļus, kas nepārsniedz ievadīto skaitli n (n≤255), ar Eratostena sieta palīdzību: sākumā ir visi naturālie skaitļi no 2 līdz n; tad divnieku atstāj, bet tālāk katru otro (pāra skaitļus) izsvītro; tad nākošo no neizsvītrotajiem (3) atstāj, bet tālāk katru trešo izsvītro (ieskaitot jau izsvītrotos); tad nākošo no neizsvītrotajiem (5) atstāj, bet tālāk katru piekto izsvītro (ieskaitot jau izsvītrotos); utt. Beigās palikušie ir pirmskaitļi.
3. Spēles dalībnieki nostājušies aplī un skaita pantiņu. Kam tiek pēdējā zilbe, tas izstājas. Skaitīšanu atsāk no nākošā. Līdz paliek tikai viens dalībnieks. Sastādīt programmu, kas noskaidro, kurš paliek.
   Precīzāk: dalībniekus sanumurēsim skaitīšanas virzienā, sākot ar to, no kura sāk skaitīšanu. Ievadīsim bērnu skaitu n (n≤255) un skaitāmpantiņa zilbju skaitu k. Jāatrod pēdējā palikušā dalībnieka numurs.
4. Sastādīt programmu, kas atrisina rēbusu:

```
  VIEGLI
  IEGŪTS
  VIEGLI
  AIZIET
```

Ieraksti

```
type gramata=record
               autors:string[20];
               nosaukums:string[50];
               gads,apjoms:integer;
               saturs:string;
             end;
var eksemplars:gramata;
    katalogs:array[1..100] of gramata;
```

Vērtību piešķiršana ieraksta tipa mainīgajam

1. vaids

```
  eksemplars.autors:='Misans I.';
  eksemplars.nosaukums:='Stelitu televizija';
  eksemplars.gads:=1992;
  eksemplars.apjoms:=209;
  eksemplars.saturs:='Rokasgramata inzenieriem';
```

2. veids

```
  with eksemplars do begin
    autors:='Misans I.';
    nosaukums:='Stelitu televizija';
    gads:=1992;
    apjoms:=209;
    saturs:='Rokasgramata inzenieriem';
  end;
```

Visu ierakstu var piešķirt citam tā paša tipa mainīgajam:

```
Katalogs[5]:=eksemplārs;
```

Piemērs grāmatas meklēšanai pēc autora, kura uzvārds tiek ievadīts no tastatūras:

```
readln(a);
for i:=1 to 100 do
  if a=katalogs[i].autors then
    writeln(a,' ir autors grāmatai: ',katalogs[i].nosaukums);
```

Ieraksts ar mainīgu daļu

```
type publikacija=(GRAMATA,ZURNALS,KRAJUMS);
     avots=record
             autors:string[20];
             nosaukums:string[50];
             case veids:publikacija of
               GRAMATA:(pilseta:string[20];
                        izdevnieciba:string[20]);
               ZURNALS:(nosauk:string[30];
                        numurs:integer);
               KRAJUMS:(nosauku:string[30];
                        izdevniec:string[20]);
           end;
```

```
with katalogs[10] do begin
  autors:='Germanis';
  nosaukums:='latviesu tautas piedzivojumi';
  veids:=GRAMATA;
  pilseta:='Riga';
  izdevnieciba:='Zvaigzne';
end;
```

Piemērs, kur vienā ierakstā iekļauti citi ieraksti

```
type gramata=record
               Autors:string[20];
               Nosaukums:string[50];
               Saturs:record
                        TemasKods:string[10];
                        Atslegvards:string[20];
                        Orientacija:string[30];
                        GalvenaIdeja:string[100];
                      end;
               Gads:integer;
             end;
```

Lietošana:

```
eksemplars.Saturs.Orientacija:='Iesacejiem';
```

vai:

```
with eksemplars.Saturs do begin
  readln(TemasKods);
  readln(Atslegvards);
  readln(Orientacija);
  readln(GalvenaIdeja);
end;
```

