Primjeri zadataka

Doc. dr. sc. Hrvoje Kalinić

2016-10-26

Uvod

- Motivacija
- Preduvjeti
- Primjeri

不闻不若闻之

闻之不若见之

见之不若知之

知之不若行之

荀子

Čuti je bolje no slušati

Vidjeti je bolje no čuti

Znati je bolje no vidjeti

Raditi je bolje no znati

Xun Zi[1]

Učenje programskog jezika je kao i učenje bilo kojeg jezika - u početku je teško, a znamo da smo ga naučili tek kad ga tečno govorimo.

Svrha ovog dokumenta je sabiranje primjera na kojima možemo pokazati neke posebnosti C jezika i tako ga bolje *naučiti*.

Da bismo nešto *naučili* trebamo mnogo prakse[2] da bismo od pukog prepoznavanja došli do znanja i razlikovanja koji su preduvijeti mudrog rasuđivanja[3].

[1] Učitelj Xun, pravim imenom Xun Kuang bio je konfucijanski filozof iz 4. st. prije Krista.
[2] Dakle, rada.
[3] u smislu: "ispravnog odabira što od svojih znanja i sposobnosti upotrijebiti u danom trenutku".

Preduvjeti

Za izvršavanje programa napisanih u C jeziku potrebno je prevođenje programa. Prevoditelj[4] s kojim ćemo se služiti je gcc. Instalacija gcc-a na Linuxu je jednostavna[5] kao upisati:

```
apt-get install gcc
```

Ako ste na nekom drugom operacijskom sustavu i ne želite koristiti komercijalne proizvode možete pogledati MinGW i Cygwin ili Code::Blocks ukoliko želite objedinjenu/integrirarnu razvojnu okolinu.

Primjeri - osnove

- Primjeri za učenje osnova/posebnosti C jezika
  - "Zdravo svijete!"
  - ...

"Zdravo svijete!"

Napišimo program koji svijetu javlja da radi pozdravom: "Zdravo svijete!"

Upotrijebit ćemo:

- Direktivu koja uključuje biblioteku za standardni ulaz i izlaz[6]
- Glavnu funkciju[7]
- Naputak koji poziva funkciju za ispis teksta[8]
- Naputak kojim funkcija završava[9]

Funkcija:

```
## #include <stdio.h>
## int main(void){
##     printf("Zdravo svijete!\n");
##     return 0;
##     }
```

Nakon prevođenja i izvršavanja programa kao rezultat dobijemo:

```
## >> Zdravo svijete!
```

[4] eng. compiler, jezični prevoditelj, dok starije generacije koriste i izraz "jezični procesor".
[5] Za distribucije koje nisu temeljene na apt-u koristi se yum, pacman, emerge ili ekvivalentna naredba
[6] `stdio.h`
[7] `int main()`
[8] `printf()`
[9] `return`

Volumen kutije

Napišimo program koji računa volumen kutije dimenzije 7x15x12 cm.

Program je sličan prethonom, ali nešto složeniji. Također, obraćamo malo više pozornosti na rezervatore koji ne moraju biti odvojeni od ostalih znakova[10].

```
## /* Program računa volumen kutije dimenzije 7x15x12 cm */
##
## #include <stdio.h>
##
## int main(void){
##     int duljina, visina, sirina, volumen;
##
##     duljina = 7;
##     visina = 15;
##     sirina = 12;
##     volumen = duljina * visina * sirina;
##
##     printf("Dimezije kutije: %dx%dx%d \n", duljina, visina, sirina);
##     printf("Volumen (u kubnim centimetrima): %d\n", volumen);
##
##     return 0;
## }
```

Rezultat izvođenja programa:

```
## >> Dimezije kutije: 7x15x12
## >> Volumen (u kubnim centimetrima): 1260
```

Što ako želimo prethodni program izmijeniti na način da može računati volumen kutije (ili čak zgrade) proizvoljnih dimenzija, uz pretpostavku da su svi objekti kvadratnog oblika?

