

# **Tipuri de date multime(set)**

## **Operatii cu tipuri de date structurate multime**

**Material didactic pentru Informatică**  
**(În corespondență cu curriculum-ul la Informatică)**  
**Clasa a X-a**

# Obiectivele lecției:

- O1** - să poată defini un tip mulțime;
- O2** - să poată declara variabile și constante de tip mulțime;
- O3** - să cunoască cum se scriu și cum se citesc mulțimile;
- O4** - să cunoască operațiile cu tipul set;
- O5** - să cunoască algoritmi de determinare a reuniunii, intersecției și diferenței mulțimilor și să le poată aplica la rezolvarea problemelor.

# Tipul Set (Mulțime)

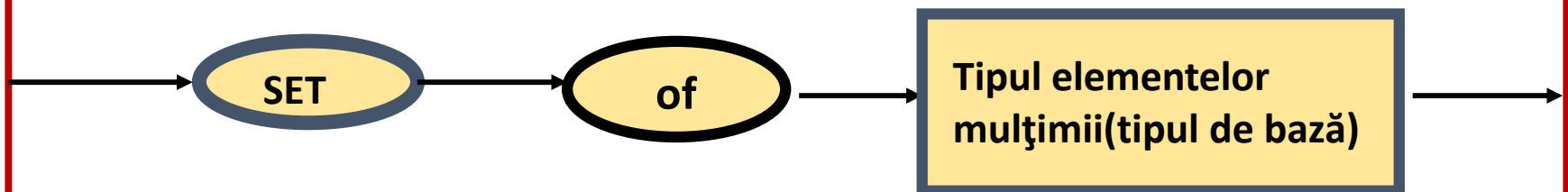
O mulțime (SET) PASCAL este o implementare limitată a conceptului matematic de mulțime. În cele ce urmează noi vom încerca să demonstrăm teoreme despre proprietățile mulțimilor.

# Diagrama de sintaxă a tipului SET

*Definirea tipului mulțime*

**<Tip mulțime> ::= [pached] set of <tip>**

**Declararea tipului mulțime**



# Exemple de declarații:

Type cifre=set of '0'..'9';

a=set of byte;

raspuns=set of boolean;

decor=set of (rosu,verde,alb,roz,galben);

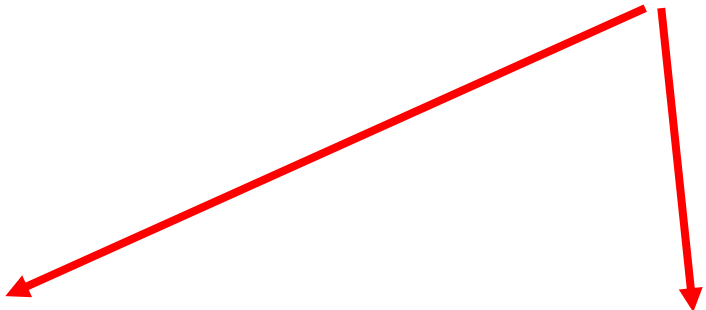
var d: decor;

c1,c2: cifre;

# Remarcă

- Limbajul Pascal limitează numărul de elemente ale unei mulțimi la 256;
- Tipul de bază al unei mulțimi de numere **NU** poate fi **integer**, **word** sau **longint**, ci numai subdomenii ale acestor tipuri cu cardinal cel mult 256 {0..255};
- Dacă tipul de bază este enumerare și conține zilele săptămânii, putem forma mulțimi de forma: **[luni, marti]** sau **[miercuri, joi, vineri]**.

**Pentru variabilele și constantele de tip mulțime sunt definite toate operațiile întâlnite în teoria mulțimilor: reuniune, intersecție, diferența.**



Matematică	Pascal
$c = a \cup b$	$c := a + b$
$c = a \cap b$	$c := a * b$
$c = a - b; d = b - a;$	$c := a - b; d := b - a$
$e \in a$ {da sau nu}	$e \text{ in } a$ {true sau false}

# Remarcă

1. Operatorul **IN** testează apartenența unui element la o mulțime;
2. În partea stângă se poate găsi o variabilă sau o expresie;
3. În partea dreaptă a operatorului **IN** trebuie să existe o expresie de tip mulțime;
4. Tipul membrului stâng trebuie să fie tip de bază al mulțimii din membrul drept.



