

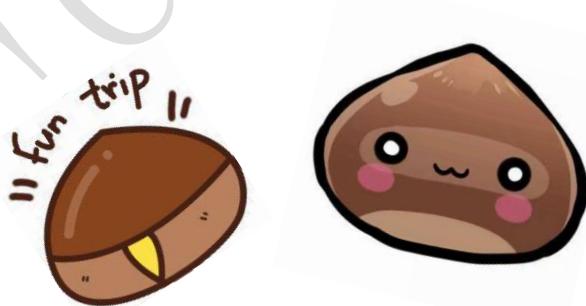
---

- 计算思维和信息学编程·系列课程 -

# YA! C++

## 编 程

### 语 法 学 习 手 册



编者：冯薇

YA!C++ · WAIT



## 目录

第 1 章 C++ 编程入门 .....	1
1.1. 走进 C++ 语法 .....	1
1.1.1. C++ 程序基本结构 .....	1
1.1.2. 程序的 IPO 流程和变量 .....	4
1.2. 运算表达和数据类型 .....	8
1.2.1. 运算符和表达式 .....	8
1.2.2. 数据类型 .....	13
第 2 章 程序三大控制结构 .....	18
2.1. 顺序结构：勇往直前，无一遗漏 .....	18
2.2. 选择结构：有“选择”，想“任性” .....	21
2.2.1. if 结构：“如果”的选择 .....	21
2.2.2. switch 结构：选择“转换器” .....	27
2.3. 循环结构：重复，重复，再重复 .....	31
2.3.1. for 循环：重要的事情做 N 遍 .....	31
2.3.2. while 循环：当重复次数是个谜 .....	35
第 3 章 数组和字符串 .....	44
3.1. 数型数组：数以类聚 .....	44
3.1.1. 一维数组：将“数”排队！ .....	44
3.1.2. 二维/多维数组：将“数”成表！ .....	49
3.2. 字符串：词以群分 .....	53
第 4 章 函数 .....	66

4.1. 返值函数和 void 函数：模块 DIY .....	66
4.2. 递归函数.....	75
第 5 章 结构体和数据的文件存储 .....	78
5.1. 结构体.....	78
5.2. 文件的输入输出.....	82
~我是彩蛋~ .....	87
附录 .....	88
1. 运算符优先级.....	88
2. 数据类型.....	90
3. ASCII 码表 .....	94
4. scanf () 与 printf () 的格式说明符 .....	95
scanf () 独有格式说明符.....	96
printf () 独有格式说明符.....	97
5. 转义字符表.....	98

\*注：

(1) 各章节中【关联题】是洛谷网站 (网址: <https://www.luogu.com.cn/>)  
的对应题号，可登录网站搜索题号，查看题目的细节，并可将完成的程序提交，  
进行在线评测。

(2) 【固定格式】中标有【】的部分，在编程中可省略，或按需增加。



# 第1章 C++ 编程入门

## 1.1. 走进 C++ 语法

### 1.1.1. C++ 程序基本结构

让计算机“说”出你指定的“话”。



#### A. 【000-Hello World】

问题描述：输出特定的文字内容。

```

1 //Hello World程序
2 //my first program
3 #include<iostream>
4 #include<cstdio>
5 using namespace std;
6 int main()
7 {
8     cout<<"Hello C++ World!"<<endl;
9     printf("HELLO C++ WORLD!\n");
10    system("pause");//运行exe时，便于看结果；OJ勿用
11    return 0;
12 }
```

【运行结果】 Hello C++ World!  
 HELLO C++ WORLD!

#### 【说明】

第 1、2 行：是注释行。注释语句只是添加关于程序的说明信息，编译和运行时，不起任何作用。

**【固定格式】注释的两种方式：一是，//单行注释；二是，/\*多行注释\*/。**

第 3、4 行：是对编译器做出预处理指示的语句，用来将程序与所要用到的头文件关联起来。

**【固定格式】#include<头文件名>**

第 5 行：是表明使用标准 (std) 名字空间。

**【固定格式】using namespace std;**

第 6 行：为主函数 (main function) 的声明，它是程序运行时被最先执行的起点。

**【固定格式】int main() { }**

第 7-12 行：{}之间的语句，就是实现程序具体功能的代码语句。每一条语句都必须以英文状态的 “；” 结尾。

第 11 行：是主函数 main() 的返回语句，用来表示程序的结束，它是程序运行时被最后执行的终点。

**【固定格式】return 0;****【知识点 000】C++的基本结构**

```
#include<头文件>

using namespace std;

int main () { //主函数，程序运行的“起点”
    /*----多----行----
     ----注----释----*/
    .....程.....序....;
    .....语.....句....;
    return 0;//主函数返回值，程序运行的“终点”
} //单行注释
```





## B. 【随堂练】000-欢迎卡

```
#####*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#*#
```

Welcome to FZYA Middle School!

