مبانی کامپیوتر و برنامه‌سازی

عملگرهای انتساب

- زبان C تعدادی عملگر انتساب برای خلاصه‌سازی عبارات انتساب دارد

بعنوان مثال

```
C = C + 3;
```

- می تواند به صورت عملگر =+ خلاصه شود:

```
C += 3;
```

- عملگر =+ مقدار عبارتی که در سمت راست عملگر قرار دارد را با مقدار فعلی متغیر جمع کرده و نتیجه را در متغیر ذخیره می کند.

VARIABLE = VARIABLE OPERATOR EXPRESSION ;

- که عملگر (Operator) آن یکی از چهار عمل اصلی و یا باقیمانده باشد، را می‌توان به شکل زیر نوشت:

VARIABLE OPERATOR= EXPRESSION ;

- بنابراین انتساب C+=3، 3 را به C اضافه می کند

انتساب

عملگرهای ریاضی با مثال:

Assignment operator	Sample expression	Explanation	Assigns
Assume: `int c = 3, d = 5, e = 4, f = 6, g = 12;`			
+=	`c += 7`	`c = c + 7`	10 to c
-=	`d -= 4`	`d = d - 4`	1 to d
*=	`e *= 5`	`e = e * 5`	20 to e
/=	`f /= 3`	`f = f / 3`	2 to f
%=	`g %= 9`	`g = g % 9`	3 to g

عملگرهای افزایشی و کاهشی

- عملگرها:
  - افزایشی ++
  - کاهشی --
- برای مقدار متغیر C که قرار است یک واحد افزایش یابد، استفاده از عملگر ++ نسبت به عبارات C=C+1 یا C+=1 ارجحیت دارد

اگر عملگرهای افزایشی یا کاهشی، قبل از متغیر قرار گیرند (پیشوندی)، آنها به ترتیب عملگرهای «افزایشی پیشوندی» یا «کاهشی پیشوندی» شناخته می‌شوند.

اگر عملگرهای افزایشی یا کاهشی، بعد از متغیر قرار گیرند (پسوندی)، آنها به ترتیب عملگرهای «افزایشی پسوندی» یا «کاهشی پسوندی» شناخته می‌شوند.

عملگر افزایشی پیشوندی

- در عملگر افزایشی پیشوندی برای یک متغیر، ابتدا مقدار متغیر یک واحد افزایش یافته و سپس مقدار کل عبارت (شامل عملگر و متغیر)، مقدار جدید متغیر خواهد بود

مثال

```
int NUMBER=5;
++NUMBER ;
```

  - مقدار متغیر NUMBER یک واحد افزایش می‌یابد
  - مقدار جدید متغیر (عدد 6) خواهد بود

- یعنی:

```
int NUMBER=5;
++NUMBER;
printf(NUMBER);
```

یا

```
int NUMBER=5;
printf(++NUMBER);
```

هر دو عدد 6 را در خروجی چاپ می‌کنند و مقدار متغیر NUMBER بعد از آن عدد 6 خواهد بود

عملگر افزایشی پسوندی

- در عملگر افزایشی پسوندی برای یک متغیر، مقدار متغیر یک واحد افزایش یافته ولی مقدار کل عبارت (شامل عملگر و متغیر)، مقدار قبلی (قبل از افزایش) متغیر خواهد بود

مثال

```
int NUMBER=5;
NUMBER++ ;
```

  - مقدار متغیر NUMBER یک واحد افزایش می‌یابد
  - مقدار قبلی متغیر (عدد 5) خواهد بود

- یعنی:

```
int NUMBER=5;
NUMBER++;
printf(NUMBER);
```

خروجی 6 و

```
int NUMBER=5;
printf(NUMBER++);
```

عدد 5 را در خروجی چاپ می‌کنند و مقدار متغیر NUMBER بعد از آن عدد 6 خواهد بود

عملگرهای افزایشی و کاهشی

پسوندی و پیشوندی

Operator	Sample expression	Explanation
++	++a	Increment a by 1, then use the new value of a in the expression in which a resides.
++	a++	Use the current value of a in the expression in which a resides, then increment a by 1.
--	--b	Decrement b by 1, then use the new value of b in the expression in which b resides.
--	b--	Use the current value of b in the expression in which b resides, then decrement b by 1.

