Lekcija 9 – Ulazno-izlazne operacije

Pregled

Ciljevi

- U ovoj lekciji:
  - Shvatićete ulazne i izlazne tokove (streams).
  - Koristićete sve mogućnosti za formatirano štampanje podataka.
  - Koristićete sve mogućnosti za formatirano unošenje podataka.

9.1 Uvod

- Kako da
  - Prezentujemo rezultate
  - Koristimo `scanf` i `printf`
  - Ulazni i izlazni tokovi (streams - input and output)
    - `gets`, `puts`, `getchar`, `putchar` (u `<stdio.h>`)

9.2 Tokovi (streams)

- Tokovi (streams)
  - Nizovi karaktera organizovani u linije
    - Svaka linija se sastoji od 0 ili više karaktera i završava karakterom newline
    - ANSI C mora podržavati linije sa najmanje 254 karaktera
  - Odrađuju sav ulaz i izlaz
  - Često može biti preusmjeren
    - Standardni ulaz – tastatura
    - Standardni izlaz – ekran
    - Standardna greška – ekran (screen)
    - Više u lekciji 11

9.3 Formatiranje izlaza pomoću `printf`

- `printf`
  - Precizno formatiranje izlaza
    - Specifikacije konverzije: flagovi, širina polja, preciznost, itd.
  - Može izvoditi zaokruživanje, poravnavanje kolona i lijevo i desno grupisanje, umetanje karaktera, eksponencijalni format, heksadecimalni format i fiksna širina i preciznost
- Format
  - `printf(` *format kontrolni string*, *ostali argumenti* `);`
  - Format kontrolni string: opisuje izlazni format
  - Ostali argumenti: odgovaraju svakoj specifikaciji konverzije u format kontrolnom stringu
    - Svaka specifikacija počinje znakom %, završava sa specifikatorom konverzije

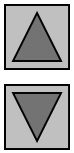
9.4 Štampanje cijelih brojeva

Specifikator konverzije	Opis
`d`	Prikazuje decimalni cio broj sa znakom.
`i`	Prikazuje decimalni cio broj sa znakom. (*Napomena:* `i` i `d` su različiti kada se koriste sa `scanf`.)
`o`	Prikazuje oktalni cio broj bez znaka.
`u`	Prikazuje decimalni cio broj bez znaka.
`x` ili `X`	Prikazuje heksadecimalni cio broj bez znaka. `X` prikazuje `0-9` i slova `A-F`, a `x` prikazueje cifre `0-9` i slova `a-f`.
`h` ili `l` (slovo `l`)	Prije svakog specifikatora daje indikaciju da li je broj `short` ili `long` integer. Slova `h` i `l` se preciznije zovu modifikatori dužine.

Fig. 9.1 Specifikatori konverzije za cijele brojeve

9.4 Štampanje cijelih brojeva

- Cio broj (integer)
  - Bez decimalne tačke: `25`, `0`, `-9`
  - Pozitivni, negativni ili nula
  - Samo se znak minus štampa po default-u

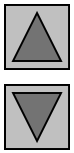


Outline

fig09_02.c

```c
/* Fig 9.2: fig09_02.c */
/* Using the integer conversion specifiers */
#include <stdio.h>

int main()
{
   printf( "%d\n", 455 );
   printf( "%i\n", 455 );  /* i same as d in printf */
   printf( "%d\n", +455 );
    printf( "%d\n", -455 );
    printf( "%hd\n", 32000 );
    printf( "%ld\n", 2000000000 );
    printf( "%o\n", 455 );
    printf( "%u\n", 455 );
    printf( "%u\n", -455 );
    printf( "%x\n", 455 );
    printf( "%X\n", 455 );

    return 0; /* indicates successful termination */

} /* end main */
```

```
455
455
455
-455
32000
2000000000
707
455
4294966841
1c7
1c7
```



Outline

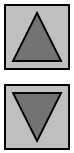
Program Ouptut

9.5 Štampanje realnih brojeva

- Realni brojevi (Floating Point Numbers)
  - Imaju decimalnu tačku (`33.5`)
  - Eksponencijalna notacija (kompjuterska verzija naučne-scientific notacije)
    - `150.3` je `1.503 x 10²` u naučnoj notaciji
    - `150.3` je `1.503E+02` u eksponencijalnoj (`E` za eksponent)
    - koristite `e` ili `E`
  - `f` – štampa realan broj sa najmanje jednom cifrom lijevo od decimalne tačke
  - `g` (ili `G`) - štampa realan broj sa `f` ili `e` bez završnih nula (`1.2300` postaje `1.23`)
    - Koristite eksponencijalno ako je eksponent manji od `−4` ili veći ili jednak od preciznosti (`6` cifara po defaultu)