Da bismo to mogli načiniti potrebno je upamtiti:

- kako upisujemo tekst u program[11]
  - to ćemo trebati napraviti za svaku dimenziju kvadra

```
## /* Program računa volumen kutije za dimenzije dane od korisnika programa */
##
## #include <stdio.h>
##
## int main(void){
```

[10] U našem slučaju to je znak x koji simbolizira "puta"

[11] riječ je o naputku koji poziva funkciju `scanf()` iz biblioteke `stdio.h`

```
##     int duljina, visina, sirina, volumen;
##
##     printf("Učitaj duljinu kutije: \n");
##     scanf("%d", &duljina);
##
##     printf("Učitaj visinu kutije: \n");
##     scanf("%d", &visina);
##
##     printf("Učitaj širinu kutije: \n");
##     scanf("%d", &sirina);
##
##     volumen = duljina * visina * sirina;
##
##     printf("Dimezije kutije: %dx%dx%d\n", duljina, visina, sirina);
##     printf("Volumen (u kubnim centimetrima): %d\n", volumen);
##
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 2
## << 3
## << 4
```

Dok će program ispisivati:

```
## >> Učitaj duljinu kutije:
## >> Učitaj visinu kutije:
## >> Učitaj širinu kutije:
## >> Dimezije kutije: 2x3x4
## >> Volumen (u kubnim centimetrima): 24
```

Pretvorba Farenheitovih stupnjeva u Celzijuseve

Napišimo program koji pretvara Farenheitovih stupnjeva u Celzijuseve. Riječ je o programu koji implementira jednostavnu matematičku formulu:

$$T_C = (T_F - T_{F0}) * \frac{5}{9}$$

Gdje treba znati da je $T_{F0} = 32$ stupnja.

```
## #include <stdio.h>
##
##
##
```

```
##
## int main(void){
##     float TC, TF;
##
##     printf("Unesi temepraturu u Fahrenheitovim stupnjevima: \n");
##     scanf("%f", &TF);
##
##     TC = (TF - 32.0f) * (5.0f / 9.0f);
##
##     printf("Temperatura izražena u Celsiusevim stupnjevima iznosi: %.1f\n", TC);
##
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 85
```

Dok će program ispisivati:

```
## >> Unesi temepraturu u Fahrenheitovim stupnjevima:
## >> Temperatura izražena u Celsiusevim stupnjevima iznosi: 29.4
```

Isti zadatak možemo riješiti na način da zadamo *makro konstante* kroz direktive i na taj način izbjegnemo "hardkodiranje" konstanti.

To je posebno zgodno ako:

- ima više mijesta na kojima koristimo istu konstantu
- ili očekujemo da ćemo konstatne iz nekog razloga mijenjanti

```
## #include <stdio.h>
##
## #define TF0 32.0f
## #define FAKTOR (5.0f / 9.0f)
##
## int main(void){
##     float TC, TF;
##
##     printf("Unesi temepraturu u Fahrenheitovim stupnjevima: \n");
##     scanf("%f", &TF);
##
##     TC = (TF - TF0) * FAKTOR;
##
##     printf("Temperatura izražena u Celsiusevim stupnjevima iznosi: %.1f\n", TC);
```

```
##
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 85
```

Dok će program ispisivati:

```
## >> Unesi temepraturu u Fahrenheitovim stupnjevima:
## >> Temperatura izražena u Celsiusevim stupnjevima iznosi: 29.4
```

Pretvorba Rankineovih stupnjeva u Celzijeve

Napišimo program koji pretvara Rankineove stupnjeve u Celzijuseve. Riječ je o programu koji implementira jednostavnu matematičku formulu:

$$T_C = (T_R - T_{R0}) * \frac{5}{9}$$

Gdje treba znati da je $T_{R0} = 491.67$ stupnja.

- Primijetite sličnost s pretvorbom u Farenheitove stupnjeve
- Sličnost sam uočio (a i lijen sam) pa ću samo prilagoditi postojeći program
  - Dovoljeno je promijeniti konstantu `TF0`, no promijeniti ćemo i dio teksta u funkciji `printf()`...

```
## #include <stdio.h>
##
## #define TF0 491.67f
## #define FAKTOR (5.0f / 9.0f)
##
## int main(void){
##     float TC, TF;
##
##     printf("Unesi temperaturu u Rankineovim stupnjevima: \n");
##     scanf("%f", &TF);
##
##     TC = (TF - TF0) * FAKTOR;
##
##     printf("Temperatura izražena u Celsiusevim stupnjevima iznosi: %.1f\n", TC);
##
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 85
```

Dok će program ispisivati:

```
## >> Unesi temperaturu u Rankineovim stupnjevima:
## >> Temperatura izražena u Celsiusevim stupnjevima iznosi: -225.9
```

Primjeri s `printf()` funkcijom

Promotrit ćemo kako se ponaša `printf()` funkcija u slučaju nekih neispravnih pozivanja:

