Složeni tipovi podataka

strukture, unije, (polja bitova, pobrojavanja -enumeracije)

Strukture
(tip `struct`)

Podatkovni tip `struct`

- **Derivirani - izvedeni podatkovni tip** – stvara ga programer
- grupa logički povezanih podataka **različitih** podatkovnih tipova (npr. `char`, `int`, `double`) sa zajedničkim imenom
- podaci (članovi strukture) su međusobno povezani identifikatorom – imenom strukture; deklaracijom članova strukture određuje se ime i tip pojedinih članova strukture
- ako je definiran izvan tijela funkcije `main()` – sve funkcije programa mogu koristiti taj tip
- između imena strukturne varijable i imena člana te strukture stavlja se operator '.'

Strukture i njihovo korištenje (msdn)

- struktura je korisnički definiran složeni tip
- sastavljen je od polja ili članova koji mogu biti različitih tipova
- u C++, struktura je ista kao i klasa osim što su njeni članovi podrazumijevano javni
- varijable se kod definicije strukturnog tipa može deklarirati nizanjem jednog ili više imena varijabli odvojenih zarezima između zatvorene zagrade i ;

Strukture - primjeri

```
struct OSOBA                    // deklaracija strukture OSOBA
{int starost;                   // deklaracije tipova članova
float tezina;
char ime[25]; } clan_obitelji;  // definira objekt tipa OSOBA

OSOBA brat;
OSOBA sestra;                   // C++ stil deklaracije strukture
sestra.starost = 13;
brat.starost = 7;               // pridružuje vrijednosti članovima

struct TOCKA                    // deklaracija strukture TOCKA
{ int x;                        // definira članove x i y
  int y; } mjesto = { 20, 40 }; // varijabla mjesto ima vrijednosti x = 20, y = 40

TOCKA tamo;                     // varijabla tamo je tipa TOCKA
```

Vježba

- **Deklarirati strukturu “sportski_klub” koja će moći pamtiti podatke za neki sportski klub, i to:**
  - Naziv kluba (polje znakova ili string)
  - Grad iz kojeg je klub
  - Ime i prezime trenera
  - Broj igrača
  - Broj postignutih pogodaka
  - Broj primljenih pogodaka
  - Da li je član prve lige
  - Iznos godišnjeg “proračuna”
- **Deklarirati dvije varijable tog tipa i pridružiti im vrijednosti**

Rješenje

```
struct KLUB

{string NAZIV_KLUBA;
     string GRAD;
      char ime[100];
     int BROJ_IGRACA;
     int BROJ_POSTIGNUTIH_GOLOVA;
     int BROJ_PRIMLJENIH_GOLOVA;
     bool PRVA_LIGA;
     float GODISNJI_PRORACUN;
     } klub1,klub2;

KLUB klubx;


klub1.NAZIV_KLUBA="PIPO IPC";
     klub1.GRAD="ČAKOVEC";
     klub1.PRVA_LIGA=true;

klub2.NAZIV_KLUBA="CROATIA ZAGREB";
     klub2.GRAD="ZAGREB";
```

`struct` – definiranje, deklariranje, upotreba varijabli

```
struct Tekucina
{
   public:                                          (ili private)
         char oznaka;
         double temp, gustoca;
};

int main()
{
   Tekucina voda, ulje, mineralna;
   voda.oznaka = 'V';
   voda.temp = 28.6;
   voda.gustoca = 9.81;

   mineralna = voda;

   cout << "voda: "<< voda.oznaka << endl; ...}
```

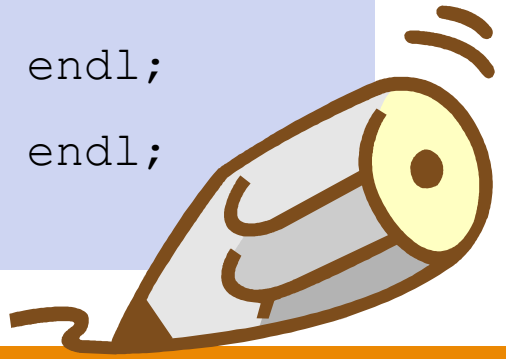
Deklariranje `struct` tipa i `struct` varijabli