Uzdevumi

1. Sastādīt programmu, kas, izmantojot ierakstus, veic sekojošas darbības ar grāmatu katalogu:
    a) ļauj ievadīt datus no tastatūras un saglabā tos diskā;
    b) parāda katalogu no diska uz ekrāna;
    c) meklē katalogā grāmatu pēc nosaukuma vai autora;
    d) meklē katalogā grāmatu, kuras nosaukums satur ievadīto frāzi.
2. Definēsim tipu VEKTORS:record x,y:real end, kur x un y ir vektora koordinātas. Izmantojot to, programmēt procedūras SUM, REIZ, SKAL, GARUMS, LENKIS, kas atrod attiecīgi 2 vektoru summu, vektora reizinājumu ar skaitli, 2 vektoru skalāro reizinājumu, vektora garumu, leņķi starp 2 vektoriem.
3. Definēsim komplekso skaitļu tipu KOMPL:record Re,Im:real end. Izmantojot to, programmēt procedūras SUM, REIZ, DAL, MODULIS, kas atrod attiecīgi 2 kompleksu skaitļu summu, reizinājumu, dalījumu, kompleksa skaitļa moduli.

Dinamiskās datu struktūras

```
type NoradesTips=^integer;
var a:NoradesTips;
begin
  new(a);
  readln(a^);
  a^:=a^*a^;
  writeln(a^);
  dispose(a);
end.
```

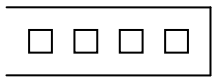
Atvēl atmiņā vietu vienam integer tipa skaitlim. Mainīgajam a piešķir šīs vietas adresi.

Atbrīvo vietu atmiņā.

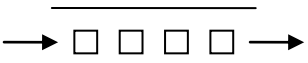
Datu struktūras:

- Rinda;
- Steks
- Saraksts (ķēde)
- Koks

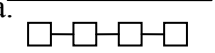
Steks – šaurs tunelis ar aizbrukušu galu. Kurš pēdējais ieiet, tas pirmais iznāk.



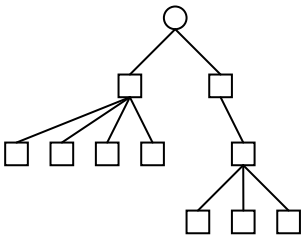
Rinda – šaurs tunelis, kam iet cauri. Kurš pirmais ieiet, tas pirmais laukā.



Saraksts – rinda, kurā var iestāties jebkurā vietā un no jebkuras vietas iziet laukā.

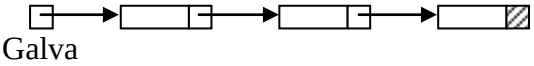


Koks



Steks

```
type NoradesTips=^DatuTips;
     DatuTips=record
                nakamais:NoradesTips;
                dati:string;
              end;
var galva:NoradesTips;
begin
  galva:=nil;
  ...

procedure pievieno(t:string);
var jaunais:NoradesTips;
begin
  new(jaunais);
  jaunais^.dati:=t;
  jaunais^.nakamais:=galva;
  galva:=jaunais;
end;
```



```
procedure iznem(var t:string);
var vecais:NoradesTips;
begin
  t:='';
  if galva<>nil then begin
    t:=galva^.dati;
    vecais:=galva;
    galva:=galva^.nakamais;
    dispose(vecais);
  end;
end;
```

Rinda

Rindu definē tāpat kā steku. Jaunu elementu pievieno tāpat kā stekam.

```
procedure iznem(var t:string);
var vecais,prp:NoradesTips;
begin
  t:='';
  if galva<>nil then begin
    vecais:=galva;
    prp:=nil;
    while vecais^.nakamais<>nil do begin
      prp:=vecais;
      vecais:=vecais^.nakamais;
    end;
    t:=vecais^.dati;
    if prp=nil then galva:=nil else prp^.nakamais:=nil;
    dispose(vecais);
  end;
end;
```

Vai arī tā:

```
procedure iznem(var t:string);
var vecais:NoradesTips;
begin
  if galva=nil then begin t:='';exit end;
  if galva^.nakamais=nil then begin
    t:=galva^.dati;
    galva:=nil;
    exit;
  end;
  vecais:=galva;
  while vecais^.nakamais^.nakamais<>nil do
vecais:=vecais^.nakamais;
  t:=vecais^.nakamais^.dati;
  dispose(vecais^.nakamais);
  vecais^.nakamais:=nil;
end;
```

Saraksts

Sarakstu definē tāpat kā steku un rindu.