- Korištenja neispravnog rezervatora
- Predavanje neispravnog broja parametara

```
## #include <stdio.h>
##
## int main(void){
##     int i,j;
##     float x, y;
##
##     scanf("%d%d%f%f", &i, &j, &x, &y);
##     printf("cijeli brojevi: i = %d, j = %d\n", i, j);
##     printf("decimalni brojevi: x = %f, y = %f\n", x, y);
##
##     printf("Primjer 1: Ispisujem koritećи neispravan rezervator (int kao float):\n");
##     printf("i = %f\n", i);
##     printf("Primjer 2: Ispisujem koritećи neispravan rezervator (float kao int):\n");
##     printf("x = %d\n", x);
##     printf("Primjer 3: Ispisujem s dva rezervatora, no predajem jedan broj:\n");
##     printf("i = %d, j = %d\n", i);
##     printf("Primjer 4: Ispisujem s jednim rezervatorom, a predajem dva broja:\n");
##     printf("i = %f\n", i, x);
##     printf("Primjer 5: Brojeve predajem neispravnim redoslijedom u ispis:\n");
##     printf("i = %d, x = %f\n", x, i);
##
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 117
## << 55
## << 73.1534
## << 2015.0
```

Dok će program ispisivati:

```
## >> cijeli brojevi: i = 117, j = 55
## >> decimalni brojevi: x = 73.153397, y = 2015.000000
## >> Primjer 1: Ispisujem koriteći neispravan rezervator (int kao float):
## >> i = 73.153381
## >> Primjer 2: Ispisujem koriteći neispravan rezervator (float kao int):
## >> x = 4196421
## >> Primjer 3: Ispisujem s dva rezervatora, no predajem jedan broj:
## >> i = 117, j = -1633187936
## >> Primjer 4: Ispisujem s jednim rezervatorom, a predajem dva broja:
## >> i = 73.153397
## >> Primjer 5: Brojeve predajem neispravnim redoslijedom u ispis:
## >> i = 117, x = 73.153397
```

Ponovljenim izvršavanjem programa s istim ulaznim podatcima možemo uočiti:

- u *Primjeru 3* se kao drugi broj (j) pojavljuje neki slučajno generirani broj
- u *Primjeru 2* ispisani broj (x) ostaje uvijek isti neispravni broj

```
## >> cijeli brojevi: i = 117, j = 55
## >> decimalni brojevi: x = 73.153397, y = 2015.000000
## >> Primjer 1: Ispisujem koriteći neispravan rezervator (int kao float):
## >> i = 73.153381
## >> Primjer 2: Ispisujem koriteći neispravan rezervator (float kao int):
## >> x = 4196421
## >> Primjer 3: Ispisujem s dva rezervatora, no predajem jedan broj:
## >> i = 117, j = -939571296
## >> Primjer 4: Ispisujem s jednim rezervatorom, a predajem dva broja:
## >> i = 73.153397
## >> Primjer 5: Brojeve predajem neispravnim redoslijedom u ispis:
## >> i = 117, x = 73.153397
```

Primjer nedefiniranog ponašanja

kao primjer nedefiniranog ponašanje može poslužiti kratki program:

```
## #include <stdio.h>
## int main(void){
##     int a = 3, b, c;
##     c = (b = a + 1 ) - (a = 5);
##     printf("c = %d\n", c);
##     return 0;
## }
```

ili

```
## #include <stdio.h>
## int main(void){
##     int i=1;
##     i = i++;
##     printf("i = %d\n", i);
##     return 0;
## }
```

Koji ukoliko su prevedeni s `gcc` prevoditeljem vraćaju pogrešku oblika...

```
warning: operation on ... may be undefined
```

...no samo ukoliko su su pozvani sa zastavicom `-Wsequence-point`[12] ili ukoliko je više zastavica ("sve") za upozorenja podignuto zastavicom `-Wall`.

Rezultat izvršavanja programa[13] je:

```
## >> c = -1
```

odnosno:

```
## >> i = 1
```

I oboje predstavljaju (donekle) smisleno izvršavanje programa, no treba ih izbjegavati jer je rezultat operacija *nedefiniran*.

Odbrojavanje

Napišimo program koji za zadani broj sekundi odbrojava sekunde do 0.