9.5 Štampanje realnih brojeva

Specifikator konverzije	Opis
`e` ili `E`	Eksponencijalna notacija.
`f`	Prikaz vrijednosti realnog broja.
`g` ili `G`	Prikaz vrijednosti realnog broja bilo u `f` bilo u `e` (ili `E`) obliku.
`L`	Postavlja ispred bilo kog specifikatora indikator za `long double` realan broj



Outline

fig09_04.c

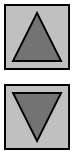
```
1  /* Fig 9.4: fig09_04.c */
2  /* Printing floating-point numbers with
3     floating-point conversion specifiers */
4
5  #include <stdio.h>
6
7  int main()
8  {
9     printf( "%e\n", 1234567.89 );
10    printf( "%e\n", +1234567.89 );
11    printf( "%e\n", -1234567.89 );
12    printf( "%E\n", 1234567.89 );
13    printf( "%f\n", 1234567.89 );
14    printf( "%g\n", 1234567.89 );
15    printf( "%G\n", 1234567.89 );
16
17    return 0; /* indicates successful termination */
18
19 } /* end main */
```

Program Output

```
1.234568e+006
1.234568e+006
-1.234568e+006
1.234568E+006
1234567.890000
1.23457e+006
1.23457E+006
```

9.6 Štampanje stringova i karaktera

- `c`
  - Štampa `char` argument
  - Ne može se koristiti za štampanje prvog karaktera u stringu
- `s`
  - Zahtijeva pokazivač na `char` kao argument
  - Štampa karaktere do pojave `NULL` (`'\0'`)
  - Ne može štampati `char` argument
- Zapamtite
  - Jednostruki navodnici za karaktere (`'z'`)
  - Dvostruki navodnici za stringove `"z"` (koji u stvari sadrži dva karaktera, `'z'` i `'\0'`)



Outline

fig09_05.c

```
1  /* Fig 9.5: fig09_05c */
2  /* Printing strings and characters */
3  #include <stdio.h>
4
5  int main()
6  {
7     char character = 'A'; /* initialize char */
8     char string[] = "This is a string"; /* initialize char array */
9     const char *stringPtr = "This is also a string"; /* char pointer */
10
11    printf( "%c\n", character );
12    printf( "%s\n", "This is a string" );
13    printf( "%s\n", string );
14    printf( "%s\n", stringPtr );
15
16    return 0; /* indicates successful termination */
17
18 } /* end main */
```

```
A
This is a string
This is a string
This is also a string
```

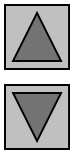
9.7 Ostali specifikatori konverzije

- p
  - Prikazuje pokazivačku vrijednost (adresu)
- n
  - Čuva broj karaktera koje je već prikazazao tekući `printf`
  - Ima pokazivač na cio broj kao argument
  - Ništa se ne štampa sa `%n` specifikacijom
  - Svaki poziv `printf` vraća vrijednost
    - Broj prikazanih karaktera
    - Negativan broj u slučaju greške
- %
  - Prikazuje znak procenta
  - %%

9.7 Ostali specifikatori konverzije

Specifikator konverzije	Opis
p	Prikazuje pokazivačku vrijednost.
n	Čuva broj već prikazanih karaktera u tekućoj `printf` naredbi. Odgovarajući argument mora biti pokazivač na cio broj. Ništa se ne prikazuje.
%	Prikazuje simbol procenta

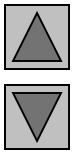
Fig 9.6 Ostali specifikatori.



Outline

fig09_07.c (1 of 2)

```
1  /* Fig 9.7: fig09_07.c */
2  /* Using the p, n, and % conversion specifiers */
3  #include <stdio.h>
4
5  int main()
6  {
7     int *ptr;      /* define pointer to int */
8     int x = 12345; /* initialize int x */
9     int y;         /* define int y */
10
11    ptr = &x;      /* assign address of x to ptr */
12    printf( "The value of ptr is %p\n", ptr );
13    printf( "The address of x is %p\n\n", &x );
14
15    printf( "Total characters printed on this line:%n", &y );
16    printf( " %d\n\n", y );
17
18    y = printf( "This line has 28 characters\n" );
19    printf( "%d characters were printed\n\n", y );
20
21    printf( "Printing a %% in a format control string\n" );
22
23    return 0; /* indicates successful termination */
24
25 } /* end main */
```



