

## NAŠTEVNI IN INTERVALNI PODATKOVNI TIPI

Pascal sam po sebi pozna že nekaj osnovnih podatkovnih tipov.

boolean = (false, true);

integer = (-maxint, ... -2, -1, 0, 1, 2...maxint);

### Naštevni podatkovni tip

Deklaracija

type

ime\_tipa = zaloga\_vrednosti;

...

type

barva = (rdeča, zelena, modra);

mesec = (januar, februar, marec, april, maj, junij, julij, avgust, september, oktober, november, december);

dan = (ponedeljek, torek, sreda, četrtek, petek, sobota, nedelja);

Kot zaloga vrednosti ne mora imeti posebnih znakov (torej ime rdeča, četrtek, ... odpade).

Prvi podatek v zalogi vrednosti ima ordinalno vrednost 0.

Prerejanje vrednosti:

var

danes, jutri: dan;

begin

a := ponedeljek;

b := torek;

a := b;

b := succ(a); // b := pred(a);

writeln(a); // to ni mogoče in je nepravilno

potrebno je izpisati s pomočjo if ali case stavka

if a = ponedeljek then writeln('Ponedeljek');

if a = torek then writeln('Torek');

case b of

ponedeljek: writeln('Ponedeljek');

torek: writeln('Torek');

sreda: writeln('Sreda');

četrtek: writeln('Četrtek');

petek: writeln('Petek');

sobota: writeln('Sobota');

nedelja: writeln('Nedelja');

end;

end.

Izpisovanje na tak način je sicer nerodno, ampak žal v programskem jeziku pascal, za izpis naštevnega podatkovnega tipa, drugačnega ničina ni.

Funkcije za delo z naštevnim podatkovnim tipom:

ord(a) – ordinalna vrednost

pred(a) – predhodnjik

succ(a) – naslednjik

Prvi in zadnji element v zalogi vrednosti sta posebneža in vračata napake (-1 predhodnik prvega in število naslednjega za zadnjega. Ti dve vrednosti sta izven intervala zaloge vrednosti).

```
a := ponedeljek;  
writeln(pred(a)); // izpiše -1  
a := nedelja;  
writeln(succ(a)); // izpiše 7
```

Naslednji programska sekvenca tudi **ne deluje**, zaradi prekoračitve intervala znotraj naštevnega tipa. Paziti je potrebno pri obeh mejnih primerih (prvi in zadnji podatek naštevnega tipa).

```
type  
  barva = (belo,zeleno,modro,crno);  
var  
  a:barvA;  
begin  
  a := belo;  
  a := pred(belo);  
  // tukaj prevajalnik vidi, da hočemo vrniti predhodnjika prvega števila in javi napako  
  writeln(ord(a));  
  readln;  
end.
```

```
type  
  barva = (belo,zeleno,modro,crno);  
var  
  a:barvA;  
begin  
  a := crno;  
  a := succ(belo);  
  // tukaj prevajalnik vidi, da hočemo vrniti naslednjika zadnjega števila in javi napako  
  writeln(ord(a));  
  readln;  
end.
```

Za večanje/manjšanje lahko uporabimo tudi proceduro inc/dec nad podatkovnim tipom.  
inc(a);

Seštevanje lastnih naštevnihih podatkovnih tipov ni dovoljeno:

```
npr:  
var  
  a,b:barva;  
begin  
  writeln(ord(a+b));  
end;
```

Primerjanje vrednosti:

```
var  
  vceraj,danes,jutri:dan;  
  
begin  
  danes := torek;  
  vceraj := pred(danes);  
  jutri := succ(danes);
```

if (danes < sreda) then

stavek bo imel vrednost true, saj je torek glede na podani vrstni red manjši kot sreda.