Zadatak ćemo rješiti:

- `while` petljom
- `do ... while` petljom
- `for` petljom *(x2)*

Odbrojavanje `while` petljom

```
## #include <stdio.h>
## int main(void){
##     int i, n;
##
##     printf("Unesite sekunde od kojih odbrojavamao: \n");
```

[12] koja detektira neke jednostavnije slučajeve nedefiniranog ponašanja.
[13] Nakon prevođenja s `gcc` prevoditeljem.

```
##     scanf("%d", &n);
##     i = n;
##     while (i>0) {
##         printf("Još %d sekundi preostalo...\n", i--);
##     }
##
##     return 0;
## }
```

Odbrojavanje `do ... while` petljom

```
## #include <stdio.h>
## int main(void){
##     int i, n;
##
##     printf("Unesite sekunde od kojih odbrojavamao: \n");
##     scanf("%d", &n);
##     i = n;
##     do {
##         printf("Još %d sekundi preostalo...\n", i--);
##     } while (i>0);
##
##     return 0;
## }
```

Odbrojavanje `for` petljom (1)

```
## #include <stdio.h>
## int main(void){
##     int i, n;
##
##     printf("Unesite sekunde od kojih odbrojavamao: \n");
##     scanf("%d", &n);
##     for (i = n; i > 0; i--)
##         printf("Još %d sekundi preostalo...\n", i);
##
##     return 0;
## }
```

Rezultat odbrojavanja

Svi programi se izvršavaju gotovo jednako, bez obzira koja je petlja korištena. Tako za ulaz

```
## << 10
```

... na izlazu dobijemo:

```
## >> Unesite sekunde od kojih odbrojavamao:
## >> Još 10 sekundi preostalo...
## >> Još 9 sekundi preostalo...
## >> Još 8 sekundi preostalo...
```

```
## >> Još 7 sekundi preostalo...
## >> Još 6 sekundi preostalo...
## >> Još 5 sekundi preostalo...
## >> Još 4 sekundi preostalo...
## >> Još 3 sekundi preostalo...
## >> Još 2 sekundi preostalo...
## >> Još 1 sekundi preostalo...
```

Rezultat odbrojavanja (2)

Značajna razlika se dobije ukoliko na ulaz dovedemo

```
## << 0
```

Tada u `for` petlji na izlazu imamo:

```
## >> Unesite sekunde od kojih odbrojavamao:
```

U `while` petlji na izlazu imamo:

```
## >> Unesite sekunde od kojih odbrojavamao:
```

Dok `do ... while` petlja daje:

```
## >> Unesite sekunde od kojih odbrojavamao:
## >> Još 0 sekundi preostalo...
```

Odbrojavanje `for` petljom (2)

Ukoliko nas užasno živcira što se računalo ne zna izražavati na gramatički pravilnom hrvatskom jeziku - naučimo ga:

- 1 sekunda, a ne 1 sekundi
- 2, 3 ili 4 sekunde, a ne 2, 3 ili 4 sekundi
- ...

```
## #include <stdio.h>
## int main(void){
##     int i, n;
##
##     printf("Unesite sekunde od kojih odbrojavamao: \n");
##     scanf("%d", &n);
##     for (i = n; i > 0; i--){
##         switch(i){
##             case 1: printf("Još %2d sekunda preostala...\n", i); break;
##             case 2:
##             case 3:
```

```
##             case 4: printf("Još %2d sekunde preostale...\n", i); break;
##             default: printf("Još %2d sekundi preostalo...\n", i);
##         }
##     }
##     return 0;
## }
```

Rezultat odbrojavanja

Svi programi se izvršavaju gotovo jednako, bez obzira koja je petlja korištena. Tako za ulaz

```
## << 10
```

... na izlazu dobijemo:

```
## >> Unesite sekunde od kojih odbrojavamao:
## >> Još 10 sekundi preostalo...
## >> Još  9 sekundi preostalo...
## >> Još  8 sekundi preostalo...
## >> Još  7 sekundi preostalo...
## >> Još  6 sekundi preostalo...
## >> Još  5 sekundi preostalo...
## >> Još  4 sekunde preostale...
## >> Još  3 sekunde preostale...
## >> Još  2 sekunde preostale...
## >> Još  1 sekunda preostala...
```

Primjer korištenja naputka `break`

Napišimo program koji provjerava je li broj N prost koristeći se jednostavnom logikom:

- tražimo djelitelj od N vrteći se u petlji od 1 do N
- radi efikasnoti[14] iz petlje iskačemo čim pronađemo prvi djelitelj

```
## #include <stdio.h>
## int main(void){
##     int broj, djelitelj;
##
##     printf("*** Program provjerava je li broj prost ***\n");
##     printf("Unesi broj: \n");
##     scanf("%d", &broj);
##
##     for (djelitelj = 2; djelitelj < broj; djelitelj++){
##         if (broj % djelitelj == 0)
##             break;
##     }
##     if (djelitelj < broj)
```

[14] Ovo je efikasno za brojeve koji nisu prosti. Za velike proste brojeve ovaj kod je još uvijek daleko od efikasnog.

```
##         printf("%d je djeljiv s %d\n", broj, djelitelj);
##     else
##         printf("%d je prost broj\n", broj);
##
##     return 0;
## }
```

Ako kao ulaz zadamo vrijednost:

```
## << 131
```

Program će ispisivati:

```
## >> *** Program provjerava je li broj prost ***
## >> Unesi broj:
## >> 131 je prost broj
```

Primjer korištenja naputka `break` (2)

Napišimo program koji u petlji čita korisnički unos i računa kvadrat unesenog broja, dok god se ne učita 0.