عملگر افزایشی

- تفاوت بین عملگر ++ در پیشوندی و پسوندی

```
1  /* Fig. 3.13: fig03_13.c
2     Preincrementing and postincrementing */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     int c; /* define variable */
9
10    /* demonstrate postincrement */
11    c = 5; /* assign 5 to c */
12    printf( "%d\n", c ); /* print 5 */
13    printf( "%d\n", c++ ); /* print 5 then postincrement */
14    printf( "%d\n\n", c ); /* print 6 */
15
16    /* demonstrate preincrement */
17    c = 5; /* assign 5 to c */
18    printf( "%d\n", c ); /* print 5 */
19    printf( "%d\n", ++c ); /* preincrement then print 6 */
20    printf( "%d\n", c ); /* print 6 */
21    return 0; /* indicate program ended successfully */
22 } /* end function main */
```

خروجی

```
5
5
6

5
6
6
```

- در افزایشی پسوندی، مقدار متغیرC، بعد از استفاده در دستور printf، اضافه می شود.
- در افزایشی پیشوندی مقدار متغیر C، ابتدا افزایش داده شده و سپس دستور printf اجرا می شود

عملگرهای افزایشی و کاهشی

- این سه دستور را در نظر بگیرید

```
  passes = passes + 1;
failures = failures + 1;
 student = student + 1;
```

با عملگر انتساب، بصورت زیر نوشته می‌شوند

```
  passes += 1;
failures += 1;
 student += 1;
```

با عملگر افزایشی پیشوندی، بصورت زیر نوشته می‌شوند

```
  ++passes;
++failures;
 ++student;
```

با عملگر افزایشی پسوندی، بصورت زیر نوشته می‌شوند

```
  passes++;
failures++;
 student++;
```

خطا

- در استفاده از عملگرهای افزایشی و کاهشی، تنها یک متغیر ساده می‌تواند بعنوان عملوند استفاده شود.
  - یعنی بکاربردن دستور (X+1)++ یا ++ ++X اشتباه می باشد.

- بین نام متغیر و عملگرهای افزایشی و کاهشی نباید فاصله باشد.

مثال

- عملگرهای افزایشی و کاهشی

```
int a,b,c;

a=b=20;

c = ++a - b++;

printf("%d\n%d\n%d\n",a,b,c);
```

خروجی:

```
21
21
1
```

مثال

- عملگرهای افزایشی و کاهشی

```
int number1,number2,number3;

number1=100;

number2=200;

number3= 1/2 + 1;

printf("%d\n", number3<1 ? ++number1 : number2-- );

printf("%d\n%d\n%d\n",number1,number2,number3);
```

خروجی:

```
200
100
199
1
```

انواع داده در زبان C

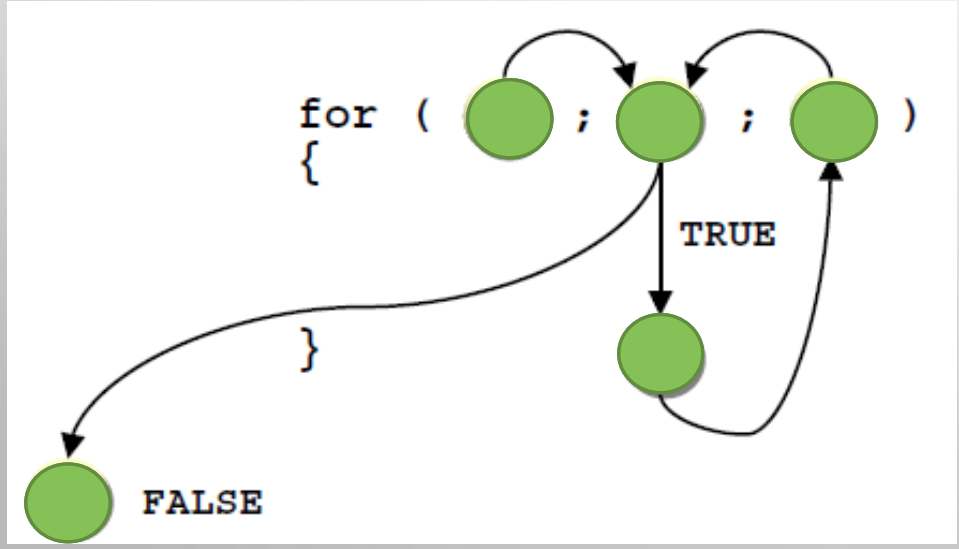
هر متغییری که در زبان C تعریف می شود، می تواند یکی از انواع موجود در این جدول باشد

