



توابع ورودی-خروجی



توابع ورودی-خروجی در C

- اغلب برنامه ها باید اطلاعاتی را از کاربر بگیرند و روی آنها پردازش دهند و نتیجه پردازش را به کاربر اعلام نمایند
- در زبان برنامه نویسی C توابع متنوعی برای عملیات ورودی-خروجی تعریف شده‌اند که هر کدام ویژگیهای خاص خود را دارند
- عمومی ترین توابع ورودی-خروجی در زبان C تابع **scanf** برای ورود اطلاعات و **printf** برای نمایش اطلاعات میباشند

توابع ورودی-خروجی در C

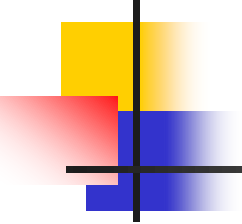
(ادامه)

■ توابع ورودی-خروجی دیگری که در C وجود دارند و از آنها میتوان استفاده کرد عبارتند از:

- getch, getchar,
- putch, putchar,
- gets, puts

تابع printf

- این تابع برای نمایش اطلاعات به کاربر بر روی خروجی استاندارد (مانیتور) بکار میرود.
- قالب استفاده از این دستور به شکل زیر میباشد
- `printf("[متغیرها], رشته کنترلی");`
- این تابع در کتابخانه `stdio.h` تعریف شده است
- رشته کنترلی که داخل کتیشن (") قرار میگیرد از متنهای ثابت و کاراکترهای کنترلی و رشته گریز تشکیل میشود
- در بخش متغیرها، متناظر با هر کاراکتر کنترلی در رشته کنترلی باید یک متغیر وجود داشته باشد
- کاراکتر کنترلی باید متناسب با نوع متغیر باشد



کاراکترهای کنترلی

- کاراکترهای کنترلی با % آغاز میشوند.
- با استفاده از کاراکترهای کنترلی میتوان خروجی را به شکل دلخواه نشان داد
- فهرست کاراکترهای کنترلی به شرح ذیل میباشد:

Char	Expected Input	Format of output
%d	Integer	Signed decimal integer
%i	Integer	Signed decimal integer
%o	Integer	Unsigned octal integer
%u	Integer	Unsigned decimal integer
%x	Integer	Unsigned hexadecimal int (with a, b, c, d, e, f)
%X	Integer	Unsigned hexadecimal int (with A, B, C, D, E, F)
%f	Floating point	Signed value of the form [-]dddd.dddd.
%e	Floating point	Signed value of the form [+/-]dddd.dd e[+/-]ddd
%g	Floating point	Signed value in either e or f form, based on given value and precision. Trailing zeros and the decimal point are printed if necessary.
%E	Floating point	Same as e; with E for exponent.
%G	Floating point	Same as g; with E for exponent if e format used
%c	Character	Single character
%s	String pointer	Prints characters until a null-terminator is pressed or precision is reached

- نمایش یک عدد صحیح در خروجی. Ex 1.

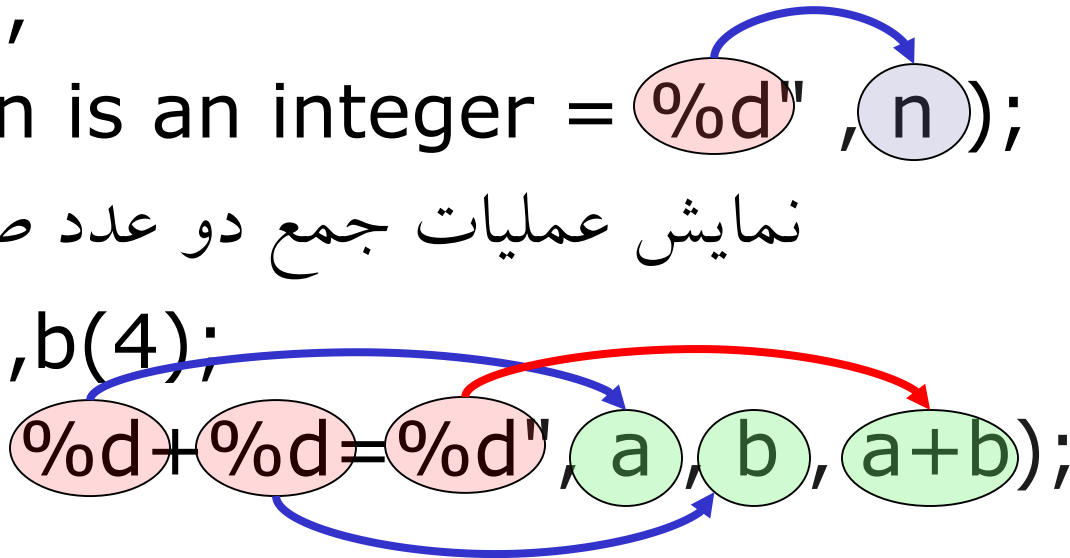
- `int n=2;`

- `printf("n is an integer = %d", n);`

- نمایش عملیات جمع دو عدد صحیح. Ex 2.

