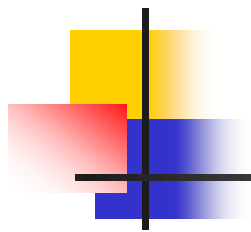


عبارتها



انواع عبارتها در زبان برنامه نویسی

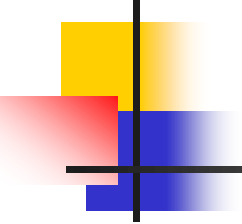
- عبارتهای ریاضی
- عبارتهای شرطی
- عبارتهای منطقی



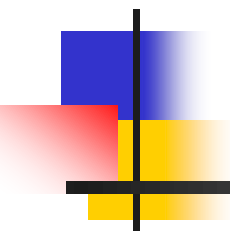
نوع عبارت	اپرندها (عملوندها)	اپراتورها (عملگرها)	خروجی عملیات	مثال
عبارت ریاضی	اعداد و عبارات ریاضی	+ - * / %	یک مقدار عددی	$3*(4+x/5)$
عبارت شرطی	اعداد و عبارات ریاضی	> , >= , < <= , == , !=	یک مقدار منطقی	$x \geq y$ $z == 4$
عبارت منطقی	عبارات شرطی	&& , , !	یک مقدار منطقی	$(x \geq y) \&\& (z == 4)$

مقادیر منطقی و مقادیر عددی

- همانطور که میدانید در کامپیوتر کوچکترین متغیری که میتوان تعریف کرد. متغیرهای یک بایتی هستند
- حاصل یک عملیات شرطی یک مقدار منطقی است.
- در کامپیوتر مقدار منطقی **true** بصورت معادل عددی 1 تعبیر میشود (00000001)
- و مقدار منطقی **false** بصورت معادل عددی 0 تعبیر میشود (00000000)
- از طرف دیگر اگر یک مقدار عددی داشته باشیم و بخواهیم آنرا بعنوان یک مقدار منطقی استفاده کنیم باید توجه داشته باشیم
 - اگر عدد یا حاصل ارزیابی عبارت ریاضی معادل صفر بود یعنی **false**
 - و اگر عدد یا حاصل ارزیابی عبارت ریاضی غیر از صفر بود یعنی **true**



■ در ادامه مباحث درسی در مورد اینکه عدد غیر صفر از نظر منطقی یعنی **true** (صحیح)، بیشتر صحبت خواهیم کرد



عبارات ریاضی

- عبارات C برای توصیف محاسبات استفاده می شوند.
- عبارات شامل عملها و عملوندهای آنها می باشند.
- عملوندها می توانند یک متغیر یا فراخوانی تابع باشند.



عبارتهای ریاضی

■ نتیجه محاسبه عبارات ریاضی یک عدد است!.

■ مثالها:

$$1+2$$

$$(x - 32) * (5/9)$$

$$1 * (2 * (3 * (4 * 5)))$$

عملگرهای ریاضی

جمع	+
تفریق	-
ضرب	*
تقسیم	/
باقیمانده	%

C++ Math Operator Rules

اولویت	Associativity	عملگر
زیاد	چپ به راست	()
متوسط	چپ به راست	* / %
کم	چپ به راست	+ -

- $2 / 3 / 4 + 5$
- $(7 * 3 / 4 - 2) * 5$

مثال: عبارات ریاضی زیر را بصورت عبارات زبان C بنویسید

1) $2x + 6$

■ 1) $2 * x + 6$

2) $3a(2x + 6)$

■ 2) $3 * a * (2 * x + 6)$

3) $\frac{3 + 4a}{6x}$

■ 3) $(3 + 4 * a) / (6 * x)$

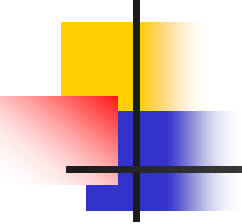
■ 4) $2 * x * ((a - 1) * (3 - 2 * a) + 4 / b)$

4) $2x[(a - 1)(3 - 2a) + 4 / (b)]$



عملگر انتساب

- در زبان C برای نسبت دادن یک مقدار به یک متغیر از عملگر انتساب (=) استفاده میشود.
- در زبان C عملگر انتساب پس از نسبت دادن مقدار موردنظر به متغیر مقدار آنرا به برنامه بر میگردداند. لذا ما میتوانیم چند متغیر را همزمان مقداردهی کنیم:
- $a=b=c=x=100;$
- عملگر انتساب برای تمامی انواع داده‌ای زبان C به شکل یکسان استفاده میشود.