Outline

Program Output

```
The value of ptr is 0012FF78
The address of x is 0012FF78

Total characters printed on this line: 38

This line has 28 characters
28 characters were printed

Printing a % in a format control string
```

9.8 Štampanje sa širinom polja i preciznošću

- Širina polja (field width)
  - Veličina fiktivnog polja u kojem se štampa podatak
  - Ako je širina veća od podatka, default je desno poravnavanje (right justified)
    - Ako je polje male veličine, povećava se za podatak
    - Minus znak koristi jedan karakter u polju
  - Širina polja umeće se između % i specifikatora konverzije
  - %4d – polje širine 4

9.8 Štampanje sa širinom polja i preciznošću

- Preciznost
  - Značenje varira u zavisnosti od tipa
  - Cio broj (default 1)
    - Minimalni broj cifara za prikaz
      - Ako je podatak premali, dodaju se nule kao prefiksi
  - Floating point
    - Broj cifara koji se pojavljuju iza decimalne tačke (`e` i `f`)
      - Za `g` – maksimalan broj značajnih cifara
  - Stringovi
    - Maksimalan broj karaktera iz stringa
  - Format
    - Koristite tačku (`.`) pa preciznost poslije `%`

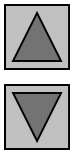
```
%.3f
```

9.8 Štampanje sa širinom polja i preciznošću

- Širina polja i preciznost istovremeno
  - `%width.precision`

    `%5.3f`
  - Negativna širina polja – lijevo poravnavanje (left justified)
  - Pozitivna širina polja – desno poravnavanje (right justified)
  - Preciznost mora biti pozitivna
  - Može se koristiti cjelobrojni izraz za određivanje širine polja i preciznosti
    - Stavite (*) na mjesto za širinu polja ili preciznost
      - Poklapaju se sa `int` argumentima u listi argumenata
    - Primjer:

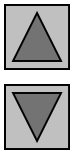
      ```
      printf( "%*.*f", 7, 2, 98.736 );
      ```



Outline

fig09_08.c

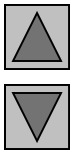
```
1  /* Fig 9.8: fig09_08.c */
2  /* Printing integers right-justified */
3  #include <stdio.h>
4
5  int main()
6  {
7     printf( "%4d\n", 1 );
8     printf( "%4d\n", 12 );
9     printf( "%4d\n", 123 );
10    printf( "%4d\n", 1234 );
11    printf( "%4d\n\n", 12345 );
12
13    printf( "%4d\n", -1 );
14    printf( "%4d\n", -12 );
15    printf( "%4d\n", -123 );
16    printf( "%4d\n", -1234 );
17    printf( "%4d\n", -12345 );
18
19    return 0; /* indicates successful termination */
20
21 } /* end main */
```



Outline

Program Output

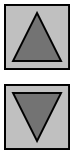
```
   1
  12
 123
1234
12345

  -1
 -12
-123
-1234
-12345
```



Outline

fig09_09.c

```
1  /* Fig 9.9: fig09_09.c */
2  /* Using precision while printing integers,
3     floating-point numbers, and strings */
4  #include <stdio.h>
5
6  int main()
7  {
8     int i = 873;                  /* initialize int i */
9     double f = 123.94536;         /* initialize double f */
10    char s[] = "Happy Birthday"; /* initialize char array s */
11
12    printf( "Using precision for integers\n" );
13    printf( "\t%.4d\n\t%.9d\n\n", i, i );
14
15    printf( "Using precision for floating-point numbers\n" );
16    printf( "\t%.3f\n\t%.3e\n\t%.3g\n\n", f, f, f );
17
18    printf( "Using precision for strings\n" );
19    printf( "\t%.11s\n", s );
20
21    return 0; /* indicates successful termination */
22
23 } /* end main */
```



Outline

Program Output

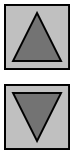
```
Using precision for integers
        0873
        000000873

Using precision for floating-point numbers
        123.945
        1.239e+002
        124

Using precision for strings
        Happy Birth
```

9.9 Korišćenje flagova u `printf`

- Flagovi
  - Dopuna mogućnostima formatiranja
  - Postaviti flag desno od znaka %
  - Više flagova se može kombinovati