【运行结果】 #####\*#\*#\*#\*#\*#\*#\*#\*#\*#\*#\*#\*#



## C. 【随堂想 000】

(1) 以下 C++ 基本结构完整的是：

<pre>a) #include&lt;头文件&gt; using namespace std;  int main( ) {     .....程.....序....;     .....语.....句....;     return 0; }</pre>	<pre>b) /*----多----行----* -----注-----释-----*/ #include&lt;头文件&gt; int main( ) {     .....程.....序....;     .....语.....句....;     return 0; }</pre>
<pre>c) #include&lt;头文件&gt; using namespace std;  int main( ) {     .....程.....序....;     .....语.....句....; } //单行注释</pre>	<pre>d) #include&lt;头文件&gt; using namespace std; { .....程.....序....;     .....语.....句....;     return 0; }</pre>



### 【关联题 000】

- 1) 复制以下程序到 Dev C++, 编译纠错, 使其可正常运行。

```
\n纠错版 “Hello World”

#include<cstdio>

using namespace std;

int main () {

    cout<<"Hello C++!";
    cout<<"Hello World"<<\n;
    cout<<"Hello C++ World"<<endl
    return 0;
}
```

### 1.1.2. 程序的 IPO 流程和变量

让计算机“解”出你作业中的“题”。



### 【知识点 001】0- 编程的 I-P-O 流程

编程的目的是让计算机按照人的要求操作数据, 解决具体问题, 它的标准流程:



\*一切不以“解决问题”为目的的编程, 都是在“耍帅”!



#### A. 【001-三角形面积】

问题描述: 已知三角形的底为 23, 高为 51, 求三角形面积。



```

1 #include<iostream>
2 using namespace std;
3 int main()
4 {
5     int l,h,s;
6     l=23;
7     h=51;
8     s=l*h/2;
9     cout<<s<<endl;
10    return 0;
11 }

```

【运行结果】

【联想】程序的运行结果和手工计算的结果之间，为何存在差异？



### 【知识点 001】1- 变量

变量：是程序运行中数据值可以被改变的数据“容器”。

(1) 定义变量：声明变量的类型和名称，并分配存储空间，指定初始值。

**【固定格式】** 类型关键字 变量名 【=初值】；

**【固定格式】** 类型关键字 变量名 1 【=初值】， 变量名 2 【=初值】， .....

(2) “变量名”通常使用“标识符”来进行命名。



### 【知识点 001】2- 标识符命名规则

(1) 由字母 (a~z, A~Z)、数字 (0 ~ 9)、下划线 ( \_ ) 三类符号组成；

(2) 不能以数字开头，32个有效字符；区分大小写，表示不同标识符；

(3) 不能与 C++ 关键字重名。重要技巧：见名知义。



### B. 【随堂练】001-求三角形面积（升级）

若三角形的底和高为任意值，升级程序，求三角形面积。

**【运行结果】** 输入样例 1#：12 34；输出样例 1#：204

输入样例 2#：56 78；输出样例#：2184



### 【知识点 001】3- 输入输出 cin-cout 语句

键盘输入数据：

**【固定格式】** `cin >> 变量 ;`

**【固定格式】** `cin >> 变量 1 >> 变量 2 >> ..... >> 变量 n ;`

屏幕输出数据：

**【固定格式】** `cout<<表达式;`

**【固定格式】** `cout << 表达式 1(变量 1) << 表达式 2(变量 2) << ..... << 表达式 n (变量 n) ;`

换行的实现：转义字符\n、关键字 endl

C.



### 【随堂想 001】

(1) 以下合法标识符是：，不合法的说明原因。

- |              |              |            |
|--------------|--------------|------------|
| a) ya-12     | a) _Myschool | d) xyz(7)  |
| b) endl      | b) my_family | e) max&min |
| c) my friend | c) 34ya      |            |

(2) 下列 `cin` 输入语句正确的是：

- |            |                    |               |
|------------|--------------------|---------------|
| a) cin a;  | d) cin>>a>>;       | g) cin>>a>>b; |
| b) cin<<a; | e) cin>>a>>endl;   |               |
| c) cin>>a; | f) cin>>" a=" >>a; |               |



(3) 下列 cout 输出语句正确的是:

- a) cout << "Welcome to FZYA Middle School!" ;
- b) cout>>" Welcome to FZYA Middle School!" ;
- c) cout<<' Welcome to FZYA Middle School!' << endl;
- d) cout<<" Welcome to FZYA Middle School!" ;
- e) cout<<" a+b=" << a+b<<;
- f) cout<<" a+b=" <<a+b<<' \n' ;