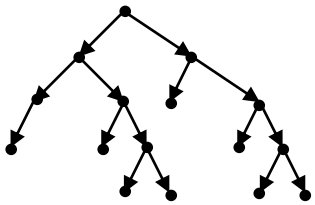
```
procedure visi;
var n:NoradesTips;
begin
  n:=galva;
  while n<>nil do begin
    writeln(n^.dati);
    n:=n^.nakamais;
  end;
end;
```

Visa saraksta parādīšana uz ekrāna

Procedūra pievieno datus t2 aiz datiem t1. Ja t1 ir vairākās vietās, tad aiz pirmā t1. Ja t1 vispār nav, tad t2 tiek pievienots aiz pēdējā saraksta elementa.

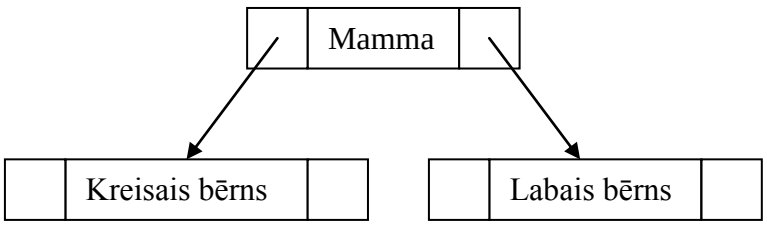
```
procedure PievienoAiz(t1,t2:string);
var n,jauns:NoradesTips;
begin
  new(jauns);
  jauns^.dati:=t2;
  n:=galva;
  while (n^.dati<>t1) and (n^.nakamais<>nil) do
n:=n^.nakamais;
  jauns^.nakamais:=n^.nakamais;
  n^.nakamais:=jauns;
end;
```

Koks



Aplūkosim bināru koku, t.i., tādu, kuram nevienam elementam nav vairāk kā 2 bērnu (labais un kreisais).

```
type NoradesTips=^DatuTips;
     DatuTips=record
                kreisais,labais:NoradesTips;
                vards:string;
              end;
var galva:NoradesTips;
    t:string;

begin
  galva:=nil;
  ...
```



Pie kreisās rokas liksim par mammu mazāku bērnu, pie labās – lielāku.

```
procedure pielikt(var n:NoradesTips;dati:string);
var jaunais:NoradesTips;
begin
  if n=nil then begin
    new(jaunais);
    jaunais^.vards:=dati;
    jaunais^.kreisais:=nil;
    jaunais^.labais:=nil;
    n:=jaunais;
    end
  else begin
    if dati<n^.vards then pielikt(n^.kreisais,dati);
    if dati>n^.vards then pielikt(n^.labais,dati);
  end;
end;

procedure apstaigaat(n:NoradesTips);
begin
  if n<>nil then begin
    apstaigaat(n^.kreisais);
    writeln(n^.vards);
    apstaigaat(n^.labais);
  end;
end;
```

Uzdevumi

1. Sastādīt programmu, kas ļauj pievienot veselu skaitļu sarakstam jaunu skaitli tā, ka saraksts visu laiku paliek sakārtots nedilstošā kārtībā. Programmai jāspēj arī parādīt visu sarakstu.
2. Sastādīt programmu, sarakstam ļauj pievienot jaunu vārdu, spēj parādīt visu sarakstu un spēj izņemt no saraksta ievadīto vārdu.
3. Matemātiskā izteiksmē lietotas apaļās iekavas, kvadrātiekavas un figūriekavas. Visi pārējie simboli izdzēsti, palikušas tikai iekavas. Sastādīt programmu, kas noskaidro, vai iekavas bija saliktas korekti. Piemēram, {[]{()}} ir korekti, bet)(nav korekti. (Izmantot steku.)

Noderīgas taustiņu kombinācijas

Komandu redaktors.