Flag	Description
- (minus sign)	Lijevo poravnavanje.
+ (plus sign)	Prikazuje + ispred pozitivnih i – ispred negativnog broja.
space	Štampa space ispred pozitivnog broja – ne štampa se sa + flagom.
#	Prefiks `0` ako se štampa oktalna vrijednost.
	Prefiks `0x` ili `0X` ako se štampa heksadecimalna vrijednost.
	Obavezno se štampa decimalna tačka za brojeve koje ne sadrže decimalnu tačku a štampaju se sa `e`, `E`, `f`, `g` ili `G` (Normalno, decimalna tačka se štampa samo ako ima cifara desno od nje). Za `g` i `G` specifikatore, ne eleminišu se vodeće nule.
0 (zero)	Popunjavanje polja vodećim nulama.

Fig. 9.10 Flagovi.



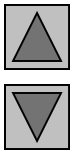
Outline

fig09_11.c

```
1  /* Fig 9.11: fig09_11.c */
2  /* Right justifying and left justifying values */
3  #include <stdio.h>
4
5  int main()
6  {
7     printf( "%10s%10d%10c%10f\n\n", "hello", 7, 'a', 1.23 );
8     printf( "%-10s%-10d%-10c%-10f\n", "hello", 7, 'a', 1.23 );
9
10    return 0; /* indicates successful termination */
11
12 } /* end main */
```

Program Output

```
 hello         7         a  1.230000

hello     7         a         1.230000
```



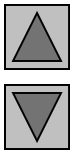
Outline

fig09_12.c

```
1  /* Fig 9.12: fig09_12.c */
2  /* Printing numbers with and without the + flag */
3  #include <stdio.h>
4
5  int main()
6  {
7     printf( "%d\n%d\n", 786, -786 );
8     printf( "%+d\n%+d\n", 786, -786 );
9
10    return 0; /* indicates successful termination */
11
12 } /* end main */
```

Program Output

```
786
-786
+786
-786
```



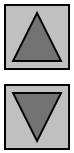
Outline

fig09_13.c

```
1  /* Fig 9.13: fig09_13.c */
2  /* Printing a space before signed values
3     not preceded by + or - */
4  #include <stdio.h>
5
6  int main()
7  {
8     printf( "% d\n% d\n", 547, -547 );
9
10    return 0; /* indicates successful termination */
11
12 } /* end main */
```

Program Output

```
 547
-547
```



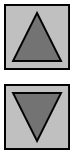
Outline

fig09_14.c

```
1  /* Fig 9.14: fig09_14.c */
2  /* Using the # flag with conversion specifiers
3     o, x, X and any floating-point specifier */
4  #include <stdio.h>
5
6  int main()
7  {
8     int c = 1427;      /* initialize c */
9     double p = 1427.0; /* initialize p */
10
11    printf( "%#o\n", c );
12    printf( "%#x\n", c );
13    printf( "%#X\n", c );
14    printf( "\n%g\n", p );
15    printf( "%#g\n", p );
16
17    return 0; /* indicates successful termination */
18
19 } /* end main */
```

Program Output

```
02623
0x593
0X593

1427
1427.00
```



Outline

fig09_15.c

```
1  /* Fig 9.15: fig09_15.c */
2  /* Printing with the 0( zero ) flag fills in leading zeros */
3  #include <stdio.h>
4
5  int main()
6  {
7     printf( "%+09d\n", 452 );
8     printf( "%09d\n", 452 );
9
10    return 0; /* indicates successful termination */
11
12 } /* end main */
```

Program Output

```
+00000452
000000452
```

9.10 Štampanje literala i escape sekvenci

- Štampanje literala
  - Većina karaktera se može štampati
  - Postoje "problematični" karakteri, kao navodnici "
  - Moraju se prikazati pomoću escape sekvenci
    - Predstavljaju se pomoću backslash (\) pa zatim escape karakter

9.10 Štampanje literala i escape sekvenci

Escape sekvence	Òpis
\'	Prikazuje jednostruki navodnik - single quote (') character.
\"	Prikazuje dvostruki navodnik - (") character.
\?	Prikazuje (?) karakter.
\\	Prikazuje backslash (\) karakter.
\a	Zvučni signal ili vizuelno upozorenje.
\b	Pomjera kursor jednu poziciju unazad u tekućoj liniji.
\f	Pomjera kursor na početak sledeće logičke stranice.
\n	Pomjera kursor na početak sledeće linije.
\r	Pomjera kursor na početak tekuće linije.
\t	Pomjera kursor na sledeću horizontalnu tab poziciju.
\v	Pomjera kursor na sledeću vertikalnu tab poziciju.

