

Programiranje 1

11. predavanje

Saša Singer

`singer@math.hr`

`web.math.hr/~singer`

PMF – Matematički odsjek, Zagreb

Sadržaj predavanja

- Ulaz i izlaz podataka:
 - Funkcije `getchar` i `putchar`.
- Obrada znakova:
 - Funkcije iz `ctype.h`.
 - Primjeri obrade niza znakova.

Informacije — Praktični kolokvij — prvi krug

Praktični kolokvij (prvi krug) — kratki komentar:

- Rezultati su “pristojni” — oko 68.05% svih pridošlih,
- tj. prošlo vas je 181 od 266.
- Dakle, šansu za drugu rundu ima 85 studenata.

Prošlogodišnja prolaznost je bila 65.40% u prvom krugu.

U aplikaciji za zadaće imamo prijavljenih 283 studenta.

- 10 studenata se prijavilo za PK, ali nije izašlo,
- a 7 studenata je “nestalo”, tj. zaboravilo na PK!

Informacije — Praktični kolokvij

Drugi krug PK2 ide

- sljedeći tjedan = 11. tjedan nastave.

Treći krug PK3 ide

- za tri tjedna = 13. (zadnji) tjedan nastave.

Prijava za PK2 (zauzimanje termina) je

- do kraja ovog tjedna, preko aplikacije za zadaće!

Napomena: Nedolazak na bilo koji PK treba opravdati ispričnicom (nastavniku),

- inače gubite pravo na popravak!

Informacije — Praktični kolokvij — općenito

Rok za predaju **ispričnica** za **nedolazak** na PK i sl.:

- zadnje **predavanje prije** kraja semestra (praznika).
- Konkretno: **petak, 16. 12.**, do **14** sati.

Sve **nakon toga** rješava se **molbama** u referadi!

Informacije — Zadaće

Trenutna statistika za domaće zadaće:

| zadatak | broj rješenja |
|---------------------------|---------------|
| Brojevnii sustavi | 240 |
| Prikaz broja u računalu | 212 |
| Osnovni programi i petlje | 51 |
| Složenije petlje | 21 |
| Nizovi | 6 |

Nije dobro! Programske zadaće

🚫 ili ne rješavate, ili u tome niste uspješni.

U ovoj fazi, morali biste moći riješiti sve zadaće, osim zadnje.

🚫 Dakle, ogromna većina kasni dvije zadaće!

Informacije

Teme sljedećih predavanja su redom:

- “Polja (nizovi)” i “Operacije s nizovima podataka”,
- “Pretraživanje i sortiranje nizova”.

Predavanje “Ulaz i izlaz podataka” prebačeno je u **dodatak** ovom predavanju (dostupno na [webu](#)).

- To ćemo napraviti u **Prog2**, kod [stringova](#) i [datoteka](#).

Informacije — ulaz i izlaz

Na mom [webu](#), pod **dodatnim** materijalima za **Prog1** i **Prog2**, nalazi se tekst

- 📄 [in_out.pdf](#) (7 stranica, 59 kB),

koji sadrži **detaljan** opis funkcija

- 📄 za **formatirani ulaz** i **izlaz** podataka.

Najveći dio teksta govori o funkcijama **fprintf** i **fscanf**.

U imenima ovih funkcija,

- 📄 **prvo** slovo **f** dolazi od riječi “file” (datoteka), a

- 📄 **zadnje** slovo **f** dolazi od “formatted” (formatirani).

Ove funkcije, u principu, rade za **bilo koju** datoteku, a

- 📄 **izlazna** ili **ulazna** datoteka se **zadaje** kao **argument**.

Informacije — ulaz i izlaz (nastavak)

Funkcije `printf` i `scanf` (bez prvog slova `f`)

- rade na **standardnim** datotekama za **izlaz**, odnosno **ulaz**, i zato se datoteka **ne zadaje**. To je **jedina** razlika!

Veza između **osnovne** funkcije (s prvim slovom `f`) i funkcije za **standardnu** datoteku dana je na **kraju** opisa osnovne funkcije.

Osnova za tekst je

- **Dodatak B** iz knjige **KR2**.

Tamo je opis **svih** funkcija iz standardne **C** biblioteke.

Međutim, “prijevod” **nije** doslovan. Neki dijelovi su prošireni i

- popravljen je opis **fscanf** (original nije skroz korektan).

Lijepo molim, ako uočite “**tipfelere**” — **javite** mi!

Ulaz i izlaz podataka

Sadržaj

- Ulaz i izlaz podataka:
 - Funkcije getchar i putchar.

Funkcije za ulaz/izlaz

U standardnoj ulazno–izlaznoj biblioteci postoje sljedeće funkcije za **ulaz/izlaz** podataka (za standardne ulazne, odnosno, izlazne datoteke **stdin**, **stdout**):

- **getchar**, **putchar** — za **znakove**,
- **gets**, **puts** — za **stringove**,
- **scanf** i **printf** — za **formatirani** ulaz/izlaz.