Variable Type	Keyword	Bytes Required	Range
Character	char	1	-128 to 127
Unsigned character	unsigned char	1	0 to 255
Integer	int	2	-32768 to 32767
Short Integer	short int	2	-32768 to 32767
Long Integer	long int	4	-2,147,483,648 to 2,147,438,647
Unsigned Integer	unsigned int	2	0 to 65535
Unsigned Short integer	unsigned short int	2	0 to 65535
Unsigned Long Integer	unsigned long int	4	0 to 4,294,967,295
Float	float	4	1.2E-38 to
Double	double	8	2.2E-308 to
Long Double	long double	10	3.4E-4932 to 1.1E+4932

حلقه for

ساختار کلی حلقه for

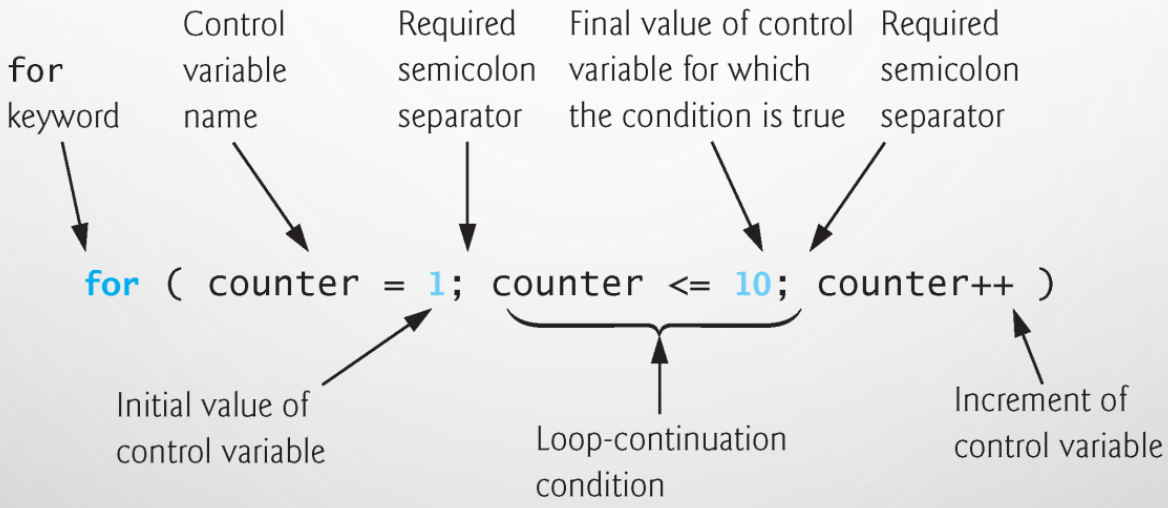
```
for(عبارت 1;عبارت 2;عبارت 3)
{
دستور;
عبارت 1;
}
```



عبارت ۱: مقداردهی اولیه
عبارت ۲: شرط حلقه
عبارت ۳: تغییر مقدار (گام حرکت)

حلقه for

مثال



حلقه for

- هر یک از سه بخش for اختیاری است
  - حذف شرط: همیشه صحیح، تشکیل حلقه بینهایت
  - حذف مقداردهی اولیه: اگر قبل از حلقه انجام شود
  - حذف تغییر مقدار: اگر تغییر مقدار در بدنه انجام شود یا اصلا نیازی نباشد
- برای حذف هر یک از سه بخش: علامت ; نباید حذف شود

```
for(;;)
```

- بعد از پرانتز بسته، علامت ; نداریم
  - در صورت اضافه کردن علامت semicolon پس از پرانتر بسته: معمولا خطای منطقی گرفته می شود
- اگر شرط حلقه از ابتدا نادرست باشد، اصلا دستورات درون حلقه اجرا نمی‌شود.