- deklaracijom određujemo koji će se tipovi informacija moći spremiti u memoriju rezerviranu za **strukturnu varijablu** tog tipa (za sve podatkovne članove te strukture)
- strukturna varijabla je složena varijabla koja sadrži niz vrijednosti istog ili različitog tipa
- deklaracijom **strukture** određuje se **oblik strukture** i ne zauzima se memorijski prostor; deklaracijom **strukturne varijable rezervira se memorijski prostor**
- za spremanje vrijednosti podatkovnih (public) članova pišemo ime `struct` varijable (objekta), točku (strukturni operator) i ime podatkovnog člana

Pridruživanje i ispis vrijednosti podatkovnih članova

- vrijednosti **svih članova** jedne `struct` varijable možemo pridružiti članovima druge istovrsne `struct` varijable jednom naredbom pridruživanja
- za ispis vrijednosti **pojedinih** podatkovnih članova `struct` varijable (kao i za sve ostale operacije nad strukturama) moramo navoditi **svaki član pojedinačno** upotrebom operatora (.), npr.:

```
cout << "voda: "<< voda.oznaka << endl;
cout << "temperatura: "<< voda.temp << endl;
cout << "gustoca: " << voda.gustoca << endl;
```



Usporedba struktura i razreda

- **u C++ strukture imaju ista svojstva kao razred**
- **način deklaracije je isti s tom razlikom što se koristi ključna riječ `struct`**
- **struktura može sadržavati podatkovne i funkcijske članove, konstruktore, destruktore i ključne riječi za dodjelu prava pristupa**
- **razlika:**
  - ako se ne navede pravo pristupa, članovi imaju podrazumijevani javni pristup
  - strukture ne omogućuju nasljeđivanje
- **nakon što se usvoji koncept razreda, strukture se obično izbacuju iz upotrebe**

Primjeri …

- Napraviti program koji će uz upotrebu strukture izračunati opsege dva pravokutnika, pa ispisati koji pravokutnik ima veći opseg. Duljine stranica su određene realnim brojevima. (Pravokutnike se može označiti slovima p, odnosno r.)

- **Napraviti program koji će uz upotrebu strukture izračunati površinu dva trokuta zadanih stranica čije su duljine realni brojevi. Ispisati za oba trokuta površinu. Ispisati koliko puta drugi trokut ima veću/manju površinu od prvoga.**

Primjeri - 1

- Napravite program koji (uz upotrebu strukture Kompleksni) omogućuje učitavanje dvaju kompleksnih brojeva, pa računa i ispisuje njihov zbroj, razliku, produkt i kvocijent.
- Napravite program koji će izračunati udaljenost točke T od ishodišta koordinatnog sustava. Za zadavanje točke iskoristite strukturu.
- **Napravite program koji će učitati dvije točke u koordinatnom sustavu. Program treba izračunati opseg i površinu pravokutnika kojemu su učitane točke vrhovi dijagonale. Koristite se strukturama.**

Primjeri - 2

- Napravite program koji će omogućiti unošenje podataka (ime, prezime, matični, broj, prosjek, datum rođenja(struktura)) za sve učenike jednog razreda. Program treba ispisati koji učenik ima najbolji prosjek. Uputa: Učenike smjestite u polje.
- Napravite program koji će iz datoteke učitavati imena i prezimena učenika te njihove ocjene iz pismenog ispita. Program treba ispisati koliko učenika je dobilo odličan, koliko vrlo dobar, dobar, dovoljan i nedovoljan te kolika je prosječna ocjena razreda iz pismenog. Ispisati imena i prezimena učenika koji su dobili odličan.