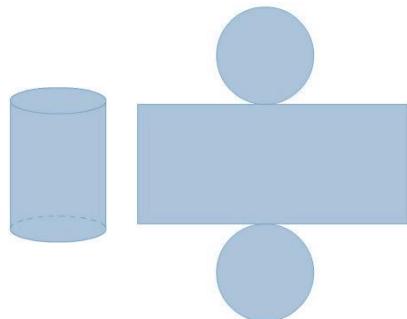


### 【关联题 001】

1) 圆柱面积:

输入高和底面圆半径，求圆柱体面积。

圆柱体面积公式:  $S = S_{\text{上底圆}} + S_{\text{下底圆}} + S_{\text{侧面}}$



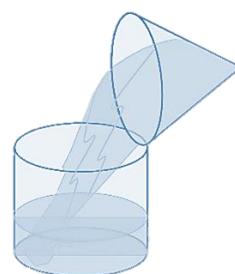
【运行结果】 输入样例#: 12 7;

输出样例#: 835.24 (或 822)

2) 圆锥体积:

输入高和底面圆半径，求圆锥体体积。

圆锥体体积公式:  $V = \frac{1}{3} \times s \times h$



s 表示圆锥底面圆的面积, h 表示圆锥的高。

【运行结果】 输入样例#: 12 7;

输出样例#: 615.44 (或 612)

3) P1001 A+B Problem

## 1.2. 运算表达和数据类型

### 1.2.1. 运算符和表达式

“运算高手”是如此炼成。

A.  【002-球的体、面积】

问题描述： 输入球体半径，用公式，求球体的面积和体积。

$$\text{球面积 } S = \pi D^2; \text{ 球体积 } V = \frac{4}{3} \pi R^3$$

```

1 #include<iostream>
2 #define D 2*r
3 using namespace std;
4 int main(){
5     const float PI=3.141;
6     int r;
7     float s,v;
8     cin>>r;
9     s=PI*D*D;
10    v=4/3.0*PI*r*r*r;
11    cout<<s<<" ";<<v;
12    return 0;
13 }
```

【运行结果】输入样例#：7；输出样例#：615.636；1436.48



#### 【知识点 002】1- 常量

常量：是程序运行中数据值不可以被改变的数据“容器”。

(1) 定义字面常量：

【固定格式】**const** 类型关键字 常量名 1=初值 [, 常量名 2=初值, .....];

字面常量类型：



整型常量：如 12、-34、+5、067（八进制数）、0x89a（十六进制）；

实型常量：如 0.12、3.4、-5.0、+.67、8.9e10；

字符（串）常量：如' \n'（转义字符）、' A'、" bcd"；

逻辑常量：true、false

(2) 定义符号常量：

### 【固定格式】#define 常量名 字符序列

例如：#define A 2+5

int x=A\*A; cout<<x; //x 的值：

## B. 【随堂练】002-海伦公式

问题描述：传说古代的叙拉古国王海伦二世发现的公式，可利用三角形的三边长，求出三角形面积，由此命名为海伦公式。已知三角形的三边长分别为 a、b、c，利用海伦公式求三角形面积。（键盘输入 a、b、c 值时，注意任意两边长度大于第三边）

海伦公式：

$$\begin{array}{c} \text{A} \quad \text{B} \quad \text{C} \\ \diagdown \quad \diagup \\ \text{b} \quad \quad \quad \text{a} \\ \text{---} \\ S = \sqrt{p(p-a)(p-b)(p-c)} \\ p = \frac{a+b+c}{2} \end{array}$$

