Programování

Polev Pascalu

There are
10 types
of people
in the world:

Those who
understand binary
and those
who don't.

Martin Urza

Motivace

- Při programování často pracujeme s mnoha entitami stejného typu.
  - Příklady: Postava v RPG ovládá 32 kouzel; v okně účetního programu je 100 faktur; klingonský dravec je vyzbrojen 14 disruptory; v balíku je 52 karet.
- K těmto entitám potřebujeme často přistupovat hromadně (například klingonský dravec bude pálit ze všech svých zbraní).
- Kdyby byla pro každou entitu vyhrazena zvláštní proměnná, programování by bylo velmi nepohodlné.
  - Programátor by pokaždé, když chce něco dělat se všemi entitami, musel napsat mnoho řádek kódu.

Co je pole?

- Pole je datový typ.
  - Obsahuje mnoho proměnných jednoho typu.
- K proměnným v poli přistupujeme pomocí **indexu**, což je číslo, které slouží k rozlišování mezi položkami pole.
  - Příklad: Pole o deseti položkách typu integer obsahuje deset proměnných typu integer, které jsou od sebe odlišeny svým očíslováním (typicky budou očíslovány čísly 0 až 9, případně 1 až 10, ale je možné i jinak).
- Proměnné v poli se implicitně nijak neovlivňují, je tedy možné s každou z nich pracovat zvlášť.
  - Jen ve stejnou chvíli vznikají a zanikají.

Syntaktický zápis pole

- Pole je typem nějaké proměnné, který deklarujeme klíčovým slovem array, za nímž následuje rozsah v hranatých závorkách, dále klíčové slovo of a poté název typu prvků pole:

```
var
  identifier : array [n..m] of type;
```

```
var
  x     : array [0..7]   of boolean;
  names : array [1..128] of string;
  nums  : array [32..69] of integer;
```

- V levém rámečku je *identifier* název proměnné, *n* a *m* jsou hranice pole (takže toto pole má $m - n + 1$ prvků) a *type* jsou typy prvků.
- V pravém rámečku jsou příklady jednotlivých polí různých typů (proměnná x je tedy pole, ve kterém je 8 hodnot typu boolean, names obsahuje 128 hodnot typu string, nums obsahuje 38 integerů).

Přístup do pole

- Do jednotlivých položek pole lze přistupovat pomocí jejich indexů v hranatých závorkách.
  - Tato položka se pak chová jako proměnná, tedy je do ní možné přiřazovat, i je možné hodnotu z ní přečíst.
    - Je-li *a* pole typu *boolean* a *b* platný index do pole, pak *a[b]* se chová jako proměnná typu boolean.
- Pokusí-li se program přistoupit na neplatný index, spadne.
  - Př.: V poli 0..3 je neplatný index 8.

```
var
  foo : array [4..7] of char;
  bar : array [0..3] of boolean;
```

```
foo[4] := bar[0];
x      := foo[7];
bar[2] := true;
for x := 0 to 3
  do bar[x] := 0;
```

```
program revert8;

var
  nms : array [1..8] of integer;
  loop1 : integer;

begin
  writeln('Zadejte cisla.');

  for loop1 := 1 to 8
    do readln(nms[loop1]);

  writeln('Reverzni vypis:');

  for loop1 := 8 downto 1
    do writeln(nms[loop1]);

  readln();
end.
```

Cvičení

- Napište program, který přečte posloupnost maximálně osmi čísel ukončenou nulou (není-li nula zadána, posloupnost automaticky skončí v případě, že má délku 8). Poté posloupnost vypíše pozpátku.
- ** Upravte program, aby při výpisu vynechával čísla menší než nejvyšší doposud vypsané. Například posloupnost 2, 32, 8, 16, 4 je převrácena na 4, 16, 8, 32, 2. S vynechanými čísly pak 4, 16, 32.*

```
program revert;


var
  n : array [0..7] of integer;
  loop1 : integer;


begin

  writeln('Zadejte cisla.');

  loop1 := 0;
  readln(n[loop1]);

  while ((0 < n[loop1]) AND
         (loop1 < 7))
    do
    begin
      loop1 := loop1 + 1;
      readln(n[loop1]);
    end;

  writeln('Reverzni vypis:');

  for loop1 := loop1 downto 0
    do writeln(n[loop1]);

end.
```

Proč vůbec používat pole?

- Mnohé možná napadlo, že by nebylo vůbec nutné používat pole, když se jednotlivé položky polí chovají ve všech ohledech stejně jako obyčejné proměnné.
  - Toto je do jisté míry pravda, protože skutečně vše, co lze napsat pomocí statických polí, je možné napsat i bez nich, jen za použití mnoha proměnných.
    - Rozdíl je v počtu řádků takového kódu. Zkusíte-li si napsat cvičení na pole bez použití polí, bude výsledný kód řádově mnohonásobně delší.
- Největší výhodou polí je, že do nich lze přistupovat v cyklu, což do ostatních proměnných nelze.
  - Proměnné *a, b, c, d* nelze procházet v cyklu. Pole o rozměrech 0..3 procházet cyklem lze.

Rekapitulace

- Pole je datový typ, který obsahuje předem zvolený počet proměnných jednoho typu.
- K proměnným v poli přistupujeme pomocí indexu, což je číslo, kterým rozlišujeme jednotlivé položky pole.
- Položky pole se chovají jako proměnné, lze do nich číst i zapisovat.
- Jednotlivé položky v poli jsou na sobě nezávislé, tedy změna kterékoliv položky nijak neovlivní žádnou jinou.
- Při pokusu o přístup na neplatný index program skončí chybou.