Program koji koristi neku od tih funkcija

- **mora** uključiti datoteku zaglavlja **<stdio.h>**.

Funkcije `getchar` i `putchar`

Deklaracija ovih funkcija (zaglavlje, prototip) ima oblik:

```
int getchar(void);  
int putchar(int c);
```

Funkcija `getchar` čita **jedan znak** sa standardnog **ulaza** (tipično — tipkovnice).

Funkcija **nema** argumenata pa je sintaksa poziva:

```
c_var = getchar();
```

Funkcije `getchar` i `putchar` (nastavak)

Funkcija `putchar` šalje (tj. piše) **jedan znak** na standardni **izlaz** (tipično — ekran).

Ona uzima **jedan** argument (**znak** koji treba ispisati) i **vraća** cjelobrojnu vrijednost.

Poziv funkcije, najčešće, ima oblik

```
putchar(c_var);
```

pri čemu se vraćena vrijednost **ignorira**.

Pitanje. Zašto u deklaracijama piše tip `int`, a **ne** tip `char`?

Problem. Kako prepoznati i “označiti” **kraj** podataka, odnosno, **grešku**?

Funkcije `getchar` i `putchar` (nastavak)

Kada funkcija `getchar` naiđe na **kraj** ulaznih podataka — **vraća** vrijednost `EOF` (skraćeno od engl. `End of File`).

`EOF` je **simbolička** konstanta definirana u `<stdio.h>` koja signalizira **kraj** datoteke, odnosno, kraj ulaznih podataka (ulaz je tretiran kao datoteka `stdin`).

Konstanta `EOF` mora se **razlikovati** od znakova iz sustava znakova koje računalo koristi. Stoga funkcija `getchar` ne vraća vrijednost tipa `char`, već vrijednost tipa `int`, što daje dovoljno prostora za kôdiranje konstante `EOF` (obično `-1`).

Isto tako `putchar` uzima vrijednost tipa `int` i **vraća** vrijednost tipa `int`. **Vraćena** vrijednost je **znak** koji je ispisan ili `EOF`, ako ispis znaka **nije uspio**.

Primjer za getchar i putchar

Primjer. Program koji **kopira znak po znak** s ulaza na izlaz i pritom sva slova **pretvara** u velika.

U datoteci zaglavlja `<ctype.h>` deklarirana je funkcija **toupper** koja **pretvara mala** slova u **velika**, a sve druge znakove ostavlja na miru.

Ova funkcija radi **isto** što i funkcija `malo_u_veliko` s prošlog predavanja.

Primjer za getchar i putchar (nastavak)

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

int main(void) {
    int c;

    while ((c = getchar()) != EOF)
        putchar(toupper(c));

    return 0;
}
```

Pitanje. Što se događa ako piše samo `putchar(c);` ?

Primjer rekurzivne funkcije — naopako pisanje

Primjer. Funkcija čita znakove sa standardnog ulaza, sve dok ne naiđe na prijelaz u novu liniju, i ispisuje učitane znakove obrnutim redosljedom, tj. naopako, pa se tako i zove.

```
void naopako(void) {
    char znak;
    if ((znak = getchar()) != '\n') naopako();
    putchar(znak);
    return;
}
```

Rekurzija služi pamćenju učitanih znakova u lokalnoj varijabli znak. Ispis nakon rekurzivnog poziva daje ispis unatrag (kako se vraćamo iz rekurzije).

Naopako pisanje (nastavak)

Glavni program — funkcija `main` (v. `naopako.c`):

```
int main(void) {  
    printf(" Unesite niz znakova: ");  
    naopako();  
    return 0;  
}
```

Izvršavanjem s ulazom: `Zdravo`, dobit ćemo ovaj rezultat:

```
Unesite niz znakova: Zdravo
```

```
ovardZ
```

Prvo je ispisan **zadnji** učitani znak `\n`.

Čitanje i pisanje znakova — `scanf`, `printf`

Za čitanje jednog znaka, umjesto `c = getchar()`, možemo koristiti i

```
scanf("%c", &c);
```

Funkcija `scanf` vraća (nenegativan) broj uspješno učitanih podataka, ili `EOF`, ako je do greške ili kraja datoteke došlo prije početka čitanja prvog podatka.

Analogno, za pisanje jednog znaka, umjesto `putchar(c)`, možemo koristiti i