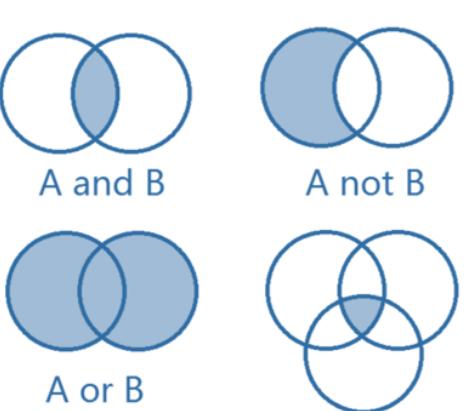


【运行结果】输入样例 1#：3 4 5；输出样例 1#：6

输入样例 2#：3 5 7；输出样例 2#：6.49519



## 【知识点 002】2- 常用运算符和优先级 (完整版参见：附录 1)

运算 类型	运算符	目 向	举例和备注						
数值 运算	加 (+)、减 (-)、乘 (*)、除 (/)、 模 (%) *注：%运算操作数必为整数	双 →	a+b; c%4;						
赋值 运算	=、 +=、 -=、 *=、 /=、 %=  *注 “=”：  <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <th>左侧</th><th>赋值符号</th><th>右侧</th></tr> <tr> <td>变量</td><td>=</td><td>值; 表达式; 变量</td></tr> </table>	左侧	赋值符号	右侧	变量	=	值; 表达式; 变量	双 ←	a+=8; //等同于 a=a+8
左侧	赋值符号	右侧							
变量	=	值; 表达式; 变量							
自增 自减	前置：++变量、--变量  后置：变量++、变量--	单 →	前置：先变后算  后置：先算后变						
关系 运算	<、>、<=、>=、==、!=  返回结果值：True (1) 或 False (0)	双 →	x!=y;  m==0;						
逻辑 运算	非 (!) 与 (&&)、或 (  )  	单 双 →	!T 为 F; !F 为 T; T&&T 为 T; T  T 为 T; T&&F 为 F; T  F 为 T; F&&F 为 F; F  F 为 F;						



	返回结果值: <b>True (非 0) 或 False (0)</b>								
<b>条件运算</b>		表达式 1 ? 表达式 2 : 表达式 3 ; 表达式 1 为要判断的条件命题，之后根据条件命题的判断结果，返回值；判断结果为“真”，返回表达式 2；反之，返回表达式 3。		$\equiv x > y ? x : y$ $\rightarrow y = x > 0 ? "T" : "F"$					
<b>数学函数</b>		abs(x): 整数 x 的绝对值; fabs(x): 实数 x 的绝对值; ceil(x): 向上取整，大于 x 的最小值; floor(x): 向下取整，小于 x 的最大值; sqrt(x): 求实数 x 的平方根; pow(x,y): 求 $x^y$ , 结果为双精度实数; ..... *注: #include<cmath>//需此头文件	abs(-12)=12 fabs(-34.56)=34.56 ceil(-4.3)=-4      ceil(4.3)=5 floor(-4.3)=-5 floor(4.3)=4 sqrt(25)=5 pow(2,3)=8						
<b>运算优先级: (高 → 低)</b>									
后++ 后-- ( )	-负号 ++前 --前 (类型) !	*	+	<< < >= <=	> == !=		&& 	:	= += -= *= /= %= %



### C. 【随堂想 002】

(1) 以下合法赋值语句是:  , 不合法的说明原因。

- |           |          |            |
|-----------|----------|------------|
| a) m=4n   | c) m=2*m | e) xx=y*y  |
| b) -x=y+2 | d) a=b2  | f) x-2=y+5 |

(2) 若 a=12, n=5, 写出下列表达式运算后 a 的值。

- |           |             |
|-----------|-------------|
| 1) a+=a   | 4) a/=a+a   |
| 2) a-=2   | 5) a%=(n%2) |
| 3) a*=2+3 |             |

(3) 执行下列语句后, a、b、c 这 3 个变量的值分别是多少?

```
int a,b,c; a=20; b=++a; c=a++;
```

a= ; b= ; c=

(4) 把下列公式写成程序可识别的表达式。

- |                         |                                   |
|-------------------------|-----------------------------------|
| 1) $y = mx + b$         | 3) $m = \frac{x-yz}{\frac{2}{c}}$ |
| 2) $\frac{2x-y}{x+y^2}$ | 4) $a = \sqrt{(x - 3y)z}$         |

### 【关联题 002】

1) 温度换算: 将华氏温度转为摄氏温度, 公式:  $C=5/9(F-32)$

【运行结果】输入样例 1#: 50; 输出样例 1#: 10

输入样例 2#: 18; 输出样例 2#: -7.77778

2) P1425 小鱼的游泳时间



## 1.2.2. 数据类型

程序里的数据“容器”。



问题描述：分别输入键盘字符和数字，查看转换后的输出结果。

```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     char io_c; int asc;
5     // 输入字符，输出ASCII码
6     cout<<"请输入任一英文符号：" ;
7     cin>>io_c;
8     asc=io_c;
9     cout<<asc<<'\n';
10    // 输入ASCII码，输出字符
11    cout<<"请输入33-127间任一数：" ;
12    cin>>asc;
13    io_c=asc;
14    cout<<io_c<<endl;
15    return 0;
16 }
```

```

请输入键盘中任一英文符号：a
97
请输入33-127之间任一数：66
B