# Remarcă

1. Tipul elementelor mulțimii trebuie să fie ordinal;
2. Dacă tipul de bază are  $n$  valori, atunci tipul mulțime va avea  $2^n$  valori.
3. Constructorul `[]` reprezintă mulțimea vidă.

# Exemplul 1

Alcătuiți un program ce afișează pe ecran rezultatele operațiilor +, \* și -, efectuate asupra valorilor de tip Mulțime.

# Exemplul 1

**Program Multime\_p1;**

**type Indice=1..10;**

**MultimeIndici=set of Indice;**

**var A, B, C : MultimeIndici;**

**i :integer;**

**begin**

**A:=[1..5, 8];**

**B:=[1..3, 9, 10];**

**C:=[];**

**C:=A+B;**

**writeln ('Reuniune');**

**for i:=1 to 10 do**

**if i in C then write(i:3);**

**writeln;**

**C:=A\*B;**

**writeln ('Intersectie');**

**for i:=1 to 10 do**

**if i in C then write(i:3);**

**writeln;**

**C:=A-B;**

**writeln ('Diferenta');**

**for i:=1 to 10 do**

**if i in C then write (i:3);**

**writeln;**

**readln;**

**end.**

# Operatorii aplicabili

=	<b>Egalitatea mulțimilor</b>	$A=B$ returnează true dacă $A=B$ altfel false
$\leq$	<b>incluziune</b>	$A \leq B$ returnează true dacă A se include în B, altfel false
$\geq$	<b>incluziune</b>	$A \geq B$ returnează true dacă B se include în A, altfel false
$\neq$	<b>Neegalitatea mulțimilor</b>	$A \neq B$ returnează true dacă A Diferit de B, altfel false

# Important

1. Datele de tip mulțime nu se pot citi cu procedurile **read, readln;**
2. Datele de tip mulțime nu se pot scrie cu ajutorul procedurilor **write, writeln.**

# Exemplul 2

Să se scrie un program care introduce de la tastatură două mulțimi de numere pozitive din două cifre și afișează reuniunea acestor mulțimi. Numărul de elemente pentru fiecare mulțime se citește de la tastatură.

**NB:** Se știe că elementele unei mulțimi nu pot fi citite în mod direct de la tastatură, deci vom folosi o variabilă auxiliară

# Exemplul 2

**Program Multime\_p2;**

**type Indice=1..99;**

**var A, B, C : set of Indice;**

**i , na, nb, aux : byte;**

**begin**

**Write('na='); readln(na);**

**A:=[];**

**For i:=1 to na do begin**

**readln(aux);**

**A:=A+[aux];**

**end;**

**Write('nb='); readln(nb);**

**B:=[];**

**For i:=1 to nb do begin**  
**readln(aux);**  
**B:=B+[aux];**  
**end;**

**C:=A+B;**

**writeln ('Reuniune');**

**for i :=10 to 99 do**

**if i in C then**

**write(i:3);**

**readln;**

**end.**

# Exemplul 3

Se consideră cuvântul  $X$  format din litere majuscule ale alfabetului latin. Să se scrie un program care determină literele ce apar o singură dată și literele ce apar de mai multe ori în cuvântul dat.

M a r i a G u t u



# Exemplul 3

**Program Multime\_p3;**

```
var X: string;  
apar1, apar2: set of char;  
J: char; i : integer;  
begin  
  Writeln('Introdu cuvantul');  
  readln(X);  
  apar1:=[]; apar2:=[];  
  For i:=1 to length(X) do  
    If X[i] in apar1 then  
      apar2:=apar2+[x[i]]  
    Else apar1:=apar1+[x[i]];  
  apar1:=apar1-apar2;
```

```
  writeln ('Literale ce apar o  
data');  
  for j := 'A' to 'Z' do  
    if j in apar1 then  
      write(j, ' ');  
  Writeln;  
  writeln ('Literale ce apar  
de mai multe ori');  
  for j := 'A' to 'Z' do  
    if j in apar2 then  
      write(j, ' ');  
end.
```

# Exemplul 4

Fie mulțimile  $A=\{1, 3, a, 4, c, d, 5, 8, 2\}$ ,  $B=\{2, a, c, 8, 4, 9, e, 3\}$ . Să se calculeze mulțimea:  $C=(A \cup B) - (A \cap B)$ .