- rješenje koristeći se beskonačnom petljom i `break` naputkom
- alternativno rješenje izbjegavajući `break`
- također primijetite razliku među `while` i `do...while` petljama

```
## #include <stdio.h>
## int main(void){
##     int n;
##
##     for (;;){ // ili while(1)
##         printf("Učitaj cijeli broj (ili 0 za prekid): \n");
##         scanf("%d", &n);
##         if (n == 0)
##             break;
##         printf("%d na kvadrat iznosi %d\n", n, n*n);
##     }
##     return 0;
## }
```

Ako kao ulaz zadamo vrijednost:

```
## << 7
## << 12
## << 3
## << 495
## << 0
```

Program će ispisivati:

```
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 7 na kvadrat iznosi 49
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 12 na kvadrat iznosi 144
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 3 na kvadrat iznosi 9
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 495 na kvadrat iznosi 245025
## >> Učitaj cijeli broj (ili 0 za prekid):
```

Alternativno rješenje

Zadatak je često moguće rješiti ne koristeći `break` naputak. Tako smo mogli računati kvadrate unesenih brojeva *dok god se ne učita 0* koristeći `while` petlju koja nije beskonačna:

```
## #include <stdio.h>
## int main(void){
##     int n;
##
##     printf("Učitaj cijeli broj (ili 0 za prekid): \n");
##     scanf("%d", &n);
##     while(n != 0){
##         printf("%d na kvadrat iznosi %d\n", n, n*n);
##         printf("Učitaj cijeli broj (ili 0 za prekid): \n");
##         scanf("%d", &n);
##     }
##     return 0;
## }
```

No, nasažetije rješenje je rješenje koje koristi `do...while` petlju:

```
## #include <stdio.h>
## int main(void){
##     int n;
##
##     do{
##         printf("Učitaj cijeli broj (ili 0 za prekid): \n");
##         scanf("%d", &n);
##         printf("%d na kvadrat iznosi %d\n", n, n*n);
##     }while(n != 0);
##     return 0;
## }
```

Za zadani ulaz:

```
## << 7
## << 12
## << 3
## << 495
## << 0
```

... program koji se koristi `while` petljom će ispisati:

```
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 7 na kvadrat iznosi 49
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 12 na kvadrat iznosi 144
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 3 na kvadrat iznosi 9
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 495 na kvadrat iznosi 245025
## >> Učitaj cijeli broj (ili 0 za prekid):
```

... dok će program koji se koristi `do...while` ispisati[15]:

```
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 7 na kvadrat iznosi 49
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 12 na kvadrat iznosi 144
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 3 na kvadrat iznosi 9
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 495 na kvadrat iznosi 245025
## >> Učitaj cijeli broj (ili 0 za prekid):
## >> 0 na kvadrat iznosi 0
```

Primjeri - složeniji

- Primjeri za poboljšanje programerski vještina u C jeziku
  - Čitanje znamenki i sl.
  - ...

Suma i umnožak znamenki

Napišimo program koji za zadani dvoznamenkasti proj računa sumu i umnožak njegovih znamenki.

[15] Primijetie kako smo izbjegli dupliciranje koda u izvornom kodu programa, a da na izlaz imamo ispisivanje kvadrata i posljednjeg unešenog broja (nule).

Studenti tipično rješavaju ovaj zadatak koristeći modulo % operator:

```
## #include<stdio.h>
## int main(void){
##     int broj, z1, z2, suma, umnozak;
##     printf("Upišite dvoznamenkasti broj: \n");
##     scanf("%d", &broj);
##     z1 = broj / 10;
##     z2 = broj % 10;
##     suma = z1 + z2;
##     umnozak = z1 * z2;
##     printf("Suma znamenki iznosi: %d\n", suma);
##     printf("Umnožak znamenki iznosi %d\n", umnozak);
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 25
```

Dok će program ispisivati:

```
## >> Upišite dvoznamenkasti broj:
## >> Suma znamenki iznosi: 7
## >> Umnožak znamenki iznosi 10
```

Ipak, zadatak se može jednostavnije riješiti, izbjegavajući nepotrebne operacije, a koristeći rezervatore:

```
## #include<stdio.h>
## int main(void){
##     int z1, z2, suma, umnozak;
##     printf("Upišite dvoznamenkasti broj: \n");
##     scanf("%1d%1d", &z1, &z2);
##     suma = z1 + z2;
##     umnozak = z1 * z2;
##     printf("Suma znamenki iznosi: %d\n", suma);
##     printf("Umnožak znamenki iznosi %d\n", umnozak);
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 25
```

Dok će program ispisivati:

```
## >> Upišite dvoznamenkasti broj:
## >> Suma znamenki iznosi: 7
## >> Umnožak znamenki iznosi 10
```

Brojanje znamenki

Napišimo program koji broji od koliko se znamenki sastoji neki prirodni broj (pozitivan cijeli broj).