```

【运行结果】输入输出样例#：



问题描述：分别输入字母，进行大、小写转换，并输出转换后 ASCII 码。

```

请输入一个大写字母：A
请输入一个小写字母：b
大小写转换：a, B
a, B的ASCII码：97, 66

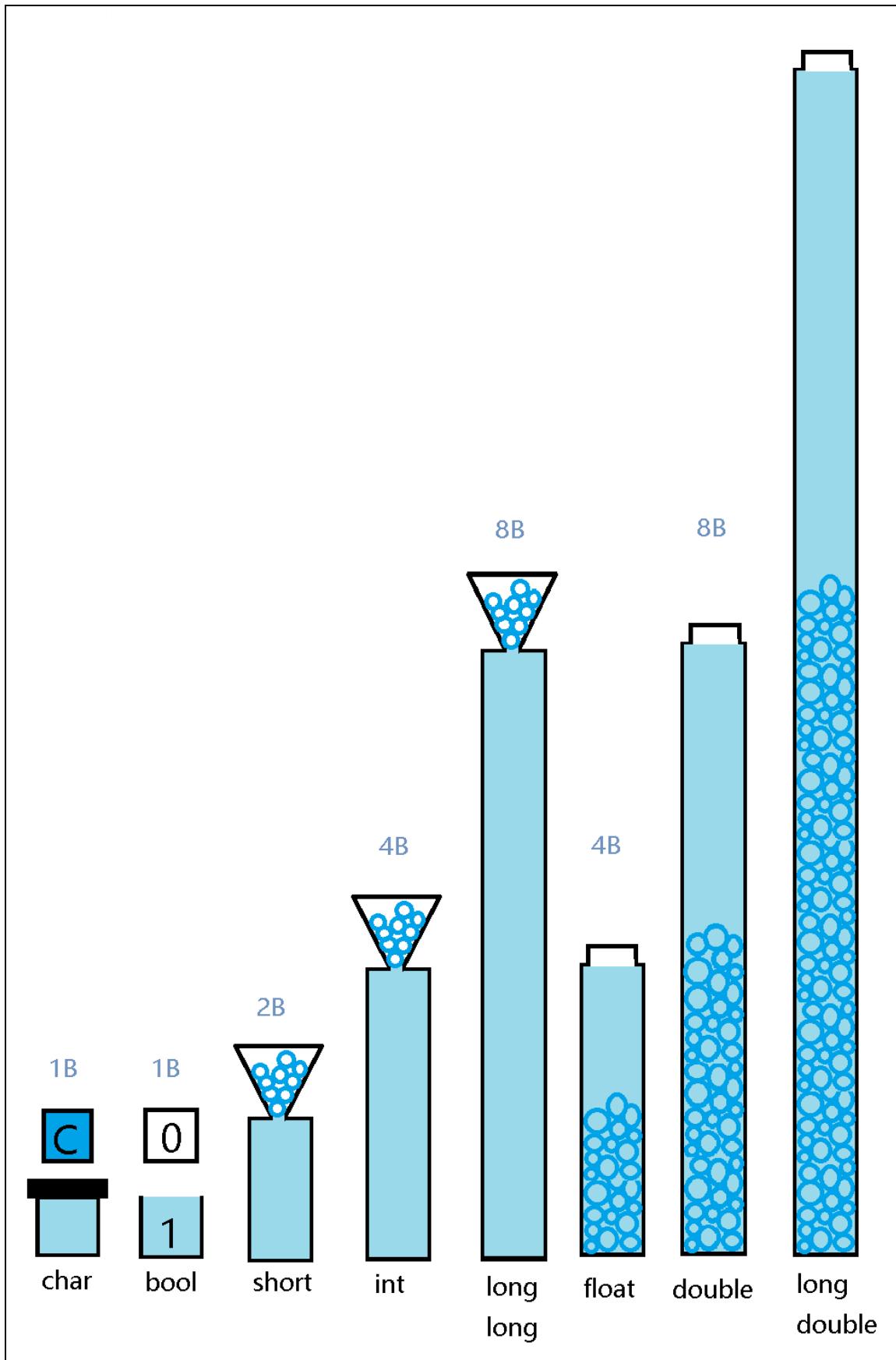
```

【运行结果】输入输出样例#：



【知识点 003】1- 数据类型：“容器”的形状 (扩展参见：附录 2)

字符型	布尔型	整型	实型
• char	• bool	<ul style="list-style-type: none"> <li>• short (int)</li> <li>• (long) int</li> <li>• long long</li> </ul>	<ul style="list-style-type: none"> <li>• float</li> <li>• double</li> <li>• long double</li> </ul>