Fig. 9.16 Escape sekvence.

9.11 Formatiranje ulaza sa scanf

Specifikator konverzije	Opis
Cijeli brojevi	
d	Učitava cio broj, moguće sa znakom. Odgovarajući argument – pokazivač na cio broj.
i	Učitava cio broj, decimalni, oktalni ili heksadecimalni, moguće sa znakom. Odgovarajući argument – pokazivač na cio broj.
o	Učitava cio oktalni broj, moguće sa znakom. Odgovarajući argument – pokazivač na cio neoznačen broj (unsigned integer).
u	Učitava neoznačen cio broj. Odgovarajući argument – pokazivač na neoznačen cio broj.
x ili X	Učitava heksadecimalan broj. Odgovarajući argument – pokazivač na neoznačen cio broj.
h ili l	Indikacija da li se učitava `short` ili `long` integer.

Fig. 9.17 Specifikatori konverzije za `scanf`.

9.11 Formatiranje ulaza sa scanf

Specifikatori konverzije	Opis
Floating-point brojevi	
`e`, `E`, `f`, `g` ili `G`	Učitava floating-point vrijednost. Odgovarajući argument – pokazivač na floating-point promjenljivu.
`l` ili `L`	Indikacija da li se učitava `double` ili `long double` vrijednost.
Karakters i stringovi	
`c`	Učitava karakter. Odgovarajući argument – pokazivač na `char`, ne dodaje se NULL `('\0')`.
`s`	Učitava string. Odgovarajući argument – pokazivač na niz tipa `char` koji je dovoljno veliki za čuvanje stringa i završnog karaktera `('\0')` koji se automatski dodaje.
Scan set	
[scan characters]	Skenira string u potrazi za nizom karaketra.
Posebno	
`p`	Učitava adresu u istoj formi koju proizvodi printf sa upotrebom `%p`.
`n`	Čuva broj karaketra učitanih do tog trenutka u tekućeoj naredbi `scanf`. Odgovarajući argument – pokazivač na integer
`%`	Preskače znak (`%`) u ulazu.

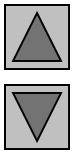
Fig. 9.17 **S**pecifikatori konverzije za scanf.

9.11 Formatiranje ulaza sa scanf

- scanf
  - Formatiranje ulaza
  - Mogućnosti
    - Ulaz svih tipova podataka
    - Ulaz specifičnih karaktera
    - Izbjegavanje specifičnih karaktera
- Format
  - scanf(*format kontrolni string*, *ostali argumenti*);
  - Format kontrolni string
    - Opisuje formate ulaznih veličina
  - Ostali argumenti
    - Pokazivači na promjenljive koje će čuvati učitane vrijednosti
  - Mogu se uključiti širina polja da bi se pročitao samo određeni broj karaktera iz ulaznog toka

9.11 Formatiranje ulaza sa scanf

- Skeniranje skupova
  - Skup karaktera između `[]`
    - Ispred skupa mora biti znak `%`
  - *scanf* čita ulazi tok podataka i traži karaktere koji pripadaju skupu.
    - Kada se dogodi poklapanje, karakter se sačuva u specificiranom nizu
    - Skeniranje se prekida kada se naiđe na karakter koji ne pripada skupu
  - Invertovani skupovi skeniranja
    - Koristi se simbol `^`: `[^aeiou]`
    - Čuvaju se karakteri koji ne pripadaju datom skupu
- Preskakanje karaktera
  - Uključiti karaktere koji se preskaču u format kontrol stringu
  - Ili, koristiti `*` (assignment suppression character)
    - Preskače se svaki tip karaktera bez čuvanja samog karaktera



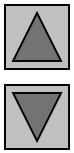
Outline

fig09_18.c

```c
/* Fig 9.18: fig09_18.c */
/* Reading integers */
#include <stdio.h>

int main()
{
   int a; /* define a */
   int b; /* define b */
   int c; /* define c */
   int d; /* define d */
   int e; /* define e */
   int f; /* define f */
   int g; /* define g */

   printf( "Enter seven integers: " );
   scanf( "%d%i%i%i%o%u%x", &a, &b, &c, &d, &e, &f, &g );

   printf( "The input displayed as decimal integers is:\n" );
   printf( "%d %d %d %d %d %d %d\n", a, b, c, d, e, f, g );

   return 0; /* indicates successful termination */

} /* end main */
```

