Spremenljivke in izrazi

Spremenljivka je količina, katere vrednost se med izvajanjem programa lahko spreminja. Določena je s tipom, imenom in vrednostjo. Izmed osnovnih tipov spremenljivk bomo uporabljali cela števila (`int`) in realna števila (`double`).

Tip spremenljivke moramo določiti pred njeno prvo uporabo. Največkrat je to na začetku funkcije ali kakšnega drugega bloka, lahko pa je tudi vmes med drugimi stavki, kjer spremenljivko prvič uporabimo. Stavku, s katerim določimo tip spremenljivke, rečemo *deklaracija*, sestavljen pa je iz oznake tipa (v naših programih bo to `int` ali `double`) in imena spremenljivke, ki si ga sami izberemo. Ime spremenljivke je lahko sestavljeno iz več znakov, a naj ne bo predolgo. Koristno je izbirati takšna imena, ki nam povedo, čemu je spremenljivka namenjena.

```
int visina;
double polmer;
```

Spremenljivki dodelimo (ali spremenimo) vrednost s *prireditvenim stavkom*. Uporabimo prireditveni operator (največkrat je to enačaj), na njegovi levi strani napišemo ime spremenljivke, na desni pa vrednost, ki jo želimo dodeliti tej spremenljivki. Seveda mora biti vrednost ustreznega tipa (realne vrednosti ne moremo shraniti v celoštevilsko spremenljivko).

```
visina = 753;
polmer = 6.249;
```

Deklaracijo in prireditev lahko zapišemo skupaj z enim samim stavkom, ki mu rečemo *definicija*:

```
int visina = 753;
double polmer = 6.249;
```

V prireditvenih stavkih in definicijah lahko namesto vrednosti na desni strani prireditvenega operatorja napišemo poljuben matematični izraz, katerega vrednost ustreza tipu spremenljivke. V takih izrazih lahko uporabljamo vrednosti, operatorje, oklepaje, klice funkcij ...

```
double obseg = 2 * 3.14 * polmer;
```

Operatorji

V programu lahko uporabljamo vse standardne operatorje (`+`, `-`, `*`, `/`). Posebej je treba opozoriti na posebnost deljenja, kadar sta oba operanda celoštevilska. Za rezultat v takem primeru dobimo celi del kvocienta. Če pa je vsaj eden od operandov realna vrednost, pa za rezultat dobimo pravilen realni kvocient.

Poleg štirih osnovnih opratorjev pozna C tudi operator `%` (ostanek pri celoštevilskem deljenju).

izraz	vrednost
`6 + 8`	`14`
`13 - 7`	`6`
`3 * 5`	`15`
`17 / 5`	`3`
`17.0 / 5`	`3.4`
`17 / 5.0`	`3.4`
`17.0 / 5.0`	`3.4`
`17 % 5`	`2`

Včasih pride prav tudi pretvorba cele vrednosti v realno in obratno. To storimo tako, da pred imenom spremenljivke ali pred izrazom (prej izraz zapišemo v oklepajih) napišemo ime tipa v oklepajih. Pretvorbo iz cele v realno vrednost potrebujemo predvsem pri deljenju dveh celih števil, kadar potrebujemo realen rezultat. Če je vsaj eden od operandov konstanta, lahko samo dopišemo decimalno piko in decimalko 0 (kot v zgornji tabeli), če pa sta oba operanda spremenljivki ali izraza, pa nam preostane samo še ta način pretvorbe. Pri pretvorbi iz realne v celo vrednost izgubimo vse decimalke (ostane samo celi del).

```
double x = 4.72;
int a = (int)x;             // dobimo 4
int b = (int)(10 * x);      // dobimo 47
double y = (double)b / a;   // dobimo 11.75
```

Matematične funkcije

Prej ko slej bomo potrebovali tudi kakšno od znanih matematičnih funkcij, kot so trigonometrične, kvadratni koren, potenciranje, absolutna vrednost in podobne. te funkcije so zbrane v posebni knjižnici, ki jo vključimo na začetku programa z ukazom:

```
#include <math.h>
```

V spodnji tabeli so zbrane najpomembnejše matematične funkcije. Pri trigonometričnih funkcijah morajo biti vsi koti podani v radianih.

izraz	pomen
`sin(x)`	sinus kota `x`
`cos(x)`	kosinus kota `x`
`tan(x)`	tangens kota `x`
`asin(x)`	arcus sinus od `x`
`acos(x)`	arcus kosinus od `x`
`atan(x)`	arcus tangens od `x`
`atan2(y, x)`	arcus tangens od `y/x`
`sqrt(x)`	kvadratni koren iz `x`
`pow(x, y)`	`x` na potenco `y`
`exp(x)`	e na potenco `x`
`log(x)`	naravni logaritem od `x`
`log10(x)`	logaritem pri osnovi 10 od `x`
`fmod(x, y)`	realni ostanek pri deljenju `x/y`
`fabs(x)`	absolutna vrednost realnega števila `x`
`floor(x)`	največje celo število, ki ni večje od `x`
`ceil(x)`	najmanjše celo število, ki ni manjše od `x`

Funkcija, ki izračuna absolutno vrednost celega števila, ne spada med matematične funkcije. Najdemo jo v knjižnici `stdlib.h` pod imenom `abs`.