- Primijetimo, ovakav zadatak ne možemo riješiti koristeći se rezervatorima, već moramo korisiti modulo operator i petlju (`do .. while`)

```
## #include <stdio.h>
##
## int main(void){
##     int br_znamenki = 0, broj, tmp;
##
##     printf("Upišite neki prirodni broj: \n");
##     scanf("%d", &broj);
##
##     tmp = broj;
##     do {
##         tmp /= 10;
##         br_znamenki++;
##     } while (tmp > 0);
##
##     printf("Broj %d ima %d znamenki.", broj, br_znamenki);
##
##     return 0;
## }
```

Ako kao ulaz zadamo vrijednost:

```
## << 123456
```

Program će ispisati:

```
## >> Upišite neki prirodni broj:
## >> Broj 123456 ima 6 znamenki.
```

Zbir rješenih zadataka

- "Složeniji" zadataci riješeni na predavanju

Provizija - demonstracija skretnice

Napišite program koji računa proviziju na temelju vrijednosti trgovanja i to na sljedeći način, ali tako da nikada ne bude manja od 50 kn:

- Vrijednost trgovanja manja od 2500kn provizija 10kn+4% vrijednosti
- Vrijednost trgovanja 2500-5000kn provizija 30kn+3% vrijednosti
- Vrijednost trgovanja 5000-10000kn provizija 50kn+1.1% vrijednosti
- Vrijednost trgovanja 10000-20000kn provizija 100kn+0.7% vrijednosti
- Vrijednost trgovanja veća od 20000kn provizija 300kn+0.1% vrijednosti

```
## #include<stdio.h>
## int main(void){
##     float provizija, vrijednost;
##     printf("Unestite vrijednost trgovanja: \n");
##     scanf("%f", &vrijednost);
##
##     if (vrijednost < 2500.00f)
##         provizija = 10 + 4.0*vrijednost/100;
##     else if (vrijednost < 5000.00f)
##         provizija = 30 + 3.0*vrijednost/100;
##     else if (vrijednost < 10000.00f)
##         provizija = 50 + 1.1*vrijednost/100;
##     else if (vrijednost < 20000.00f)
##         provizija = 100 + 0.7*vrijednost/100;
##     else if (vrijednost < 50000.00f)
##         provizija = 200 + 0.3*vrijednost/100;
##     else
##         provizija = 300 + 0.1*vrijednost/100;
##     if (provizija < 50.00f)
##         provizija = 50.00f;
##
##     printf("Provizija za %.2f kn iznosi %.2f kn\n", vrijednost, provizija);
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 25000.00
```

Dok će program ispisivati:

```
## >> Unestite vrijednost trgovanja:
## >> Provizija za 25000.00 kn iznosi 275.00 kn
```

Datum - demonstracija skretnice

Napišimo program koji za uneseni datum u obliku dd.mm.gggg. ispisuje datum u obliku datiranja dokumenata (npr. 31. kolovoza 2008.)

```
## #include<stdio.h>
##
## int main(void){
##     int dan, misec, godina;
##     printf("Unesi datum oblika dd.mm.gggg.: \n");
##     scanf("%d .%d .%d .", &dan, &misec, &godina);
##
##     printf("Datum: %d. ", dan);
##     switch (misec) {
##         case 1:  printf("sječnja");   break;
##         case 2:  printf("veljače");   break;
##         case 3:  printf("ožujka");    break;
##         case 4:  printf("travnja");   break;
##         case 5:  printf("svibnja");   break;
##         case 6:  printf("lipnja");    break;
##         case 7:  printf("srpnja");    break;
##         case 8:  printf("kolovoza");  break;
##         case 9:  printf("rujna");     break;
##         case 10: printf("listopada"); break;
##         case 11: printf("studenog");  break;
##         case 12: printf("prosinca");  break;
##     }
##     printf(", %d.\n", godina);
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 30.2.2011.
```