- 
- عملگر انتساب دارای اولویت بسیار کم (کمترین اولویت) هست و از راست به چپ اعمال می شود.

You can do this: ■

```
x = y = z + 15;
```

- Ex1:

- `int x;`
- `x=(200*45)/6;`

- Ex2:

- `float y,z,t;`
- `y=3.6 * 7 + 9.0;`
- `z=t=y;`

عملگرهای انتساب محاسباتی

- در زبان C عملگرهایی برای **کوتاهتر کردن برنامه** و **کاهش زمان اجرا** در نظر گرفته شده است که **یک عملیات محاسباتی** را همراه با **عملیات انتساب** انجام میدهد.

عملگر	توضیح
$+=$	عملگر انتساب جمع
$-=$	عملگر انتساب تفریق
$*=$	عملگر انتساب ضرب
$/=$	عملگر انتساب تقسیم
$\%=$	عملگر انتساب تعیین باقیمانده

عملگر	حالت کلی	حالت ساده شده
$+=$	$sum = sum + x$	$sum += x$
$-=$	$sum = sum - x$	$sum -= x$
$*=$	$sum = sum * x$	$sum *= x$
$/=$	$sum = sum / x$	$sum /= x$
$\% =$	$sum = sum \% x$	$sum \% = x$

مثال: برنامه ای بنویسید که سه عدد را از کاربر گرفته و حاصل جمع آنها

را نشان دهد

```
int main()
{
    int num,sum=0;
    printf( "\nEnter number 1: ");
    scanf( "%d" , &num);
    sum += num;
    printf( "\nEnter number 2: ");
    scanf( "%d" , &num);
    sum += num;
    printf( "\nEnter number 3: ");
    scanf( "%d" , &num);
    sum += num;
    printf( "\nsum is: %d" , sum);
    getch();
    return 0;
}
```



عملگر افزایشی و کاهششی

- عملگرهای $++$ و $--$ میتوانند مقدار یک متغیر را یک واحد افزایش و یا کاهش دهند.
- از آنجا که سرعت اجرای این دستورات سریعتر است لذا باعث افزایش سرعت اجرای برنامه میگردند.
- این عملگرها میتوانند هم قبل از متغیر آورده شوند و هم بعد از نام متغیر

■ $num++$, $num--$, $++num$, $--num$

مثال ۱:

```
int main()
{
    int num=5;
    num++;
    printf( "\nnum:%d ",num);
    num--;
    printf( "\nnum:%d ",num);
    getch();
    return 0;
}
```

خروجی:

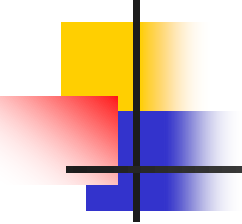
num:6
num:5

مثال ۲:

```
int main()
{
    int num=5;
    printf( "\nnum:%d ", num++);
    num=5;
    printf( "\nnum:%d ", ++num);
    num=5;
    printf( "\nnum:%d ", num--);
    num=5;
    printf( "\nnum:%d ", --num);
    getch();
    return 0;
}
```

خروجی:

```
num:5
num:6
num:5
num:4
```



■ اگر ++ ویا -- قبل از نام متغیر قرار گیرند باعث میشوند که قبل از استفاده از متغیر عملیات افزایش و یا کاهش انجام گیرد

■ اگر ++ ویا -- بعد از نام متغیر قرار گیرند باعث میشوند که بعد از استفاده از متغیر عملیات افزایش و یا کاهش انجام گیرد