حلقه for

- هر یک از سه بخش دستور for می‌توانند عبارت ریاضی داشته باشند:

```
for (int i=2*3; i>0+1/2; i+=5-6)
```

اگر: x=2 و y=10 (و در بدنه حلقه تغییر نکنند)

```
for ( j = x; j <= 4 * x * y; j += y / x )
```

معادل است با:

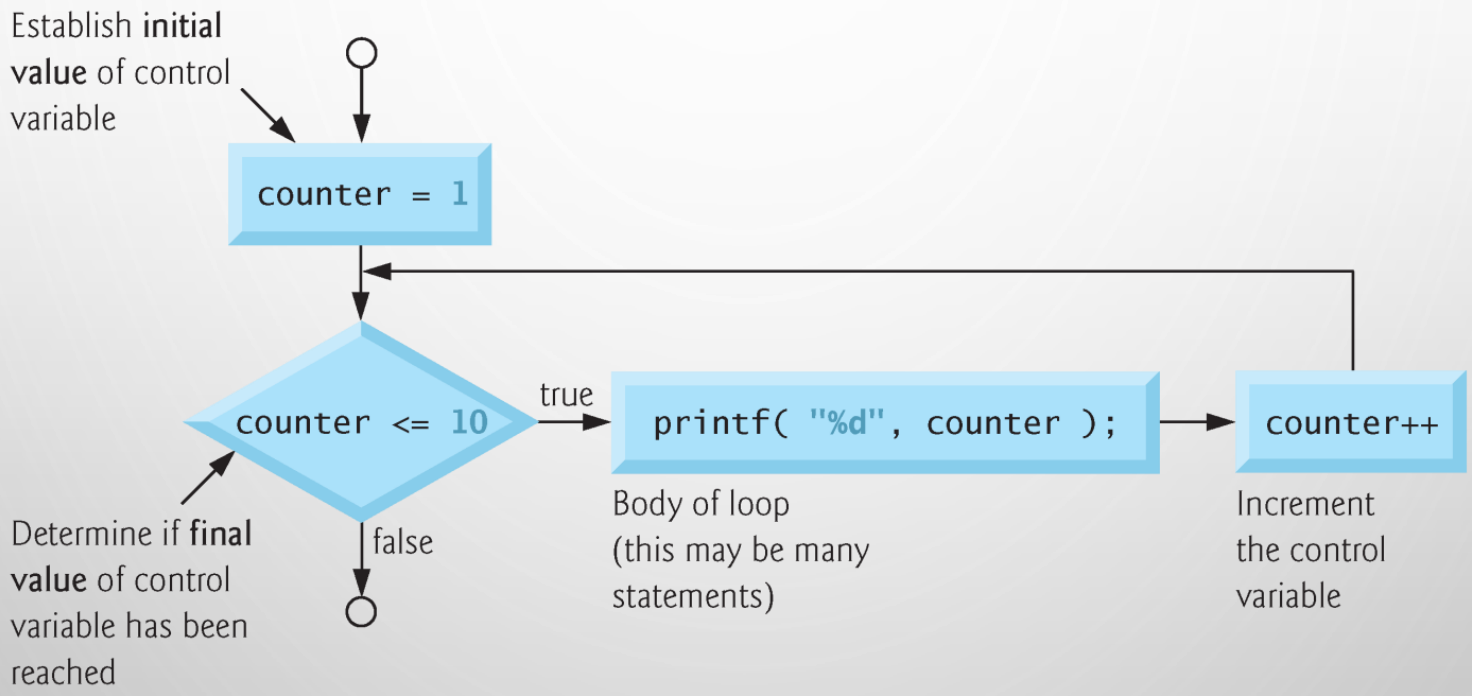
```
for ( j = 2; j <= 80; j += 5 )
```

- برای بخش سوم، همه دستورات زیر معادل هستند

```
counter = counter + 1
counter += 1
++counter
counter++
```

فلوچارت حلقه for

- ساختار فلوچارتی حلقه for



حلقه for، متغیر کنترلی

- متغیر کنترلی از 1 تا 100 هر بار به اندازه یک واحد افزایش می‌یابد

```
for ( i = 1; i <= 100; i++ )
```