## Intervalni tipi

Za primer vzamimo program, ki operira z datumi. Tak program mora poznati tudi dneve meseca. Ustrezni podatkovni tip bi lahko definirali kot naštevni tip naprimer z:

type

DanMeseca =

(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31);

Ta definicija je nerodna, ker je potrebno definirati vse dneve znotraj intervala in še problem nastopi pri izpisu podatkov. Pascal nam omogoča, da interval določimo elegantnejše na naslednji način:

Dan\_Meseca = 1..31;

Dan meseca je podinterval celih števil. Torej intervalni tipi so podmnožica drugih množic.

crke:'a'..'z'; // primer podmnožice števil – samo majhne črke od a do z;

## ZAPIS – RECORD

Kot podatkovni tip je strukturiran. Zelo kompleksen podatkovni tip in ima največje izrazne možnosti.

Vsebinska razlika:

### STRING

char	char	char	char	char
------	------	------	------	------

Vsi segmenti so tipa character

ENOLIČNO DOLOČENO!!!

### ARRAY

integer	integer	integer	integer	integer
---------	---------	---------	---------	---------

Segmenti so vsi istega tipa, ki ga mi pri deklaraciji določimo.

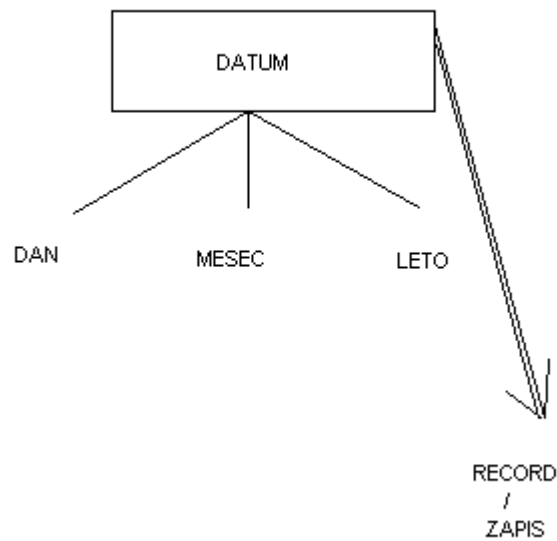
### RECORD

char	integer	string	boolean	integer
------	---------	--------	---------	---------

V tem podatkovnem tipu so lahko vsi segmenti različni. Določimo jih pri deklaraciji.

Predstavitveni model:

ARRAY	REAL	STRING	REAL	INTEGER
-------	------	--------	------	---------



## DEKLARACIJA

Deklaracija podatkovnega tipa zapis – RECORD je vezana skoraj vedno na posredni koncept naslavljanja.

TYPE

DATUM = RECORD

dan: 1.. 31;

mesec: (jan, feb, mar, apr, maj, jun, jul, avg, sep, okt, nov, dec);

leto:integer;

end;

var

a,b,c,d:datum;

a.dan := 1;

a.mesec := jan;

a.leto := 2005;

## DEKLARACIJSKA POSEBNOST NAD RECORDOM

TYPE

lik = (pravokotnik, trikotnik, krog);

lik = record

case vrsta:lik of

pravokotnik:(dolzina,visina:real);

trikotnik:(a,b,c:real);

krog:(radij:real);

end;

(\* primer, ko imamo osebo iz lokalne države in tujca. Za vsakega posebej vodimo podatke \*)

```

type
Oseba = record
  ime, priimek: string[40];
  datum: datum;
  case drzavljan: Boolean of
    True: (kraj_rojstva: string[40]);
    False: (drzava: string[20];
            datum_vnosa: string[20];
            datum_prihoda, datum_odhoda: datum);
end;

var
  vrsta: liki;

```

**CASE stavek lahko določimo samo enkrat v deklaracijskem bloku.**

```

TYPE
datum = record
  dan: 1.. 31;
  mesec: (jan, feb, mar, apr, maj, jun, jul, avg, sep, okt, nov, dec);
  leto: integer;
end;

var a,b: datum;

begin
  a.dan := 1;
  a.mesec := jan;
  a.leto := 2005;
  b := a; (* stavek pomeni enako kot če bi naredili
          b.dan := a.dan; b.mesec := a.mesec; b.leto := a.leto; *)

```

## **WITH STAVEK**

```

with a do
begin
  dan := 1;
  mesec := jan;
  leto := 2005;
end;

```