# Exemplul 4

**Program Multime\_p4;**

**var a, b, c: set of char;**

**i: integer;**

**begin**

**A:=['1', '3', 'a', '4', 'c', 'd', '5', '8', '2'];**

**B:=['2', 'a', 'c', '8', '4', '9', 'e', '3'];**

**C:=(A+B)-(A\*B);**

**{Afişarea mulţimii C}**

**For i:=1 to 255 do**

**if chr(i) in C then write(chr(i), ' ');**

**End.**

# Exemplul 5

Se dă un număr natural  $n$ ,  $n < 20$ . Se citesc de la tastatură  $n$  mulțimi de numere naturale mai mici decât 100. Să se afișeze reuniunea și intersecția acestor mulțimi.

*Indicii:* Vom păstra mulțimile într-un vector. Inițial vom atribui mulțimii-intersecție toate numerele de la 0 la 100.

# Exemplul 5

Program Multime\_p5;

Type multime = set of byte;

var a: array [1..20] of multime;

    reun, inter: multime;

    i, n, el: byte;

Begin

  Writeln('Introdu nr. De multimi');

  Readln(n);

  For i:=1 to n do begin

    a[i]:=[];

    {Scriem elementele multimii i}

    Writeln('Multimea ', i);

    Repeat

      Readln(el);

      a[i]:=a[i]+el;

    until not(el in [0..100]);

  end;

  reun:=[]; inter:= [0..100];

  For i:=1 to n do begin

    reun:=reun+a[i];

    inter:= inter\*a[i];

  End;

  Writeln('Reuniunea');

  For i:=0 to 100 do

    if i in reun then write(i, ' ');

  Writeln;

  Writeln('Intersectia');

  For i:=0 to 100 do

    if i in inter then write(i, ' ');

  End.

# Concluzii

Spre deosebire de tablouri și articole, componentele cărora pot fi referite direct, respectiv prin indicii și denumiri de câmpuri, elementele unei mulțimi **nu pot fi referite**. Se admite numai verificarea apartenenței elementului la o mulțime (operația relațională in). În pofida acestui fapt, utilizarea tipurilor de date *mulțime* mărește viteza de execuție și îmbunătățește lizibilitatea programelor Pascal.

# Extindere 1

1. Se dă un text. Să se afișeze:
  - a. Vocalele care nu apar în text;
  - b. Consoanele care nu apar în text;
  - c. Cifrele care nu apar în text.
2. Se dă un text. Să se calculeze numărul:
  - a. Vocalelor;
  - b. Consoanelor;
  - c. Simbolurilor care nu sunt litere;
  - d. Literelor mici;
  - e. Literelor mari;
  - f. Cifrelor.

# Extindere 2

1. Se dau mulțimile  $X$  și  $Y$  de numere naturale mai mici decât 200. Să se determine:
  - a.  $X \cup Y$ ;
  - b.  $X \cap Y$ ;
  - c.  $(X \setminus Y) \cup (Y \setminus X)$ ;
  - d.  $(X \setminus Y) \cap (Y \setminus X)$ .
2. Considerând  $X, Y, Z$  mulțimi de numere naturale mai mici decât 200, să se verifice legile lui Morgan:
  - a.  $\overline{X \cup Y \cup Z} = \bar{X} \cap \bar{Y} \cap \bar{Z}$ ;
  - b.  $\overline{X \cap Y \cap Z} = \bar{X} \cup \bar{Y} \cup \bar{Z}$ ;



# Literatura recomandată:

1. Gremalschi, A.(2008). Informatică: Manual pentru cl. A 11-a, Editura Știința. 192 p.
2. Braicov, A.(2005). Turbo Pascal: culegere de probleme, Editura Prut Internațional. 232 p.
3. Sacara, A.(2012). Informatică: culegere de probleme pentru clasele a IX-a – a XII-a, Editura Epigraf. 88 p.
4. Creangă-Andrunache, E.(2001). Informatica: probleme Pascal, Editura Paragon. 219 p.