- متغیر کنترلی از 100 تا 1 هر بار به اندازه یک واحد کاهش می‌یابد

```
for ( i = 100; i >= 1; i-- )
```

- متغیر کنترلی از 7 تا 77 هر بار به اندازه 7 واحد افزایش می‌یابد

```
for ( i = 7; i <= 77; i += 7 )
```

- متغیر کنترلی از 20 تا 2 هر بار به اندازه 2 واحد کاهش می‌یابد

```
for ( i = 20; i >= 2; i -= 2 )
```

- متغیر کنترلی مطابق با دنباله 2،5،8،11،14،17 (چپ به راست) تغییر می‌یابد

```
for ( j = 2; j <= 17; j += 3 )
```

- متغیر کنترلی مطابق با دنباله 0،11،22،33،44 (چپ به راست) تغییر می‌یابد

```
for ( j = 44; j >= 0; j -= 11 )
```

مثال، حاصلجمع اعداد زوج

- حاصلجمع اعداد زوج 2 تا 100 را محاسبه و چاپ کند

```
1  /* Fig. 4.5: fig04_05.c
2     Summation with for */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main( void )
7  {
8     int sum = 0; /* initialize sum */
9     int number; /* number to be added to sum */
10
11    for ( number = 2; number <= 100; number += 2 ) {
12       sum += number; /* add number to sum */
13    } /* end for */
14
15    printf( "Sum is %d\n", sum ); /* output sum */
16    return 0; /* indicate program ended successfully */
17 } /* end function main */
```

```
Sum is 2550
```

معادل حلقه for بالا:

```
for ( number = 2; number <= 100; sum += number, number += 2 )
   ; /* empty statement */
```

حلقه for، مثال

فردی 1000 دلار در یک حساب بانکی با بهره 5% سرمایه‌گذاری می کند. با این فرض که بهره نیز به اضافه شود، مقدار موجودی این حساب را در پایان هر سال تا 10 سال محاسبه و چاپسپرده کنید.

برای تعیین این مقدار از فرمول زیر استفاده کنید.

$$A = P(1 + R)^N$$

- P=مقدار اولیه سپرده (اصل سرمایه)
- R=نرخ سالانه بهره
- N=تعداد سال‌ها
- A=مقدار موجودی در پایان سال Nام

محاسبه سود، راه حل

- با استفاده از for

```
1  /* Fig. 4.6: fig04_06.c
2     Calculating compound interest */
3  #include <stdio.h>
4  #include <math.h>
5
6  /* function main begins program execution */
7  int main( void )
8  {
9     double amount; /* amount on deposit */
10    double principal = 1000.0; /* starting principal */
11    double rate = .05; /* annual interest rate */
12    int year; /* year counter */
13
14    /* output table column head */
15    printf( "%4s%21s\n", "Year", "Amount on deposit" );
16
17    /* calculate amount on deposit for each of ten years */
18    for ( year = 1; year <= 10; year++ ) {
19
20       /* calculate new amount for specified year */
21       amount = principal * pow( 1.0 + rate, year );
22
23       /* output one table row */
24       printf( "%4d%21.2f\n", year, amount );
25    } /* end for */
26
27    return 0; /* indicate program ended successfully */
28 } /* end function main */
```

خروجی مثال قبل

خروجی به صورت زیر است

```
Year    Amount on deposit
   1              1050.00
   2              1102.50
   3              1157.63
   4              1215.51
   5              1276.28
   6              1340.10
   7              1407.10
   8              1477.46
   9              1551.33
  10              1628.89
```

محاسبه توان

- زبان C عملگر محاسبه توان ندارد
- از تابع زیر استفاده می‌کنیم:

pow(x, y)

- مقدار x را به توان y می‌رساند x^y
  - هر دو عملوند، double هستند

- double مانند float است، ولی متغیر از نوع double می‌تواند اعدادی بزرگتر و با دقت بیشتر از float را ذخیره کند

هدر مورد نیاز

```
#include <math.h>
```

برنامه بدون گنجاندن فایل سرآمد `math.h` کامپایل نخواهد شد

تابع `pow` نیاز به دو آرگومان از نوع `double` دارد

اما `year` یک متغیر از نوع عدد صحیح است

فایل `math.h` شامل اطلاعاتی است که به کامپایلر می گوید تا قبل از اینکه تابع را فراخوانی کند، مقدار `year` را به یک مقدار موقتی از نوع `double` تبدیل کند.