Dok će program ispisivati:

```
## >> Unesi datum oblika dd.mm.gggg.:
## >> Datum: 30. veljače, 2011.
```

Suma niza - demonstracija `while` petlje

Napišimo program koji računa sumu niza koji korisnik unosi broj po broj. Ukoliko se unese broj 0, ispisuje se suma do tada unesenih članova niza.

```
## /* Program koji računa sumu unesenog niza */
##
## #include<stdio.h>
##
## int main(void){
##     int n, sum = 0;
##
##     printf("*** Program koji računa sumu unesenog niza ***\n");
##     printf("Unesi cijeli broj (ili 0 za prekid): \n");
##     scanf("%d", &n);
##
##     while (n != 0){
##         sum += n;
##         scanf("%d", &n);
##     }
##     printf("Suma niza iznosi: %d", sum);
##     return 0;
## }
```

Ako kao ulaz zadamo vrijednost:

```
## << 26
## << 3
## << 45
## << 92
## << 17
## << 08
## << 0
```

Program će ispisivati:

```
## >> *** Program koji računa sumu unesenog niza ***
## >> Unesi cijeli broj (ili 0 za prekid):
## >> Suma niza iznosi: 191
```

Tablica kvadrata - demonstracija petlje

Napišite program koji ispisuje tablicu kvadrata za sve brojeve do unesenog broja

- Brojeve formatirajmo u tablicu na takav način da brojevi znamenki jedinica budu međusobno poravnane
- Za iscrtavanje tablice ćemo se poigrati znakovima – i |
- Zadatak smo na predavanju rješili `while` petljom, prikladniji je za `for` petlju

Tablica kvadrata `while` petljom

```
## #include <stdio.h>
```

```
##
## int main(void){
##     int i, x;
##
##     printf("Upiši broj do kojeg želite ispis tablice kvadrata: \n");
##     scanf("%d", &x);
##
##     printf("\t|     x|         x^2|\n");
##     printf("\t|------|------------|\n");
##     i = 1;
##     while (i <= x) {
##         printf("\t|%6d|%12d|\n", i, i * i);
##         i++;
##     }
##     printf("\t|------|------------|\n");
##     return 0;
## }
```

Tablica kvadrata `for` petljom

```
## #include <stdio.h>
##
## int main(void){
##     int i, x;
##
##     printf("Upiši broj do kojeg želite ispis tablice kvadrata: \n");
##     scanf("%d", &x);
##
##     printf("\t|     x|         x^2|\n");
##     printf("\t|------|------------|\n");
##     i = 1;
##     while (i <= x) {
##         printf("\t|%6d|%12d|\n", i, i * i);
##         i++;
##     }
##     printf("\t|------|------------|\n");
##     return 0;
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 14
```

Dok će program ispisivati:

```
## >> Upiši broj do kojeg želite ispis tablice kvadrata:
## >>   |     x|         x^2|
## >>   |------|------------|
## >>   |     1|           1|
## >>   |     2|           4|
```

```
## >>   |     3|           9|
## >>   |     4|          16|
## >>   |     5|          25|
## >>   |     6|          36|
## >>   |     7|          49|
## >>   |     8|          64|
## >>   |     9|          81|
## >>   |    10|         100|
## >>   |    11|         121|
## >>   |    12|         144|
## >>   |    13|         169|
## >>   |    14|         196|
## >>   |------|------------|
```

Bankovni račun - demonstracija izbjegavanja `goto`

Napišimo jednostavan program za rad s bankovnim računom. Opcije su:

- 0: Postavi stanje računa na 0
- 1: Povuci s računa iznos koji unese korisnik
- 2: Uplati na račun iznos koji unese korisnik
- 3: Ispiši stanje računa
- 4: Izađi iz programa

U zadatku "moramo" korisiti `goto` skok jer `break` ne može iskočiti iz dvije petlje.

Ideja rješenja sa skokom

```
## for (;;){
##     /* Ispis mogućih izbora */
##     /* Učitavanje izbora */
##     switch (naputak) {
##         case 0:  /* postavi knjižicu na 0 */;
##             break;
##         case 1:  /* upitaj za željeni iznos i iznos dodaj na račun */;
##             break;
##         case 2:  /* upitaj za željeni iznos i iznos oduzmi s računa */;
##             break;
##         case 3:  /* ispiši trenutno stanje računa */;
##             break;
##         case 4:
##             goto kraj;
##         default: /* upozori da je došlo do pogreške i ponovno ispiši izbore */;
##             break;
##     }
## }
## kraj: return 0;
```

