Programski jezici i strukture podataka

2

Program u jeziku C

- Program u jeziku C se sastoji od proizvoljno mnogo izvornih datoteka i datoteka zaglavlja.
- Manje zahtevni programi (u smislu količine koda i broja funkcija) pišu se u jednoj izvornoj datoteci.
- **Izvorna datoteka** se uobičajeno sastoji iz:
  - pretptocesorskih direktiva,
  - globalnih deklaracija i
  - definicija funkcija.

Izvorna datoteka

- Izvorna datoteka se uobičajeno označava imenom sa sufiksom .c. (**z1.c**)
- Ako se program sastoji od više izvornih datoteka uobičajeno je da se deklaracije funkcija, globalne promenljive, makroi i konstante izdvoje u posebnu datoteku "zaglavlja", koja se pretprocesorskom direktivom `#include` poziva u svakoj izvornoj datoteci.
- Datoteke zaglavlja se označavaju imenom i sufiksom .h.

Izvorni kod

- Izvorna datoteka se sastoji od izvornog koda.
- Izvorni kod započinje pretprocesorskim direktivama.
- U pojedinačnom redu se može nalaziti samo jedna predprocesorska direktiva, red započinje znakom #.
- Između dva susedna tokena može biti proizvoljan broj razmaka i tabulatora.
- Uobičajeno je:
  - novu deklaraciju i naredbu započeti u novom redu,
  - uvući ugnežđenu strukturu ili blok naredbi.

Komentari

- Deo koda koji se ne kompajlira
- Blok komentar /* Kod koji se ne kompajlira. */
- Linijski komentar // Sve iza dve kose crte se ne kompajlira.
- Pretprocesor svaki komentar zamenjuje jednim praznim mestom.

```
test1/* test2*/test3
```

pretprocesor tumači kao:

```
test1 test3
```

Identifikatori

- Imena:
  - promenljivih,
  - funkcija,
  - makroa,
  - struktura,
  - konstanti,
  - drugi objekti definisani u C prog jeziku.

Pravila za izražavanja identifikatora

- Identifikator se sastoji od:
  - Slova iz osnovnog skupa znakova, uz razlikovanje malih i velikih slova.
  - Donje crte _.
  - Decimalne cifre, prvi znak identifikatora ne sme biti cifra.
- Maksimalan broj karaktera u identifikatoru je po standardu 31, ali u mnogim prevodiocima dužina nije ograničena. Prevodilac zapravo radi sa prvih n značajnih znakova a ostale zanemaruje.

Unapred definisan identifikator

- `main` Imenuje funkciju od koje počinje izvršavanje programa.
- `__func__` (po dve donje crte bez razmaka) Služi za pristupanje stringu koji sadrži ime funkcije.
- Upotreba prilikom obrade izuzetaka (**z2.c**)

Prostori imena identifikatora

- Imena labela (**z3.c**)
- Tagovi (strukture, unije, nabrojani tipovi)
- Imena članova unija i struktura.
- Ostali identifikatori

Oblasti važenja identifikatora

- file scope (oblast važenja fajl)
- block scope (oblast važenja blok)
- function prototype scope (oblast važenja prototip)
- function scope (oblast važenja funkcija) (**z4.c**)

- Identifikator se može ponovo upotrebiti u oblasti važenja, koja je ugnježdena u odnosu na njegovu izvornu oblast važenja. (**z5.c**)

Mnemonički identifikatori

- Izaberite identifikatore koji su mnemonički kako bi program bio lakši za čitanje, vama i drugima.
- Uporedi:

  void fun1(int arg1, int arg2, float arg3)

  void crtajKrug(int x, int y, float poluprecnik)
- Neka su identifikatori samoopisujući.

Ključne reči

auto	double	int	struct	_Bool
break	else	long	switch	_Complex
case	enum	register	typedef	_Imaginary
char	extern	return	union	_Noreturn
const	float	short	unsigned	
continue	for	signed	void	_Static_assert
default	goto	sizeof	volatile	inline
do	if	static	while	restrict

- Ključne reči su rezervisane i ne mogu biti identifikatori
- Takođe se za identifikatore ne mogu uzimati nazivi funkcija, konstanti i makroa iz standardne biblioteke.

Promenljive

- Promenljiva je imenovani objekat koji zauzima memorijski prostor određen tipom.
- Ime promenljive je identifikator.
- Promenljiva pre korištenja treba biti deklarisana.

Tipologija

- Osnovni tipovi (aritmetički tipovi)(skalarni tipovi)
  - celobrojni
  - realni i kompleksni
- Nabrojivi tipovi (aritmetički tipovi)
- Tip void
- Izvedeni tipovi
  - pokazivači (skalarni tipovi)
  - nizovi (agregati)
  - strukture (agregati)
  - unije
  - funkcije

Označeni celobrojni tipovi

- **signed char**
- **int** (signed, signed int)
- **short** (short int, signed short, signed short int)
- **long** (long int,signed long, signed long int)
- **long long** (long long int, signed long long, signed long long int)

Neoznačeni celobrojni tipovi

- _Bool (bool)
- unsigned char
- unsigned int (unsigned)
- unsigned short (unsigned short int)
- unsigned long (unsigned long int)
- unsigned long long

Uobičajene veličine pojedinih tipova i opsezi

- **`Operator`** `sizeof()`
- **datoteka** `limits.h`
- **makroi** `INT_MIN, INT_MAX, UINT_MAX…`

Tipovi u formatu sa pokretnim zarezom

- float (4B) prciznost 6 cifara
- double (8B) prciznost 15 cifara
- long double (10B) prciznost 19 cifara

Kompleksni brojevi

- float `_Complex`
- double `_Complex`
- long double `_Complex`
- complex.h
- `complex` sinonom za `_Complex`
- makro `I` je imaginarna jedinica (**z6.c**)

Nabrojivi tipovi

- `enum [identifikator]{lista vrednosti};`

- `enum dani {pon, ut, sr, cet, pet, sub, ned};`

- Svaka konstanta u nabrajanju ima svoju vrednost (**z7.c**).