Unije

Unije - uvod

- posebni tipovi podataka koji omogućavaju smještaj više različitih tipova podataka na isto mjesto u memoriji
- varijable koje u različitim dijelovima programa mogu sadržavati vrijednosti različitih tipova podataka

Primjer

- Unije imaju isti oblik i operatore kao strukture. Za razliku od struktura koje rezerviraju odvojene dijelove memorije za svaki podatkovni član, unije alociraju tek dovoljno memorije da zadovolje najveći podatkovni član. Na 16-bitnim i 32-bitnim računalima, npr. definicija

union jack {

long number;

char chbuf[4]; } chunk;

Alocira samo četiri bajta memorije. Varijabla chunk.number je long tipa, a chunk.chbuf[2] je char, obje u istom memorijskom području.

Unije - karakteristike

- pojedinim članovima unije pristupa se potpuno isto kao i članovima strukture, dakle direktan pristup članovima strukture ostvaruje se korištenjem operatora '.'
- dozvoljene operacije nad unijama identične su operacijama nad strukturama
- deklariraju se slično razredima i strukturama s tom razlikom što se koristi ključna riječ `union`
- mogu sadržavati podatkovne i funkcijske članove i riječi za dodjelu prava pristupa
- bitna razlika u odnosu na razrede je što **svi podatkovni članovi unije dijele isti memorijski prostor** (prilikom pridruživanja vrijednosti jednom podatkovnom članu prepisuje se vrijednost podatkovnog člana koji je rabio ranije upisan)

Unije

- kako svi članovi unije dijele isti memorijski prostor, veličina unije jednaka je veličini njenog najvećeg člana
- vrlo je važno biti oprezan prilikom korištenja unije
- Očuvanje cjelovitosti podataka za članove unije je odgovornost programera.
- unije su se intenzivno koristile u vrijeme kada su memorije na računalima bile skučene, te je trebalo maksimalno iskoristiti svaki raspoloživi bajt
- Nisu uvijek prenosive
- danas se one vrlo rijetko koriste

The union object occupies as much space as the largest member, whereas structures require space equal to at least the sum of the size of its members.

Anonimne unije

- C++ ima i anonimne unije (alocira memoriju za svoj najveći podatkovni član, ali ne kreira ime tipa).

union {

Type data_member1;

Type data_member2;

Type data_memberN; };

Program koji deklarira anonimnu uniju može pristupiti podatkovnim članovima izravno (bez . or ->). Funkcijski članovni nisu dozvoljeni unutar definicija anonimnih unija.

Primjer anonimnih unija - itoh()

- **Radi konverziju varijable tipa short u hexadecimalni znak**

```
void itoh(unsigned short n)
{ union {
unsigned short num;
unsigned char s[sizeof(short)]; };
const char *hex = "0123456789abcdef";
num = n; // store number as a short
cout << hex[s[1] >> 4]; // first byte - high nibble
cout << hex[s[1] & 15]; // first byte - low nibble
cout << hex[s[0] >> 4]; // second byte - high nibble
cout << hex[s[0] & 15]; // second byte - low nibble }
//izvodi se brzo, nema rekurzivnih poziva, petlji, operatora množenja ili dijeljenja, većinom mali
    asemblerski kod, pogodnije za programe sa memorijskim ili vremenskim ograničenjima
```

Polja bitova

C++ Bit Fields

Polja bitova

- razredi i strukture mogu sadržavati članove koji zauzimaju manje memorije od cjelobrojnog tipa – nazivamo ih polja bitova
- svakom članu strukture pridružuje se određeni broj bitova
- bit-field je podatkovni član strukture ili razreda koji sadrži 1 ili više bitova:

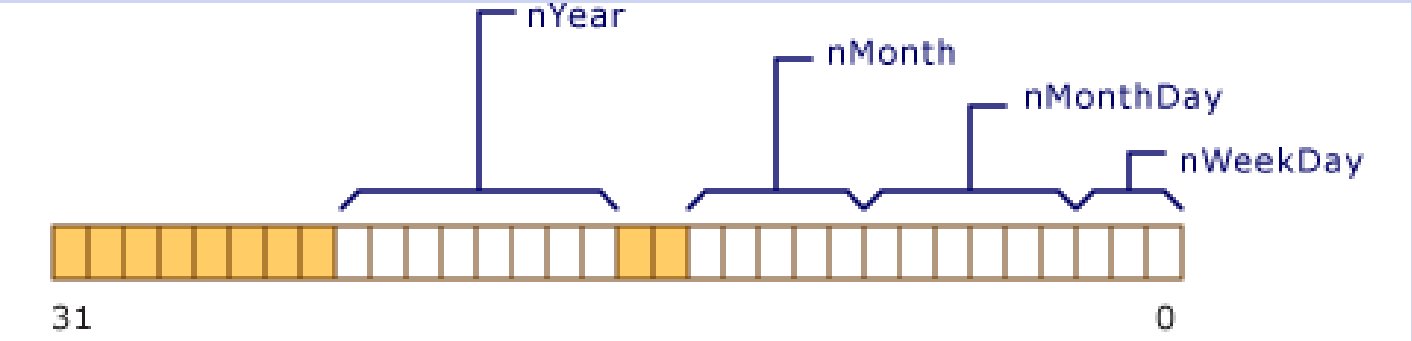
```
struct A {

unsigned int f1: 4; // 0 - 15

unsigned int f2: 4; }; //u deklaraciji se piše prvo temeljni tip (signed ili
    unsigned char, short, int, long i bool) pa ime polja i zatim : sa
    brojem bitova
```

Karakteristike

- Polja bitova osiguravaju mehanizam za optimizaciju upotrebe memorije dozvoljavajući određivanje točnog broja bitova potrebnih za spremanje podataka.
- Prilično korisno kod programiranja kad je prisutno memorijsko ograničenje (npr. mobilni uređaji, telefoni)
- Deklaracija članova tog tipa slijedi sintaksu “ime varijable : broj bitova".
- Neimenovana polja bitova širine 0 koriste se za poravnanje slijedećeg polja bitova granicama tipa polja.

Primjer

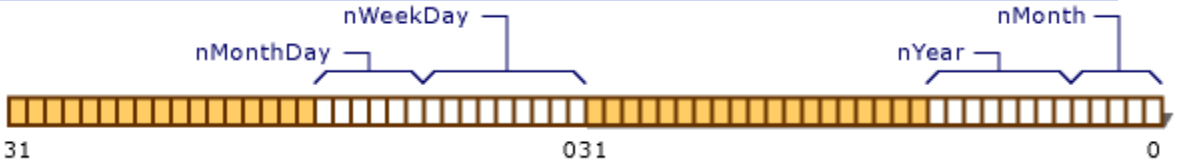
```
struct Date {
unsigned short nWeekDay : 3; // 0..7 (3 bits)
unsigned short nMonthDay : 6; // 0..31 (6 bits)
unsigned short nMonth : 5; // 0..12 (5 bits)
unsigned short nYear : 8; // 0..100 (8 bits) };
```



Neimenovano bitovno polje

- Neimenovano bitovno polje širine 0 prisiljava poravnanje slijedećeg bitovnog polja

```
struct Date {
unsigned nWeekDay : 3; // 0..7 (3 bits)
unsigned nMonthDay : 6; // 0..31 (6 bits)
unsigned : 0; // prisiljava poravnanje
unsigned nMonth : 5; // 0..12 (5 bits)
unsigned nYear : 8; // 0..100 (8 bits) };
```



Primjedbe

- Kompajler pakira susjedne bitove istog cjelobrojnog tipa u poretku u kojem se bitfield članovi pojavljuju unutar strukture ili definicije razreda. Treba uvažiti da poredak ovisi o računalu (može biti okrenut).
- Bitfield članove treba koristiti pažljivo i pamtiti da ne štede uvijek memoriju i nemaju uvijek dobitak u brzini.