- `int a(3), b(4);`

- `printf("%d+%d=%d", a, b, a+b);`

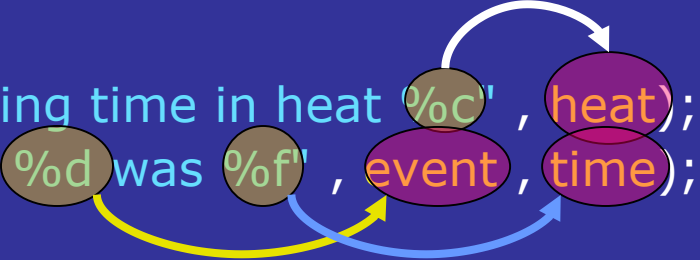


مثال ۳

```
int main()
{
    int event;
    char heat;
    float time;

    event = 5;
    heat = 'C';
    time = 27.25;

    printf( "The winning time in heat '%c',", heat);
    printf( " of event %d was %f", event, time);
    return 0;
}
```



خروجی:

The winning time in heat C of event 5 was 27.250000

```
int main()
{
    int x=31;
    float y=148.5;
    char z[10] = {"computer"};
    printf ( "%d %f %s" , x , y , z );
    return 0;
}
```

خروجی:

31 148.500000 computer

مثال ۵

```
int main()
{
    float x=50.0, y=0.25;
    printf ( "%f %f %f %f\n" , x , y , x*y , x/y );
    printf ( "%e %e %e %e" , x , y , x*y , x/y );
    return 0;
}
```

خروجی:

```
50.000000 0.250000 12.500000 200.000000
5.000000e+01 2.500000e-01 1.250000e+01 2.000000e+02
```

کاراکترهای تعیین میدان

- در برنامه قبل دیدیم که متغیر اعشاری y با 6 رقم ممیز چاپ شده است، درحالی که تنها دو رقم از این ارقام با ارزش هستند.
- از این کاراکترها برای مرتب شدن خروجی و زیر هم بودن اطلاعات استفاده میشود.

<code>%m.nf</code>	چاپ یک عدد اعشاری در طول m که n رقم اعشار دارد
<code>%md</code>	چاپ یک عدد صحیح در طول m

`%6.2f`

***.*
 └─┘
└──────┘

`%.3f`

***.*
 └─┘

`%6d`

└────────┘

کاراکترهای تعیین میدان

مثال

```
int x = 123;  
float m = 4872.458;
```

رشته کنترلی	متغیر	نحوه نمایش
%4d	x	□123
%5d	x	□□123
%2d	x	23
%10.4f	m	□4872.4580
%.4f	m	4872.4580
%.2f	m	4872.46

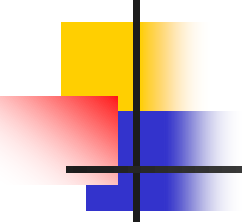
مثال ۶

```
int main()
{
    float x=123.456;
    printf ( "%7.0f\n%7.3f\n%7.1f\n\n" , x , x , x );
    printf ( "%12e\n%12.5e\n%12.3e" , x , x , x );
    return 0;
}
```

```
123
123.456
123.5

1.234560e+02
1.23456e+02
1.235e+02
```

خروجی:

- 
- در صورتی که بخواهیم خانه های خالی با صفر جایگزین شود، بعد از علامت % از کاراکتر 0 (صفر) استفاده میکنیم

```
int x = 123;  
float m = 4872.458;
```

رشته کنترلی	متغیر	نحوه نمایش
%4d	x	□123
%04d	x	0123
%10.3f	m	□□4872.458
%010.3f	m	004872.458

رشته های گریز

- در رشته کنترل کاراکترهایی میتوان استفاده کرد که موقعیت مکان نما را براساس نیاز ما تغییر میدهد.
- رشته های گریز با \ آغاز میشوند و نمونه ای از آنها عبارت است از:

عملکرد	رشته گریز
سطر جدید	\n
جهش هشت کاراکتری	\t
بازگشت به عقب	\b
چاپ علامت نقل قول تکی	\'
چاپ علامت نقل قول دوتایی	\"
نمایش علامت backslash	\\