```
printf("%c", c);
```

Funkcija `printf` vraća broj ispisanih znakova (nenegativan) ili `EOF`, ako je došlo do greške.

Funkcije za obradu znakova

Sadržaj

- Obrada znakova:
 - Funkcije iz `ctype.h`.
 - Primjeri obrade niza znakova.

Datoteka zaglavlja <ctype.h>

Funkcije za **obradu znakova** deklarirane su u <ctype.h>.

Sve funkcije imaju **jedan** argument tipa **int**, koji **smije** biti:

- ili “**znak**” **EOF** (standardno **-1**, i zato je tip **int**),
- ili **znak** prikaziv kao **unsigned char** (standardni znak).

Izlazna vrijednost je tipa **int**.

Funkcije iz <ctype.h> možemo podijeliti u **dvije** grupe:

- funkcije za **provjeru znakova** — vraćaju **int** **različit** od **nule** (**istina**), ako ulazni znak **pripada** određenoj **grupi znakova**. U **protivnom**, vraćaju **nulu** (**laž**);
- funkcije za **pretvaranje znakova** — vraćaju **konvertirani** ulazni **znak**.

Funkcije iz <ctype.h>

Funkcije za provjeru znakova:

```
int isalpha(int c);    /* Slovo, malo ili veliko */
int isdigit(int c);   /* Numer. = dec. znamenka */
int isalnum(int c);   /* Alfnumericki */
int isxdigit(int c);  /* Heksadecimalna znam. */
int islower(int c);   /* Malo slovo */
int isupper(int c);   /* Veliko slovo */
int iscntrl(int c);   /* Kontrolni znak */
int isgraph(int c);   /* Ispisiv, bez blanka */
int isprint(int c);   /* Ispisiv, uklj. blank */
int ispunct(int c);   /* Ispisiv, bez blanka,
                       slova i dec. znamenki */
int isspace(int c);   /* Bl, \n, \t, \v, \f, \r */
```

Funkcije iz <ctype.h> (nastavak)

U 7-bitnom ASCII kôdu (0 do 0x7F, ili 0 do 127):

- **ispisivi** znakovi su: 0x20 (' ', tj. blank) do 0x7E ('~'),
- **kontrolni** znakovi su: 0 (NUL) do 0x1F (US) i 0x7F (DEL).

Primjer:

```
isdigit('0') = 1;    isdigit('C') = 0;
isalpha('0') = 0;    isalpha('C') = 1;
isxdigit('0') = 1;   isxdigit('C') = 1;
```

Funkcije za **pretvaranje** — mijenjaju samo **slova**:

```
int tolower(int c);    /* Velika u mala */
int toupper(int c);    /* Mala u velika */
```

Primjer — implementacije nekih funkcija

Primjer. Moguće **implementacije** nekih funkcija iz `<ctype.h>`.

Funkcija `isdigit`:

```
int isdigit(int c) {  
    return ('0' <= c && c <= '9');  
}
```

Funkcija `isalpha`:

```
int isalpha(int c) {  
    return ('a' <= c && c <= 'z' ||  
           'A' <= c && c <= 'Z');  
}
```

Primjer — implementacije toupper

Funkcija `toupper` (v. `Prog1`, funkcija `malo_u_veliko`):

```
char toupper(char c) {  
    if ('a' <= c && c <= 'z')  
        return ('A' + c - 'a');  
    else  
        return c;  
}
```

ili, uvjetnim operatorom:

```
char toupper(char c) {  
    return ('a' <= c && c <= 'z') ?  
        ('A' + c - 'a') : c;  
}
```

Obrada znakova

Provjera znakova — zadaci

Zadaci. Napišite funkciju koja ima jedan **znak** (tipa **int**) kao argument i **provjerava** je li taj znak:

- **samoglasnik** (može malo ili veliko slovo) (v. vježbe),
- **suglasnik** (može malo ili veliko slovo),
- znak za **kraj rečenice** (**.**, **?**, **!**),
- **arimetrički operator** u C-u (**+**, **-**, *****, **/**, **%**),
- **zagrada** (otvorena ili zatvorena; okrugla, uglata ili vitičasta).

Po ugledu na funkcije za provjeru znakova iz **<ctype.h>**, ako znak **zadovoljava** navedeni uvjet, izlazna vrijednost je **1**. U **protivnom**, izlazna vrijednost je **0**.

Provjera znakova — samoglasnik

Primjer. Funkcija `samoglasnik` provjerava je li zadani znak samoglasnik (v. vježbe).

```
#include <ctype.h>

int samoglasnik(int c)
{
    /* Pretvaramo c u malo slovo
       da skratimo ispitivanje */

    c = tolower(c);
    return (c == 'a' || c == 'e' || c == 'i' ||
           c == 'o' || c == 'u');
}
```

Provjera znakova — suglasnik

Primjer. Funkcija `suglasnik` provjerava je li zadani znak `suglasnik`. Koristimo činjenicu da je znak `suglasnik`

ako (i samo ako) je slovo i nije samoglasnik.

To je lakše od gomile provjera pojedinačnih znakova.

```
int suglasnik(int c)
{
    /* Koristi samoglasnik
       da skratimo ispitivanje */

    return (isalpha(c) && !samoglasnik(c));
}
```