## 【知识点 003】2- 字符 ASCII 码 (完整版参见：附录 3)

美国标准信息交换代码

## ASCII 表

American Standard Code for Information Interchange

高四位		ASCII 打印字符													
		0010		0011		0100		0101		0110		0111			
		2		3		4		5		6		7			
低四位	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	十进制	字符	Ctrl
0000	0	32		48	0	64	@	80	P	96	`	112	p		
0001	1	33	!	49	1	65	A	81	Q	97	a	113	q		
0010	2	34	"	50	2	66	B	82	R	98	b	114	r		
0011	3	35	#	51	3	67	C	83	S	99	c	115	s		
0100	4	36	\$	52	4	68	D	84	T	100	d	116	t		
0101	5	37	%	53	5	69	E	85	U	101	e	117	u		
0110	6	38	&	54	6	70	F	86	V	102	f	118	v		
0111	7	39	'	55	7	71	G	87	W	103	g	119	w		
1000	8	40	(	56	8	72	H	88	X	104	h	120	x		
1001	9	41	)	57	9	73	I	89	Y	105	i	121	y		
1010	A	42	*	58	:	74	J	90	Z	106	j	122	z		
1011	B	43	+	59	;	75	K	91	[	107	k	123	{		
1100	C	44	,	60	<	76	L	92	\	108	l	124			
1101	D	45	-	61	=	77	M	93	]	109	m	125	}		
1110	E	46	.	62	>	78	N	94	^	110	n	126	~		
1111	F	47	/	63	?	79	O	95	_	111	o	127	□	^Backspace 代码：DEL	

注：表中的ASCII字符可以用“Alt + 小键盘上的数字键”方法输入。

\*注：字符型将依据 ASCII 码表对应的数值进行运算。



### 【知识点 003】2- 类型转换

自动类型转换遵循下面的规则：

(1) 若参与运算的数据类型相同，则运算所得结果的数据类型也为该数据类型。若参与运算的数据类型不同，则先转换成同一类型，然后进行运算。

(2) 转化按数据长度增加的方向进行，以保证精度不降低。例如 int 类型和 long 类型运算时，先把 int 类型转换成 long 类型后再进行运算。

即当参加算数或比较运算的两个操作数类型不统一时，将简单类型将复杂类型转换。

char (short) → int (long) → float → double

(3) 在赋值运算中，赋值号两边数据类型不相同时，将把右边表达式值得类型转换为左边变量的类型。如果右边表达式值的数据类型长度比左边长时，将丢失一部分数据。

(4) 在赋值语句中，赋值号两边数据类型一定是相兼容的类型，如果等号两边数据类型不兼容，语句在编译时会报错。

例：

1) 直接输出运算结果：

```
cout << 10/3.0; //输出结果为浮点型数据： 3.33333
```

2) 赋值运算后输出结果：

```
int ans;  
ans=10/3.0;  
cout << ans;  
  
//输出结果为整型数据： 3
```

强制类型转换：

**【固定格式】(类型关键字) 表达式;**

**【固定格式】(类型关键字) (表达式);**

**【固定格式】类型关键字 (表达式);**





### C. 【随堂想 003】

(1) 求下面表达式的值

```
int a=1;
float b=2.3, c=4.5;
cout<<a%6*(int)(b+c)/7.0;
```

【运行结果】

(2) 若  $a=4$ ,  $b=10$ ,  $c=2$ , 求下面算数表达式的值。

- 1)  $a+b*c/(a+c)\%3/a$
- 2)  $(float)(a+c)/3+(b-a)\%a$

(3) 若  $x=3$ ,  $y=10$ ,  $z=12$  判断下面关系表达式或逻辑表达式的真假。

- 1)  $x-y>y-z$
- 2)  $x<=y \&\&(x>0) \|\ (y>0)$
- 3)  $!(x-y)>0 \|\ (y-z>0)$
- 4)  $x+z==y \|\ z-x<0$



### 【关联题 003】

1) 时间换算：输入秒数，输出 “\*小时\*分钟\*秒”。

【运行结果】输入样例#：3800，输出样例#：1 小时 3 分钟 20 秒

2) P1421 小玉买文具

# 第2章 程序三大控制结构

## 2.1. 顺序结构：勇往直前，无一遗漏

让计算机分辨数的大小。



问题描述： 输入两个整数，辨识两数的大小后，输出结果。

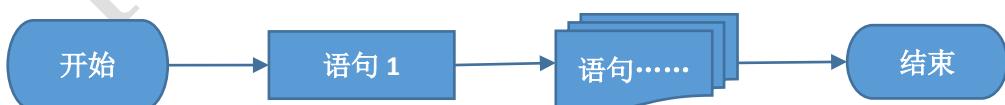
```

1 #include<cstdio>
2 #include<cmath>
3 using namespace std;
4 int main(){
5     int a,b,max,min;
6     scanf("%d%d",&a,&b);
7     max=(a+b+abs(a-b))/2;
8     min=(a+b-abs(a-b))/2;
9     printf("max=%d;min=%d\n",max,min);
10    return 0;
11 }
```