مثال ۷- مثال

■ برنامه ای بنویسید که خروجی زیر را بصورت مرتب نمایش دهد.

```
num1 num2 sumation
```

```
1 2 3
```

```
100 2 102
```

```
1010 256 1266
```

```
5 56 61
```

```

int main()
{
    printf ( "num1\t num2\t sumation\n" );
    printf ( "%4d\t %4d\t %8d\n" , 1 , 2 , 3 );
    printf ( "%4d\t %4d\t %8d\n" , 100 , 2 , 102 );
    printf ( "%4d\t %4d\t %8d\n" , 1010 , 256 , 1266 );
    printf ( "%4d\t %4d\t %8d\n" , 5 , 56 , 61 );
    return 0;
}

```

C:\ Borland C++ for DOS

num1	num2	sumation
1	2	3
100	2	102
1010	256	1266
5	56	61

■ ۱- برنامه ای بنویسید که خروجی به شکل زیر تولید نماید

```
*****
```

```
* Hello World *
```

```
*****
```

■ ۲- برنامه ای بنویسید که نام شما را در وسط صفحه نمایش چاپ کند (برای اینکار از تابع **gotoxy** استفاده کنید).

■ ۳- برنامه ای بنویسید که عدد **PI** را با دقت های 1، 2، 5 و 10 رقم اعشار نمایش دهد

تابع scanf

- مهمترین تابع ورودی در C میباشد که برای گرفتن اطلاعات از IO استاندارد یعنی کی‌برد بکار برده میشود.
- قالب استفاده از این دستور به شکل زیر میباشد
- `scanf(<رشته کنترلی>, <آدرس متغیرها>);`
- این تابع در کتابخانه `stdio.h` تعریف شده است
- رشته کنترلی داخل کتیشن (") نوشته شده و از کاراکترهای کنترلی تشکیل میشود.
- در بخش متغیرها، متناظر با هر کاراکتر کنترلی در رشته کنترلی باید آدرس یک متغیر وجود داشته باشد.
- کاراکتر کنترلی استفاده شده در این تابع همانند کاراکترهای کنترلی در تابع `printf` هستند

اپراتورهای & و *

- همانطور که اشاره شد متغیرها آدرس محللهایی در حافظه هستند
- با استفاده از نام متغیر به تنهایی ما میتوانیم به محتویات آن دسترسی پیدا کنیم
- در برخی موارد لازم است ما آدرس محل حافظه را داشته باشیم در اینصورت از اپراتور & استفاده میکنیم.
- اپراتور * برای دسترسی به محتویات یک خانه حافظه استفاده میشود. یعنی این اپراتور آدرس یک خانه حافظه را گرفته و محتویات آنرا برمیگرداند.
(در ادامه درس، بخش اشاره گرها مفصل در این زمینه صحبت خواهد شد)

مثال:

- آدرس حافظه (آدرس متغیر) $\rightarrow$ (نام متغیر) $\&$
- مقدار حافظه $\rightarrow$ (آدرس حافظه) $*$
- فرض کنید متغیر **ch** از نوع کاراکتر تعریف شده باشد و سیستم عامل خانه **110** را به آن اختصاص داده باشد و ما مقدار **44** را در آن ذخیره کرده باشیم. در اینصورت:
 - $\&(ch) \rightarrow 110$
 - $ch \leftarrow 44$
 - $*(110) \rightarrow 44$

مثال ۲: برنامه نمایش یک متغیر به همراه آدرس متغیر

```
int main()
{
    int num ;
    num = 2;
    printf ( "Value:%d, Address:%d" , num , &num );
    return 0;
}
```

Value:2, Address:3536

خروجی:

مثال ۲: برنامه تبدیل سن شخص به روزهای زندگی

```
int main()
{
    float years , days ;
    printf ( "please type your age in years: " );
    scanf ( "%f" , &years );
    days = years * 365;
    printf ( "\nyou are %.1f days old.\n" , days );
    return 0;
}
```

please type your age in years: 5

خروجی:

you are 1825.0 days old.

تمرین

- ۱- برنامه ای بنویسید که ۳ عدد صحیح را از کاربر گرفته و حاصل جمع آنها را نمایش دهد
- ۲- برنامه ای بنویسید که ۳ عدد اعشاری را از کاربر گرفته و حاصل جمع آنها را نمایش دهد
- ۳- برنامه ای بنویسید که X و Y را از کاربر گرفته X^Y را نمایش دهد. برای محاسبه توان از تابع `pow` استفاده نمایید
- ۴- برنامه ای بنویسید که یک عدد صحیح را از کاربر گرفته و در خروجی زوج یا فرد بودن آنرا نمایش دهد اگر زوج بود عدد ۰ و اگر فرد بود عدد ۱ نمایش داده شود.
- ۵- برنامه ای بنویسید که X , Y از کاربر گرفته و باقیمانده تقسیم X بر Y را نمایش دهد
- ۶- برنامه ای بنویسید که شعاع یک کره را گرفته و مساحت جانبی و حجم آنرا در خروجی نمایش دهد
- ۷- برنامه ای بنویسید که طول، عرض و ارتفاع یک مکعب مستطیل را گرفته و مساحت جانبی و حجم آنرا در خروجی نمایش دهد
- ۸- برنامه ای بنویسید که یک مقدار درخواستی پول را از کاربر گرفته و تعداد اسکناسهای ۱۰۰۰ تومانی، ۱۰۰ تومانی، ۵۰ تومانی و ۱ تومانی مورد نیاز را محاسبه نماید.

