VARTOTOJO TIPAI IR ATMINTIES NAUDOJIMAS

Be jau žinimų struktūrinių duomenų tipų (masyvų, simbolių eilučių, struktūrų) C/C++ kalbos suteikia ir kitas žemiau išvardintas galimybes kurti vartotojo duomenų tipus:

- Bitų laukai ***bit fields*** ;
- Junginiai ***union*** ;
- Išvardijimas ***enumeration*** ;
- Tipų pseudonimai ***typedef*** .

BITŲ LAUKAI

Tai atminties taupymo priemonė, tačiau bitų laukų realizacija priklauso nuo platformos (kompiuterio architektūros) ir kompiliatoriaus.

Bitų laukų taikymo pavyzdys. Jie reikalinga struktūra, kurioje būtų saugoma informacija apie tam tikro įvykio datą ir laiką, ją galima būtų apibrėžti taip:

```
struct DataLaikas {
    unsigned short   Metai;
    unsigned short   Menuo;
    unsigned short   Diena;
    unsigned short   Valanda;
    unsigned short   Minute;
    unsigned short   Sekunde;
  } d1;
```

C kalboje ši struktūra užimtų 6x2=12 baitų atminties. Tačiau suprantama, kad tai perteklinis naudojimas, atminties galima sutaupyti. Jei *Metai* gali turėti reikšmes iš intervalo 0..99, tai jiems saugoti užtektų 7 bitų atminties, *Menuo* – 4 bitų, *Data* – 5 bitų ir t.t. Taip tiksliai skirti reikiamą atmintį galima naudojant bitų laukus:

```
struct DataLaikasB {
    unsigned   Metai       : 7;
    unsigned   Menuo       : 4;
    unsigned   Diena       : 5;
    unsigned   Valanda     : 6;
    unsigned   Minute      : 6;
    unsigned   Sekunde     : 6;
  };
```

Tokia bitų laukų struktūra *DataLaikasB* pareikalautų tik 5 baitų (34 bitų) atminties.

JUNGINIAI

Kita atminties taupymo priemonė yra junginiai, t. y. struktūra su kintamais laukais, **union.** Junginiai taikomi kai reikia į tą pačią atminties vietą talpinti skirtingo tipo duomenis. Vienu metu šioje atmintyje bus saugoma tik vieno lauko reikšmė.

```
union ManoDuomenys {
        int       Metai;
        char      Pavadinimas[10];
        unsigned long    Alga;
        bool      Atsakymas;
        };
```

Tokiai duomenų struktūrai kompiliatorius skiria ilgiausio lauko dydžio atminties vietą (šiuo atveju – 10 baitų). Junginiai dažnai įeina į kitas struktūras, pvz., apibūdinat skirtingas geometrines figūras grafiniame redaktoriuje. Taip pat junginiai naudojami duomenų tipų transformacijai, nes tuos pačius atminties baitus galima traktuoti taip kaip reikia programuotojui.

IŠVARDIJIMAS

Išvardijimas **enumeration** – tai iš esmės vardinių sveikojo tipo konstantų rinkinys. Išvardijimo skelbimas panašus į struktūrų skelbimą:

```
enum coin { penny, nickel, dime, quater, half-dollar, dollar };
```

Čia *coin* yra išvardijimo tegas (žymuo), kurį galima naudoti skelbiant išvardijimo tipo kintamuosius:

```
enum coin money;
```

Dabar kintamasis *money* gali būti naudojamas, pvz.:

```
money = nickel;
if (money == nickel) printf("Gaila, turiu tik 10 centų \n");
```

Kiekvienas išvardijimo elementas turi vienetu didesnę sveikojo tipo reikšmę nei prieš tai esantis, pirmas elementas (čia *penny*) turi reikšmę 0.

TIPŲ PSEUDONIMAI

Iš tikrųjų, tipų pseudonimai **typedef** nesukuria naujų duomenų tipų, o tik suteikia sinonimus esantiems tipams:

```
typedef  float  real;
typedef  int  atstumas;
```

Tipų pseudonimai dažniausiai naudojami apibrėžiant struktūras.