Tip Void

- Nema konstanti i promenljivih tipa `void`
- funkcije tipa `void`
- lista parametara funkcije može biti `void`
- izrazi mogu biti tipa `void`
- pokazivač može biti na tip `void` (**z8.c**)

Operacije celog tipa

- Sabiranje, oduzimanje, množenje i delenje.
- Za obavljanje svake od ovih operacija potrebna su dva cela broja - **operandi**.
- Oznaka operacije se naziva **operator**.
- Kao operatori sabiranja, oduzimanja, množenja i delenja koriste se znakovi +, -, *, i /, respektivno.

Redosled izvršavanja

- Primena operatora ***** i **/** uvek prethodi primeni operatora **+** i **-**. Kažemo * i / su prioritetniji od operatora **+** i **-**.
- Operatori ***** i **/** su istog prioriteta i primenjuju se u redosledu pojave u izrazu (s leva u desno).
- Operatori **+** i **-** su istog prioriteta i primenjuju se u redosledu pojave u izrazu (s leva u desno).
- Radi uticanja na redosled primene operatora, uvedene su **male zagrade**.
- Primena operatora u malim zagradama obavezno prethodi primeni operatora koji su van njih.

Logički tip

- U C programskom jeziku tačan je onaj izraz koji ima vrednost različitu od nula, pre svega 1,
- Netačan je onaj izraz koji ima vrednost nula.
- 3==4 je celobrojni relacioni izraz čija vrednost je netačno.
- Vrednosti tačno i netačno zahtevaju uvođenje posebnog logičkog tipa (_Bool). (**z9.c**)

Označavanje vrednosti i operacija realnog tipa

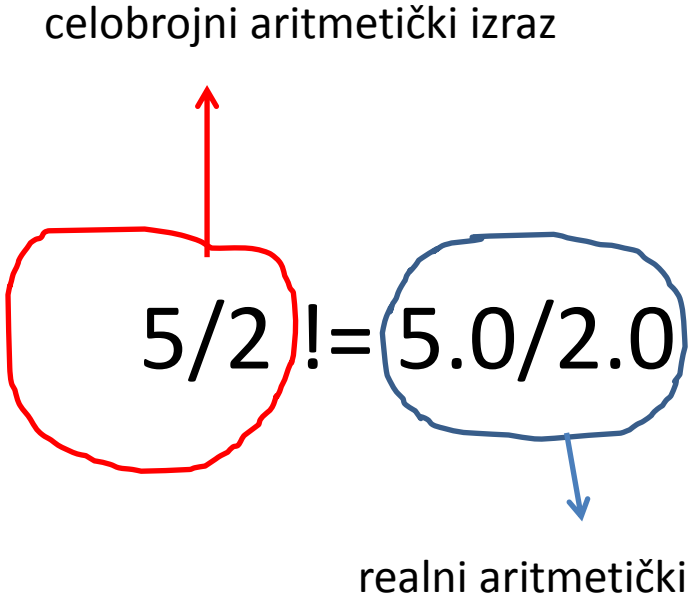
- Realni brojevi (float) se sastoje od celog i razlomljenog dela.
- Za celi deo realnog se koristi poziciona predstava, uvedena za cele brojeve.
- Razlomljeni deo realnog broja se označava nizom cifara, pri čemu svaka cifra predstavlja količnik njoj odgovarajućeg broja i onog stepena broja deset koji odgovara njenoj poziciji (s leva u desno, krajnje leva pozicija odgovara prvom stepenu).

Operacije realnog tipa

- Za realne brojeve su defnisane aritmetičke operacije sabiranje, oduzimanje, množenje i delenje.
- Značenje prva tri operatora (+, -, *) je nepromenjeno.
- Za delenje (/) realnih brojeva važi da je rezultat realan broj.

Konverzija tipova (**cast**)

- Operacija realnog delenja je predviđena samo za realne brojeve i nije definisana za cele brojeve.
- Da bi podelili dva cela broja, pre delenja ih pretvaramo u realne brojeve. Ovo se radi, na primer, pri izračunavanju srednje prolazne ocene u okviru pravljenja pregleda ispitnih rezultata.
- Posledica podrazumevajuće konverzije celog tipa u realni je da celi brojevi mogu da budu operandi operatora /.



Vrednost prvog izraza 2 se prvo pretvori u realan broj 2. i tek nakon toga se poredi sa vrednošću drugog izraza 2.5 da bi se ustanovilo da je vrednost prethodnog relacionog izraza **tačno**.

Međusobni odnos vrednosti i operacijā raznih tipova

- Operatori sa samo jednim operandom se nazivaju **unarni**, a oni sa dva operanda se nazivaju **binarni** operatori.
- Kombinovanjem malih zagrada, raznih operatora i vrednosti raznih tipova nastaju kompleksni izrazi.

Označavanje vrednosti i operacija znakovnog tipa

- Za predstavljanje pojedinačnog znaka koristi se tip **char**.
- Niz znakova zovemo **string**.
- Na primer, izraz 525-864 ako su mu elementi znakovi ne predstavlja aritmetički izraz nego string.