Osnovni tipovi podataka

Stefan Panić
1055/2104

Uvod

- Problemi i greške koje se najčešće javljaju pri korišćenju osnovnih tipova podataka
- Saveti za izbegavanje i otkrivanje grešaka pri upotrebi osnovnih tipova podataka
- Kako kreirati korisnički definisane tipove

Sadržaj

1) Numerički tipovi
2) Karakteri i stringovi
3) Boolean
4) Enumerated
5) Konstante
6) Nizovi
7) Korisnički definisani tipovi

1) Numerički tipovi

1) Koristiti konstante ili globalne promenljive umesto brojeva (literala)
   - Prednosti:
     - Lakša i sigurnija izmena koda
     - Pouzdaniji kod
2) Predvideti deljenje nulom
3) Koristiti očiglednu konverziju tipova
4) Izbegavati poređenje različitih tipova podataka
5) Obratiti pažnju na upozorenja kompajlera

1.1) Celobrojni tipovi - integers

1) Proveriti celobrojno deljenje
2) Voditi računa o prekoračenju

Integer Type	Range
Signed 8-bit	-128 through 127
Unsigned 8-bit	0 through 255
Signed 16-bit	-32,768 through 32,767
Unsigned 16-bit	0 through 65,535
Signed 32-bit	-2,147,483,648 through 2,147,483,647
Unsigned 32-bit	0 through 4,294,967,295
Signed 64-bit	-9,223,372,036,854,775,808 through 9,223,372,036,854,775,807
Unsigned 64-bit	0 through 18,446,744,073,709,551,615

1.2) Brojevi u pokretnom zarezu - floating-point numbers (1)

1) Razmotriti preciznost zapisa brojeva u pokretnom zarezu
2) Izbegavati sabiranje i oduzimanje brojeva koji imaju veliku razliku reda veličine
3) Izbegavati poređenje jednakosti

```
double  a  = 1.0;
double  b  = 0.0;
for ( int i = 0; i < 10; i++ )
{
        sum += 0.1;
}
if ( a == b )
        printf (“Equal! \n”);
else
        printf(“Not equal! \n”);
```

```
0.1
0.2
0.30000000000000004
0.4
0.5
0.6
0.7
0.8
0.79999999999999
0.89999999999999
0.99999999999999
```

1.2) Brojevi u pokretnom zarezu – floating-point numbers (2)

4) Predvideti problem zaokruživanja
5) Koristiti podršku jezika i biblioteka za rad sa specifičnim tipovima podataka

2) Karakteri i stringovi

1) Koristiti konstante i globalne promenljive umesto stringova literala
   - Brže i efikasnije izmene koda
   - Mesto u memoriji
   - Uredan i razumljiviji kod
2) Voditi računa o granicama stringa (niza karaktera)
3) Odabrati kodnu stranu i biblioteke koje rade sa odabranom kodnom stranom

2.1) Stringovi u C-u

- Razlikovati stringove i pokazivače na stringove
- Uključiti terminirajuću nulu u dužinu stringa
- Inicijalizovati stringove na NULL vrednost i postavljati terminirajući karakter
- Ukoliko je moguce koristiti nizove stringova umesto pokazivača
- Koristiti strncpy( ) umesto strcpy( )

3) Boolean tipovi

1) Koristiti boolean vrednosti za dokumentovanje programa
2) Koristiti boolean vrednosti za pojednostavljivanje testova
3) Kreirati boolean vrednosti ukoliko je potrebno

```
typedef  int BOOLEAN;     // define the boolean type
```

4) Enumerated tipovi (1)

- Primer (C#):

```
public enum Days {
        Monday,
        Tuesday,
        Wednesday,
        Thursday,
        Friday,
        Saturday,
        Sunday
};
```

4) Enumerated tipovi (2)

1) Koristiti nabrojive tipove zbog čitljivosti
2) Koristiti nabrojive tipove zbog pouzdanosti
3) Nabrojivi tipovi olakšavaju izmene u kodu
4) Koristiti nabrojive tipove kao alternativu za boolean tip
5) Proveravati nabrojive tipove za nedozvoljene vrednosti
6) Definisati prvi i poslednji član nabrojivih tipova radi korišćenja u petljama

4) Enumerated tipovi (3)

- Ukoliko programski jezik nema nabrojive tipove koristiti globalne promenljive ili klase kao alternativu (Java primer):

```
class Color {
        private Color( ) { }
        public static final Color Red = new Color ( );
        public static final Color Green = new Color ( );
        public static final Color Blue = new Color ( );
}
```

5) Konstante

1) Koristiti konstante prilikom deklaracije promenljivih
2) Koristiti konstante umesto brojeva i literala
3) Imenovati konstante da budu razumljive
4) Konstante koristiti konzistentno

6) Nizovi

1) Obratiti pažnju na granice niza
2) Razmišljati o nizovima kao linearnim strukturama
3) Proveriti promenljive koje se koriste kao indeksi višedimenzionih nizova i njihov redosled
4) Pažljivo sa indeksima nizova u petljama
5) Koristiti makroe i konstante za dimentizije nizova

7) Korisnički definisani tipovi (1)

- Korisnički definisani tipovi su veoma korisno i moćno oružje u procesu programiranja
- Pomažu pri razumevanju programa i štite ga od nepredviđenih grešaka
- Prednosti:
  1) Olakšavaju izmene i donose mnogo fleksibilnosti
  2) „Kriju“ informacije od drugih objekata, funkcija...
  3) Povećavaju pouzdanost
  4) Upotpunjuju slabosti jezika

7) Korisnički definisani tipovi (2)

- Saveti za kreiranje korisnički definisane tipove:
  1) Imenovati korisnički definisane tipove prema funkciji koji obavljaju
  2) Ukoliko je moguće koristiti korisnički definisane tipove pre nego predefinisane tipove
  3) Izbegavati izmenu predefinisanih tipova (npr. Integer, String...)
  4) Umesto izmena postojećih tipova definisati alterantivni tip (npr. int -> INT)
  5) Koristiti klase pre nego typedef

KRAJ!