【运行结果】输入输出样例#： 12 34  
max=34,min=12



### 【知识点 004】1-顺序结构模型



### 【知识点 004】2- 格式控制输入输出 (完整格式符参见：附录 4)

(1) 格式化输入函数 `scanf()`

【固定格式】`scanf(“格式控制串”, 变量地址列表);`



**【说明】** “格式控制串”：%【\*宽度】类型标识符，“ ”内**不放提示语或转义符**。变量地址列表：&变量1【&变量2, .....】。

例如：scanf( "%c" , &ch); scanf( "%f,%\*d%3d%4d" , &n1,&n2,&n3);

## (2) 格式化输出函数 printf( )

**【固定格式】** printf ( “**格式控制串**” , 变量列表 ) ;

**【说明】** “格式控制串”：%【修饰标志 宽度.精度】类型标识符，“ ”内可放提示语或转义符，修饰标志包括。变量列表：变量1【变量2, .....】。例如：printf( “输出字符%c” , ch); printf( “a=%+d,b=%-10d,c=%.2f\n” , a,b,c);

\*注：#include <cstdio>//需此头文件，函数原型也在<stdio.h>

\*注：所有转义字符和所对应意义，参见：附录5。



## B. 【随堂练】004-颠倒数

问题描述：从键盘上输入一个三位数，然后将它颠倒输出

**【运行结果】** 输入样例#：678；输出样例#：颠倒数 876



## C. 【随堂想 004】

### (1) 2014 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int a, b, c, d, ans;
5     cin >> a >> b >> c;
6     d = a - b;
7     a = d + c;
8     ans = a * b;
9     cout << "Ans = " << ans << endl;
10    return 0;
11 }
```

输入: 2 3 4      输出: Ans =

(2) 2013 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int a, b;
5     cin >> a >> b;
6     cout << a << "+" << b << "=" << a+b << endl;
7 }
```

输入: 3 5      输出:

(3) 2012 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int a,b,c,d,e,ans;
4 int main() {
5     cin>>a>>b>>c;
6     d=a+b;
7     e=b+c;
8     ans=d+e;
9     cout<<ans<<endl;
10    return 0;
11 }
```

输入: 1 2 5      输出:



【关联题 004】

1) P1909 买铅笔



## 2.2. 选择结构：有“选择”，想“任性”

### 2.2.1. if 结构：“如果”的选择

想要提“条件”!



问题描述：输入一个整数 n，判断它的正或负或零，在屏幕上输出结果。

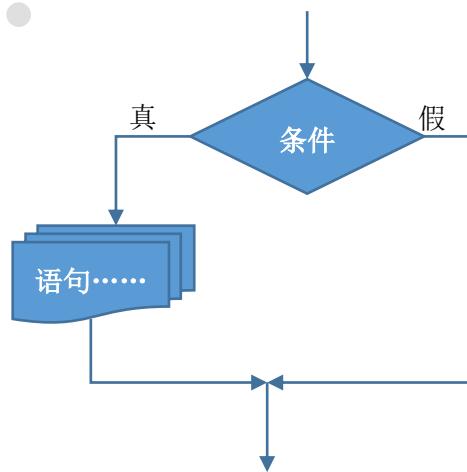
```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     int n;
5     cin>>n;
6     if(n>0)
7         cout<<"正数";
8     if(n==0)
9         cout<<"零";
10    if(n<0)
11        cout<<"负数";
12    return 0;
13 }
```

【运行结果】 输入输出样例#： -123 负数



#### 【知识点 005】 1- 单分支选择结构模型





### 【知识点 005】1.5- if 单分支选择——独一无二

“如果……，那么……。” 的造句

**【固定格式】if (条件表达式) 语句 1;**

**【固定格式】if (条件表达式) {**

语句 1; .....; 语句 n;

}

不再“左右为难”。



#### A. 【005-2.奇偶数】

问题描述：输入一个整数 n，判断它的奇、偶性，在屏幕上输出结果。

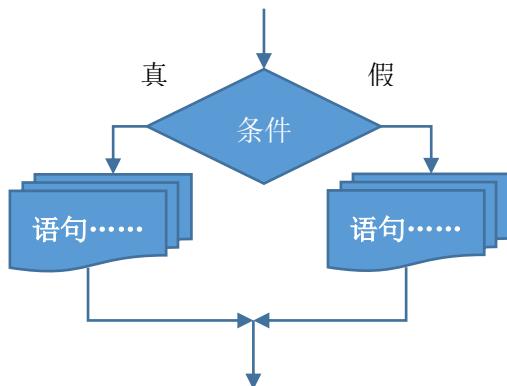
```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     int n;
5     cin>>n;
6     if(n%2==0)
7         cout<<"偶数";
8     else
9         cout<<"奇数";
10    return 0;
11 }
```

【运行结果】输入输出样例#：0 偶数



### 【知识点 005】2- 双分支选择结构模型





### 【知识点 005】2.5- if-else 双分支选择——非此即彼

“如果……，那么……；否则……。” 的造句

**【固定格式】if (条件表达式) 语句 1;**

**else 语句 2;**

**【固定格式】if (条件表达式) {**

**语句 1; .....; 语句 n;**

**}**

**else {**

**语句 1; .....; 语句 n;**

**}**



#### B. 【随堂练】005-1.a 总小于 b

问题描述：输入 a、b 两个数，若 a>b，则交换 a、b 的值，输出两数。

**【运行结果】输入输出样例 1#:** 12 34  
a=12; b=34

**输入输出样例 2#:** 123 45  
a=45; b=123



#### B. 【随堂练】005-2.行李托运费

问题描述：乘坐飞机托运行李时，当乘客行李小于等于 20 公斤时，按每公斤 1.5 元收费，大于 20 公斤时，超重部分按每公斤 2.98 元收费。

输入行李重量，计算行李托运费，并提示“超重”。(结果保留 2 位小数)

**【运行结果】输入输出样例#:** 34.5  
73.21 超重



### 【知识点 005】3- 分支嵌套——专治各种“选择困难症”

形如：