Izpisovanje

Podobno, kot smo izpisovali zaporedja znakov, lahko izpisujemo tudi vrednosti spremenljivk in izrazov. Uporabimo celo isto funkcijo, le da v nizu (ki mu bomo od zdaj naprej rekli formatni niz) s posebno kombinacijo znakov (formatnim določilom) označimo mesto, kjer bi želeli izpisati vrednost spremenljivke ali izraza, samo vrednost pa podamo kot naslednji parameter funkcije.

```
printf("Janez je star %d let.\n", starost);
```

```
printf("Obseg kroga je %g\n", obseg);
```

Za izpis celoštevilske vrednosti uporabljamo formatno določilo `%d`, za izpis realne vrednosti pa določila `%f`, `%e` in `%g`. Z določilom `%f` izpišemo vrednost v decimalni obliki, z določilom `%e` v eksponentni, pri uporabi določila `%g` pa bo program sam izbral način, ki se mu zdi najlepši (4.0 bo izpisal brez decimalnih ničel). Formatna določila lahko razširimo z različnimi dodatki, ki določajo širino, število decimalnih mest, poravnavo ...

Z enim samim stavkom lahko izpišemo tudi več vrednosti, paziti moramo samo, da navedemo toliko dodatnih parametrov, kolikor formatnih določil uporabimo.

```
printf("Rojen je %d.%d.%d.\n", dan, mesec, leto);
```

Branje

Ko določamo vrednost spremenljivke, lahko zahtevamo, da vrednost vnese uporabnik z uporabo tipkovnice. Tako program postane bolj uporaben, saj ga lahko poženemo večkrat zaporedamo in preiskusimo njegovo delovanje pri različnih vhodnih podatkih. Vnos vrednost v programu zahtevamo s funkcijo `scanf`. Za prvi parameter ji podamo formatni niz, ki določa kakšnega tipa mora biti vpisana vrednost, temu pa sledi ime spremenljivke, v katero bo program shranil vpisano vrednost. Pred imenom številske spremenljivke moramo napisati znak `&`, spremenljivko pa moramo pred tem deklarirati.

Pred vnosom vrednosti običajno izpišemo navodilo uporabniku, kaj naj vnese. Za branje celih vrednosti uporabljamo določilo `%d`, za branje realnih vrednosti pa določilo `%lf`.

```
int starost;
printf("Vnesi starost: ");
scanf("%d", &starost);

double velikost;
printf("Vnesi velikost v metrih: ");
scanf("%lf", &velikost);
```

Primer

Sestavimo program, ki bo dolžino iz yardov pretvoril v metre, decimetre in centimetre. En yard meri 0.9144 metra. Ker sta vhodna dolžina in razmerje med yardi in metri realni, bomo v računih uporabili realne spremenljivke. Končni rezultat bo seveda celoštevilski. Dobili ga bomo z rezanjem decimalk, morebitne milimetre, ki ostanejo, pa bomo zanemarili.

```
#include <stdio.h>

/* Pretvori dolžino iz yardov v m, dm in cm. */

int main(void)
{
   double yard = 0.9144;   // razmerje med yardi in metri

   double dolzinaY;
   printf("Vnesi dolzino v yardih: ");
   scanf("%lf", &dolzinaY);

   double dolzina = dolzinaY * yard;
   int m = (int)dolzina;

   dolzina = 10 * (dolzina - m);
   int dm = (int)dolzina;
   int cm = 10 * (dolzina - dm);

   printf("%g yardov = %d m %d dm %d cm\n", dolzinaY, m, dm, cm);
```

```
   return 0;
}
```

V spremenljivko `yard` smo zapisali, koliko metrov meri en yard, v spremenljivko `dolzinaY` pa dolžino v yardih, ki jo moramo pretvoriti v metre, decimetre in centimetre. Dolžino najprej iz yardov pretvorimo v metre. Dobimo realno število, ki ga shranimo v spremenljivko `dolzina`. Vrednost te spremenljivke pretvorimo v celo število (odrežemo decimalke) in tako dobimo metre (spremenljivka `m`). Če metre odštejemo od celotne dolžine, ostane nenegativno realno število manjše od 1, iz katerega moramo dobiti še decimetre in centimetre. Decimetre dobimo tako, da razliko pomnožimo z 10, shranimo nazaj v spremenljivko `dolzina`, nato pa odrežemo decimalke in dobljeno vrednost shranimo v spremenljivko `dm`. Na podoben način dobimo tudi centimetre.