Pārvietošanās pa tekstu.

<Bultiņu taustiņi>	par vienu pozīciju
<Ctrl> ←	vārdu pa kreisi
<Ctrl> →	vārdu pa labi
<PgUp>	lpp uz leju
<PgDawn>	lpp uz augšu
<Home>	uz rindiņas sākumu
<End>	uz rindiņas beigām
<Ctrl><Home>	uz ekrāna sākumu
<Ctrl><End>	uz ekrāna beigām
<Ctrl><PgUp>	uz teksta sākumu
<Ctrl><PgDawn>	uz teksta beigām
<Ctrl> QW	uz kļūdas vietu
<Ctrl> [	uz aizverošo iekavu
<Ctrl> K4	izveidot 4. iezīmi (pieļaujamas iezīmes no 0 līdz 9)
<Ctrl> Q4	pāriet uz 4. iezīmi
<Ctrl> Z	ekrāna pārvietošana uz augšu
<Ctrl> W	ekrāna pārvietošana uz leju

Dēst

<Delete>	tekošo simbolu
<BackSpace>	simbolu pa kreisi
<Ctrl> Y	visu rindiņu
<Ctrl> QY	līdz rindiņas beigām
<Ctrl> T	vārdu

Darbības ar teksta blokiem

<Ctrl> KB	atzīmēt bloka sākumu
<Ctrl> KK	atzīmēt bloka beigas
<Ctrl> KT	iezīmēt vārdu
<Ctrl> KC	kopēt bloku
<Ctrl> KV	pārvietot bloku
<Ctrl><Delete>	dzēst bloku
<Ctrl> KI	pārvietot bloku pa labi
<Ctrl> KU	pārvietot bloku pa kreisi
<Ctrl> KR	ievietot bloku no diska
<Ctrl> KW	ierakstīt bloku diskā
<Ctrl> KP	drukāt bloku
<Ctrl> QB	pāriet uz bloka sākumu
<Ctrl> QK	pāriet uz bloka beigām
<Ctrl> KH	noņemt bloka iezīmējumu
<Ctrl><Insert>	iekopēt bloku starpliktuvē
<Ctrl><Delete>	pārvietot bloku uz starpliktuvi
<Ctrl><Insert>	paņemt bloku no starpliktuves

Meklēšana un nomaiņa

<Ctrl> QF	meklēšana
<Ctrl> QA	meklēšana un nomaiņa
<Ctrl> L	atkārtot pēdējo meklēšanu

Citas komandas

<Ctrl> OI	automātiskās atkāpes režīma pārslēgšana
<Insert>	ievietošanas režīma pārslēgšana
<Alt><BackSpace>	pēdējās darbības atcelšana

Darbs ar programmu.

<Ctrl><F1>	palīdzība par paskālu
<F10>	ieeja izvēlnē
<F3>	nolasīt failu no diska
<F2>	saglabāt failu diskā
<ALT>X	izeja no paskāla
<F9>	kompilēt
<Ctrl><F9>	izpildīt
<ALT><F5>	lietotāja ekrāns
<F7>	izpildīt soli, ieejot apakšprogrammās
<F8>	izpildīt soli, neieejot apakšprogrammās
<Ctrl><F2>	pārtraukt programmas izpildi pa soļiem
<F4>	izpildīt līdz kursoram
<Ctrl><F8>	atzīmēt/noņemt apstāšanās vietu
<Ctrl><F4>	mainīgā vērtības uzstādīšana
<Ctrl><F7>	pievienot novērojamo mainīgo
<F5>	palielināt/samazināt logu
<Ctrl><F5>	pāriet uz logu pārvietošanas un izmēru maiņas režīmu (izmērus maina, turot <SHIFT>; iziet ar <ESC>)
<F6>	pāriet uz nākošo logu
<SHIFT><F6>	pāriet uz iepriekšējo logu
<ALT><F3>	aizvērt aktīvo logu
<ALT>0	atvērto logu saraksts
<ALT>1	pāriet uz 1. logu (pieļaujamie numuri no 1 līdz 9)
<F10>DO	atvērt izvades logu

Satura rādītājs