Pete vaje iz računalništva I

Filip Demšar

08. 11. 1998

1. Konstante in parametri

1.1 Vrste konstant

V c-u glede na uporabo uporabljamo dve vrsti konstant in sicer takšne, ki so odvisne od programskega jezika c, in take, ki so odvisne od pred-prejavalnika, ki ga uporablja c.

1.2 Vrste parametrov glede na uporabo

Parametre lahko prenašamo v podprogram na dva načina in sicer *po vrednosti* (po uporabi nam podprogram ne vrne morebitne spremebe) in *po referenci* (sprememba parametra je na razpolago).

2. Popravljen izpis števk

Števke želimo izpisati z besedami, za kar uporabimo `switch case` stavek.

Nadalje, želimo števke izpisati najprej od leve proti desni in nato še od desne proti levi. Razlika med obema izpisoma je samo v zaporedju klicev funkcij v funkciji `izpisi_stevilo`. Pri prvem izpisu najprej kličemo `izpisi_stevko` in nato (rekurzivno) `izpisi_stevilo`, v drugem primeru pa najprej izpišemo preostalo število (`kolicnik`) ter na koncu zadnjo števko.

Na koncu smo izpisali še število tako z leve proti desni, kot z desne proti levi. Dodali smo še ukaze za prehod v novo vrsto: ko izpišemo najbolj levo števko v prvem izpisu in najbolj desno števko v drugem izpisu.

Lahko si zamislimo, da je najbolj leva števka 0, ki je ne izpišemo, ampak namesto nje prestopimo v novo vrsto (glej `if` stavek – ustavitveni pogoj rekurzije – v funkciji `izpisi_stevilo`. Drugi prehod v novo vrsto pa izpišemo, ko se vrnemo iz funkcije `izpisi_stevilo` v glavno funkcijo (program).

Natančen postopek si lahko ogledate na našem področju.

2.1 Konstante

V programu, ki smo ga pričeli pisati na prejšnjih vajah, dopolnjujemo in spreminjamo pa ga na današnjih, dodamo zanko, da se bo izvajal večkrat – dokler ne vpišemo neke v naprej določene vrednosti. Če to vrednost spreminjamo, jo moramo spremeniti tako tam, kjer sprašujemo uporabnika programa po številu s katerim končamo izvajanje programa, kot tudi v glavnem programu, kjer programu povemo, kdaj naj preneha s svojim izvajanjem. Veliko lažje to storimo, če že na začetku programa določimo vrednost, s katero izstopimo iz programa To storimo z *definicijo konstante* :

```
const int izhodna_vrstica = 17 ;
```

Pri popravljanju besedila programa moramo vedno biti pozorni na preglednost.

3. Izris kvadrata ustreznih dimenzij

3.1 Predpriprava

Na splošno je program sestavljen iz *kode* in iz *komentarjev.* Komentarji so lahko *opisni* ali *programski* (pomagajo pri boljši čitljivosti). Konkretno ta program je sestavljen iz *glavnega programa* (ki nam pokliče funkcijo, katera prebere velikost kvadrata in nato kliče podprogram, ki bo ta kvadrat izrisal) ter *podprogramov* (le ti so po potrebi zopet razdeljeni na podprograme, ki opravijo točno določeno nalogo).

3.2 Izgled napisanega programa z nekaj opombami

Program, ki ga uporabljamo za risanje kvadrata z ustrezno dolžino stranice izgleda takole :

```
#include<stdio.h>

int preberi_velikost()
{
  int dolzina;

  printf("Vpisi dolzino stranice kvadrata:");
  scanf("%d", &dolzina);
  return dolzina;

} /* preberi_velikost */

void izrisi kvadrat(int sirina)
{

  int i;

  izrisi_osnovnico(sirina);
  for (i = 1; i <= sirina - 2; i++)
    izrisi_vmesnico(sirina);
  izrisi_osnovnico(sirina);

} /* izrisi_kvadrat */

   int main(void)
  {
```

```
    int sirina;

    sirina = preberi_velikost();

    izrisi_kvadrat(sirina);
    return 0;

} /* main */
```

Pri pisanju kateregakoli programa moramo vedno zelo natančno določiti, kaj hočemo od uporabnika. V tem primeru, kakšna naj bo dolžina stranice kvadrata. Podprogram `preberi_velikost` prebere dolžino stranice, jo shrani v spremenljivko ter nam jo vrne po klicu (vrne jo podprogramu `main`). Podprogram `main` nato pokliče podprogram, ki kvadrat z ustrezno dolžino stranice tudi nariše.

Ker se bodo pri risanju kvadrata nekatere vrstice ponavljale in sicer odvisno od dolžine stranice, bomo pri pisanju programa `izrisi_kvadrat` uporabili *for zanko*. Ta zanka izgleda takole:

```
  for(_A_ , _B_ , _C_ )
  {
    stavek
  }
```

Pri `for` zanki se najprej izvede ukaz `_A_`, nato program pogleda pogoj `_B_`, izvede `stavek`, gre na `_C_` in zopet preveri pogoj `_B_` ter to izvaja toliko časa, dokler je pogoj `_B_` izpolnjen.

Za domačo nalogo moramo definirati funkciji, ki bosta narisali *osnovnico* in *vmesnico*, ter ju vgraditi v program.

4. Uporaba ukaza Help v Turbo C-u

Do datoteke Help pridemo z pritiskom na gumba *alt* in *h*, s čimer pridemo do grafičnega vmesnika, po katerem se sprehajamo s pomočjo tabulatorja. Če nas zanima točno določeno področje, se postavimo na *topic search*, vtipkamo prve črke izbranega področja in ga izberemo med na te črke izbranimi podpodročji. Na vajah smo si v knjižnjici *podprogrami oz. rutine* ogledali *grafične rutine*: *arc, circle, ...*