محاسبه سود، اعداد اعشاری

- rate، principle و amount را از نوع double تعریف می کنیم
  - نوع double بخش اعشاری مقادیر را نگه می‌دارد
- مشکل استفاده از انواع اعشاری double و float
  - دو مقدار دلاری زیر:
    - 14.234 که با استفاده از %.2f بصورت 14.23 چاپ می شود
    - 18.673 که با استفاده از %.2f بصورت 18.67 چاپ می شود
- وقتی این دو مقدار با هم جمع می شوند، حاصلجمع آنها عدد 32.907 می‌شود ولی با %.2f بصورت 32.91 چاپ می شود
- فردی که اعداد را بصورتی که چاپ شده مشاهده و جمع می‌کند انتظار دارد بخش اعشاری جمع آنها، 90 شود

```
  14.23
+ 18.67
-------
  32.91
```

نتیجه: double و float برای محاسبات مالی مناسب نیستند

تمرین

- برنامه ای بنویسید که یک عدد مثبت را از ورودی دریافت کرده، تعداد ارقام آن را محاسبه و در خروجی چاپ کند.
- فرض کنید کامپیوتری فاقد توانایی ضرب دو عدد است. برنامه ای بنویسید که با استفاده از جمع‌های متوالی، حاصلضرب دو عدد m و n (دریافت شده از ورودی) را بدست آورده و چاپ کند.
- برنامه ای بنویسید که کلیه اعداد دو رقمی که در آنها صفر وجود ندارد را پیدا کرده و چاپ کند.
- برنامه ای بنویسید که در آن اعداد زوج بین 1 تا 1000 را که بر عدد 27 قابل قسمت می باشند را پیدا کرده و در خروجی چاپ کند.
- برنامه ای بنویسید که تمام اعداد اول کوچکتر از 500 را پیدا کرده و در خروجی چاپ کند.
- برنامه ای بنویسید که یک عدد را از ورودی خوانده و مقسوم علیه های فرد آن را تعیین کند.
- برنامه ای بنویسید که حاصلضرب اعداد بین 2 تا 100 را محاسبه و چاپ کند.

تمرین

- برنامه ای بنویسید که مجموع اعداد زیر را تا 20 جمله حساب کند.

$$s = 1 + \frac{1}{2} + \frac{1}{3} + \cdots$$

- برنامه ای بنویسید که عدد صحیح n را از ورودی دریافت کرده و مجموع زیر را محاسبه و چاپ کند

$$s = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

- برنامه ای بنویسید که دو عدد را از ورودی گرفته و تعداد اعداد زوج بین این دو عدد را چاپ کند.

- برنامه ای بنویسید که عدد n را از ورودی دریافت کرده و سپس n عدد صحیح را از ورودی گرفته و تعداد اعداد مثبت و منفی را چاپ کند.

تمرین

- برنامه ای بنویسید که سن فرد را گرفته:
  - ✓ اگر کمتر از 3 بود، عبارت «خردسال» را چاپ کند
  - ✓ اگر بین 3 تا 9 بود، عبارت «کودک» را را چاپ کند
  - ✓ اگر بین 9 تا 16 بود، عبارت «نوجوان» را چاپ کند
  - ✓ اگر بین 16 تا 30 بود، عبارت «جوان» را چاپ کند
  - ✓ اگر بزرگتر از 30 بود، عبارت «میانسال» را چاپ کند

- برنامه ای بنویسید که دو عدد را از ورودی گرفته و حاصلضرب اعدادی بین این دو که بر 3 بخش پذیر هستند را محاسبه و چاپ کند.

- برنامه ای بنویسید که دو عدد را از ورودی گرفته و حاصلجمع اعدادی که بین این دو هم بر 5 و هم بر 11 بخش پذیر هستند را محاسبه و چاپ کند