مثال ۳: گرفتن چند متغیر از کاربر و نمایش

```
int main()
{
    int event;
    char heat;
    float time;

    printf( "Type numbers for event heat and time: ");
    scanf( "%d %c %f", &event , &heat , &time);

    printf( "The winning time in heat %c" , heat);
    printf( " of event %d was %f" , event , time);
    return 0;
}
```

خروجی:

```
Type numbers for event heat and time: 4 B 36.34
The winning time in heat B of event 4 was 36.340000
```

تمرین کلاسی:

برنامه ای بنویسید که ضرایب معادله درجه ۲ را گرفته و ریشه های معادله را نمایش دهد.

```
int main()
{
    float a , b , c;
    printf("Enter a b c: ");
    scanf( "%f %f %f", &a , &b , &c);

    float delta;
    delta= b * b - 4 * a * c ;

    float x1 , x2 ;
    x1 = (-b + sqrt( delta )) / (2 * a);
    x2 = (-b - sqrt( delta )) / (2 * a);
    printf( "x1=%f , x2=%f " , x1 , x2);
    return 0;
}
```

Enter a b c: 1 -5 6

x1=3.000000 , x2=2.000000

خروجی:



توابع `getch()`، `getche()` و `getchar()`

- توابع `getch()`، `getche()` و `getchar()` برای گرفتن یک کاراکتر از کاربر استفاده میشوند.
- تابع `getch` با فشار دادن یک کلید، کد آن را برمیگرداند و هیچ کاراکتری در روی مانیتور نمایش داده نمیشود.
- تابع `getche` همانند تابع `getch` عمل میکند با این تفاوت که کاراکتر وارد شده روی مانیتور نقش میبندد
- تابع `getchar` همانند تابع `getche` عمل نموده و برای وارد کردن کاراکتر لازم است بعد از کاراکتر مورد نظر کلید `Enter` هم فشرده شود

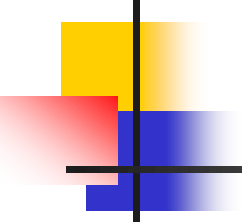
مثال ۱: برنامه ای بنویسید که یک کاراکتر از کاربر بگیرد و در خروجی با یک پیام نمایش دهد

```
int main()
{
    char ch;

    printf( "Enter a character: ");
    ch = getch();

    printf( "\nInput character is: %c" , ch);
    return 0;
}
```

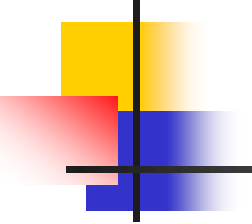
```
Enter a character:
Input character is: a
```



کد اسکی `ascii`

- هر کاراکتر در سیستم **ANSI** در یک بایت ذخیره میشود.
- در زبان **C** نوع داده ای **char** برای ذخیره یک کاراکتر استفاده میشود.
- هر کاراکتر با یک کد ارائه میگردد که به آن کد اسکی میگویند.
- برای نمونه کد مربوط به کاراکتر **A** برابر **65** میباشد

متغیرهای char در زبان C



- مقداردهی به متغیر char به شکل زیر را در نظر بگیرید:

char ch='A' ■

- دستور فوق باعث میشود کد اسکی کاراکتر A در متغیر ch قرار گیرد

- میتوان بجای دستور فوق از دستور زیر استفاده نمود

char ch=65 ■

- هر دو دستور فوق یک عمل را انجام میدهند.

- کامپایلر C انواع داده‌ای از نوع کاراکتر را به کد اسکی آنها تبدیل میکند

مثال ۲: کد اسکی

```
int main()
{
    char ch;

    ch= 'A' ;

    printf( "character is: %c\n" , ch);
    printf( "code is: %d\n" , ch);

    return 0;
}
```

```
int main()
{
    char ch;

    ch= 65 ;

    printf( "character is: %c\n" , ch);
    printf( "code is: %d\n" , ch);

    return 0;
}
```

```
character is: A
code is: 65
```