Ideja rješenja bez skoka

```
## for (;;){
```

```
##     /* Ispis mogućih izbora */
##     /* Učitavanje izbora */
##     switch (naputak) {
##         case 0:  /* postavi knjižicu na 0 */;
##             break;
##         case 1:  /* upitaj za željeni iznos i iznos dodaj na račun */;
##             break;
##         case 2:  /* upitaj za željeni iznos i iznos oduzmi s računa */;
##             break;
##         case 3:  /* ispiši trenutno stanje računa */;
##             break;
##         case 4:
##             return 0;
##         default: /* upozori da je došlo do pogreške i ponovno ispiši izbore */;
##             break;
##     }
## }
```

Rješenje

Cjelovita rješenja

- Sa skokom
- Bez skoka

. . . radi preglednosti su dani samo u popratnom pdf dokumentu

< rez >

```
## #include<stdio.h>
## int main(void){
##     int izbor;
##     float stanje_racuna = 0.0f, iznos;
##
##     printf("*** Program za rad s bankovnim računom ***\n");
##     printf("Izbornik: 0 - postavi račun na 0, 1 - dodaj na račun, 2 - povuci s računa,\n");
##     printf("          3 - ispiši stanje računa, 4 - izađi iz programa\n\n");
##
##     for (;;){
##         printf("Odaberi: \n");
##         scanf("%d", &izbor);
##         switch (izbor) {
##             case 0:
##                 stanje_racuna = 0.0f;
##                 break;
##             case 1:
##                 printf("Unesite iznos koji želite staviti na račun: \n");
##                 scanf("%f", &iznos);
##                 stanje_racuna += iznos;
##                 break;
##             case 2:
##                 printf("Unesite iznos koji želite povući s računa: \n");
##                 scanf("%f", &iznos);
```

```
##                 stanje_racuna -= iznos;
##                 break;
##             case 3:
##                 printf("Trenutno stanje računa: %.2f\n", stanje_racuna);
##                 break;
##             case 4:
##                 return 0;
##             default:
##                 printf("Unesen nepoznat izbor...\n");
##                 printf("Izbornik: 0 - postavi račun n 0, 1 - dodaj na račun, 2 - povuci s računa,\n"
##                 printf("          3 - ispiši stanje računa, 4 - izađi iz programa\n\n");
##                 break;
##         }
##     }
## }

## #include<stdio.h>
## int main(void){
##     int izbor;
##     float stanje_racuna = 0.0f, iznos;
##
##     printf("*** Program za rad s bankovnim računom ***\n");
##     printf("Izbornik: 0 - postavi račun na 0, 1 - dodaj na račun, 2 - povuci s računa,\n");
##     printf("          3 - ispiši stanje računa, 4 - izađi iz programa\n\n");
##
##     for (;;){
##         printf("Odaberi: \n");
##         scanf("%d", &izbor);
##         switch (izbor) {
##             case 0:
##                 stanje_racuna = 0.0f;
##                 break;
##             case 1:
##                 printf("Unesite iznos koji želite staviti na račun: \n");
##                 scanf("%f", &iznos);
##                 stanje_racuna += iznos;
##                 break;
##             case 2:
##                 printf("Unesite iznos koji želite povući s računa: \n");
##                 scanf("%f", &iznos);
##                 stanje_racuna -= iznos;
##                 break;
##             case 3:
##                 printf("Trenutno stanje računa: %.2f\n", stanje_racuna);
##                 break;
##             case 4:
##                 return 0;
##             default:
##                 printf("Unesen nepoznat izbor...\n");
##                 printf("Izbornik: 0 - postavi račun n 0, 1 - dodaj na račun, 2 - povuci s računa,\n"
##                 printf("          3 - ispiši stanje računa, 4 - izađi iz programa\n\n");
##                 break;
##         }
##     }
```

```
## }
```

Kao ulaz ćemo zadati vrijednosti:

```
## << 1
## << 10000
## << 2
## << 5000
## << 1
## << 100
## << 3
## << 4
```

Dok će programi ispisivati:

```
## >> *** Program za rad s bankovnim računom ***
## >> Izbornik: 0 - postavi račun na 0, 1 - dodaj na račun, 2 - povuci s računa,
## >>           3 - ispiši stanje računa, 4 - izađi iz programa
## >>
## >> Odaberi:
## >> Unesite iznos koji želite staviti na račun:
## >> Odaberi:
## >> Unesite iznos koji želite povući s računa:
## >> Odaberi:
## >> Unesite iznos koji želite staviti na račun:
## >> Odaberi:
## >> Trenutno stanje računa: 5100.00
## >> Odaberi:
```

尽信书不如无书。

Bolje je ne imati ni jednu knjigu no vjerovati sve što u knjigama piše.