

1. 自增运算符

有一个运算符叫自增运算符 符号是 ++



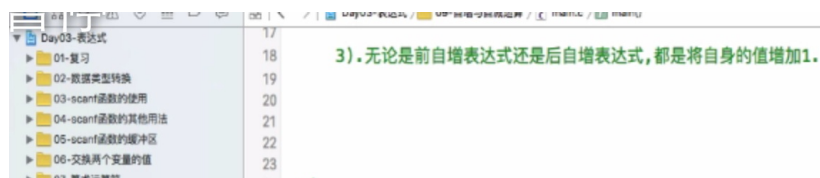
```
1 //
2 1. 自增运算符.
3
4 ++
5
```

2. 自增表达式, 由自增运算符组成的表达式



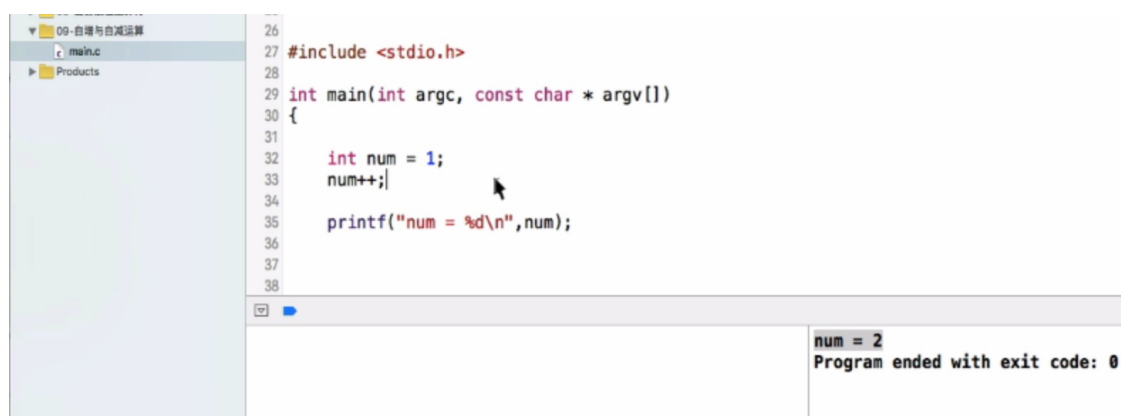
```
6 2. 自增表达式.
7
8 1). 前自增表达式.
9     int num = 1;
10    ++num;
11
12
13 2). 后自增表达式.
14    int num = 1;
15    num++;
16
```

3. 自增表达式的共性



```
17
18 3). 无论是前自增表达式还是后自增表达式, 都是将自身的值增加1.
19
20
21
22
23
```

1> 例子一



```
26
27 #include <stdio.h>
28
29 int main(int argc, const char * argv[])
30 {
31
32     int num = 1;
33     num++;
34
35     printf("num = %d\n", num);
36
37
38
```

num = 2
Program ended with exit code: 0

```
26
27 #include <stdio.h>
28
29 int main(int argc, const char * argv[])
30 {
31
32     int num = 1;
33     ++num;
34
35     printf("num = %d\n", num);
36
37
38
```

num = 2
Program ended with exit code: 0

4. 自增表达式的结果

```
20
21
22
23
24
25
26
```

3. 自增表达式是1个表达式。既然是1个表达式，那么这个自增表达式就一定有1个结果。那么我们就可以使用1个变量把这个表达式的结果存储起来。

1>

```
23
24
25
26
27
```

后自增表达式的结果的计算方式：
先将自身的值取出来作为后自增表达式的结果，然后再将自身的值+1

```
37
38 #include <stdio.h>
39
40 int main(int argc, const char * argv[])
41 {
42
43     int i = 1;
44     int i = i++; //结果是1 结果算完以后，在讲i的值+1
45
46     printf("i = %d\n", i);
47     printf("j = %d\n", j);
48
```

i = 2
j = 1
Program ended with exit code: 0

2>

06-交换两个变量的值

07-算术运算符

08-复合赋值运算符

09-自增与自减运算

main.c

Products

20
27
28
29
30
31
32
33

前自增表达式的结果的计算方式：
先将自身的值+1，然后再将自身的值取出来做为表达式的结果。

01-复习

02-数据类型转换

03-scanf函数的使用

04-scanf函数的其他用法

05-scanf函数的缓冲区

06-交换两个变量的值

07-算术运算符

08-复合赋值运算符

09-自增与自减运算

main.c

Products

39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

```
#include <stdio.h>

int main(int argc, const char * argv[])
{
    int i = 1;
    //int j = i++;//结果是1 结果算完以后,在讲i的值+1

    int j = ++i;

    printf("i = %d\n",i);
    printf("j = %d\n",j);
}
```

i = 2
j = 2
Program ended with exit code: 0

5.复杂练习

1>一个注意点

02-数据类型转换

03-scanf函数的使用

04-scanf函数的其他用法

05-scanf函数的缓冲区

06-交换两个变量的值

07-算术运算符

31
32
33
34
35

4. 自增运算度的优先级比算术运算符的优先级要高。

2>

06-交换两个变量的值

07-算术运算符

08-复合赋值运算符

09-自增与自减运算

main.c

Products

38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```
#include <stdio.h>

int main(int argc, const char * argv[])
{
    int i = 1;
    int j = 2;

    int k = i++ + ++j + i++;

    printf("k = %d\n",k);
}
```

k = 6
Program ended with exit code: 0

3>

01-复习

02-数据类型转换

03-scanf函数的使用

04-scanf函数的其他用法

05-scanf函数的缓冲区

06-交换两个变量的值

07-算术运算符

08-复合赋值运算符

09-自增与自减运算

main.c

Products

```
40
41 int main(int argc, const char * argv[])
42 {
43
44     int i = 1;
45
46
47     int j = i++ + ++i + i++ + ++i + i++;
48     //
49     printf("j = %d\n",j);
50
51
52
53
54
55
56 //     int i = 1;
57 //     int j = 2;
58 //
59 //     int k = i++ + ++j + i++;
60 //
```

Multiple unsequenced modifications to 'j'

j = 17

6.自减运算和自增运算原理相同

01-复习

02-数据类型转换

03-scanf函数的使用

04-scanf函数的其他用法

05-scanf函数的缓冲区

06-交换两个变量的值

07-算术运算符

08-复合赋值运算符

09-自增与自减运算

main.c

Products

```
43
44
45 #include <stdio.h>
46
47 int main(int argc, const char * argv[])
48 {
49
50     int i = 1, j = 2;
51
52     int k = i++ + --j + j++ + --i;
53
54     printf("k = %d\n",k);
55
56
57
58 //     int i = 1;
59 //     --i;
60 //
```

Multiple unsequenced modifications to 'j'

k = 4
Program ended with exit code: 0