Program Output

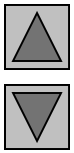
```
Enter seven integers: -70 -70 070 0x70 70 70 70
The input displayed as decimal integers is:
-70 -70 56 112 56 70 112
```



Outline

fig09_19.c

```
1  /* Fig 9.19: fig09_19.c */
2  /* Reading floating-point numbers */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main()
7  {
8     double a; /* define a */
9     double b; /* define b */
10    double c; /* define c */
11
12    printf( "Enter three floating-point numbers: \n" );
13    scanf( "%le%lf%lg", &a, &b, &c );
14
15    printf( "Here are the numbers entered in plain\n" );
16    printf( "floating-point notation:\n" );
17    printf( "%f\n%f\n%f\n", a, b, c );
18
19    return 0; /* indicates successful termination */
20
21 } /* end main */
```



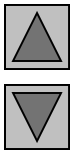
Outline

fig09_20.c

```c
/* Fig 9.20: fig09_20.c */
/* Reading characters and strings */
#include <stdio.h>

int main()
{
   char x;      /* define x */
   char y[ 9 ]; /* define array y */

   printf( "Enter a string: " );
   scanf( "%c%s", &x, y );

   printf( "The input was:\n" );
   printf( "the character \"%c\" ", x );
   printf( "and the string \"%s\"\n", y );

   return 0; /* indicates successful termination */

} /* end main */
```

Program Output

```
Enter a string: Sunday
The input was:
the character "S" and the string "unday"
```



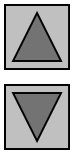
Outline

fig09_21.c

```c
/* Fig 9.21: fig09_21.c */
/* Using a scan set */
#include <stdio.h>

/* function main begins program execution */
int main()
{
   char z[ 9 ]; /* define array z */

   printf( "Enter string: " );
   scanf( "%[aeiou]", z ); /* search for set of characters */

   printf( "The input was \"%s\"\n", z );

   return 0; /* indicates successful termination */

} /* end main */
```

Program Output

```
Enter string: ooeeooahah
The input was "ooeeooa"
```



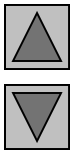
Outline

fig09_22.c

```
1  /* Fig 9.22: fig09_22.c */
2  /* Using an inverted scan set */
3  #include <stdio.h>
4
5  int main()
6  {
7     char z[ 9 ] = { '\0' }; /* initialize array z */
8
9     printf( "Enter a string: " );
10    scanf( "%[^aeiou]", z ); /* inverted scan set */
11
12    printf( "The input was \"%s\"\n", z );
13
14    return 0; /* indicates successful termination */
15
16 } /* end main */
```

Program Output

```
Enter a string: String
The input was "Str"
```



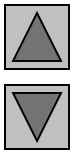
Outline

fig09_23.c

```c
/* Fig 9.23: fig09_23.c */
/* inputting data with a field width */
#include <stdio.h>

int main()
{
   int x; /* define x */
   int y; /* define y */

    printf( "Enter a six digit integer: " );
    scanf( "%2d%d", &x, &y );

    printf( "The integers input were %d and %d\n", x, y );

    return 0; /* indicates successful termination */

} /* end main */
```

Program Output

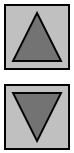
```
Enter a six digit integer: 123456
The integers input were 12 and 3456
```



Outline

fig09_24.c

```
1  /* Fig 9.24: fig09_24.c */
2  /* Reading and discarding characters from the input stream */
3  #include <stdio.h>
4
5  int main()
6  {
7     int month1; /* define month1 */
8     int day1;   /* define day1 */
9     int year1;  /* define year1 */
10    int month2; /* define month2 */
11    int day2;   /* define day2 */
12    int year2;  /* define year2 */
13
14    printf( "Enter a date in the form mm-dd-yyyy: " );
15    scanf( "%d%*c%d%*c%d", &month1, &day1, &year1 );
16
17    printf( "month = %d  day = %d  year = %d\n\n", month1, day1, year1 );
18
19    printf( "Enter a date in the form mm/dd/yyyy: " );
20    scanf( "%d%*c%d%*c%d", &month2, &day2, &year2 );
21
22    printf( "month = %d  day = %d  year = %d\n", month2, day2, year2 );
23
24    return 0; /* indicates successful termination */
25
26 } /* end main */
```



Outline

Program Output

```
Enter a date in the form mm-dd-yyyy: 11-18-2003
month = 11  day = 18  year = 2003

Enter a date in the form mm/dd/yyyy: 11/18/2003
month = 11  day = 18  year = 2003
```