```
if (条件表达式) {  
    if (条件表达式) {  
        .....;  
    }  
}  
else {  
    .....;  
}
```

**【固定格式】** if (条件表达式) {

语句 1; .....; 语句 n; //其中含有 if 或 if-else 结构的语句

}

**【固定格式】** if (条件表达式) {

语句 1; .....; 语句 n; //其中含有 if 或 if-else 结构的语句

}

else {

语句 1; .....; 语句 n; //其中含有 if 或 if-else 结构的语句

}

\*注：在分支嵌套的结构中，**else** 与最邻近的 **if** 形成配对。

\*注：在分支嵌套内外的**条件表达式**之间逻辑关系应合理——“严苛条件‘嵌’  
在内，宽松条件‘套’在外”。

\*注：建议善于利用**缩进**和**{ }**，提高程序的**易读性**——示意相互间的嵌套关系。





## B. 【随堂练】005-3.平闰年

问题描述：输入一个表示年份的数字，判断它是平年，还是闰年。

[rùn nián] 

### 闰年 (历法中的名词)

 编辑  讨论 11

闰年是公历中的名词。闰年分为普通闰年和世纪闰年。

普通闰年：公历年份是4的倍数的，且不是100的倍数，为普通闰年。

世纪闰年：公历年份是整百数的，必须是400的倍数才是世纪闰年

“平闰口诀”：四年一闰，百年不闰，四百再闰。

【运行结果】1900 年平年，1996 年闰年，2000 年闰年



## C. 【随堂想 005】

(1) 复制以下程序到 Dev C++，编译纠错，使其可正常运行。

//输入两个整数，判断它们是否相等，以下程序有 8 个错误。

```
#include<iostream>

using namespace std;

int main(){
    int a;
    scanf("%d",a,b);
    if(a=b);
    {
        printf("yes");
    }
    else ;
}
```

```

{
    printf("no");

}

return 0;

}

```

(2) 将下列描述转换成符合 C++ 语法的条件表达式。

- |   |                               |
|---|-------------------------------|
| 1) x、y、z 均为正数                           | 4) x 小于 160 或 x 大于 180        |
| 2) x 除以 4 的余数为 0, 且 x<br>除以 100 的余数不为 0 | 5) 一个数 x 既是 2 的倍数又<br>是 3 的倍数 |
| 3) x 为偶数                                |                               |

(3) 2015 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int main(){
4     int a, b, c;
5     a = 1;
6     b = 2;
7     c = 3;
8     if(a > b)
9     if(a > c)
10        cout << a << ' ';
11     else
12        cout << b << ' ';
13     cout << c << endl;
14     return 0;
15 }

```

输出:



### 【关联题 005】

1) 运动身高: 输入一个人的身高 H, 判断他适合的运动项目。

“适合运动项目”的规则:



$H \leq 160$  跑步;  $160 < H \leq 175$  羽毛球;  $H > 175$  篮球

188  
【运行结果】输入输出样例#：适合篮球

- 2) 分段函数：输入  $x$  的值，由以下分段规则计算  $y$  值。

$$y = \begin{cases} 1 + x^2 & (x > 0) \\ 0 & (x = 0) \\ 10 + x & (x < 0) \end{cases}$$

【运行结果】输入样例#：7；输出样例#：50

## 2.2.2. switch 结构：选择“转换器”

模拟遥控器的“按键”式选择~

### A. 【006-星期英文】

问题描述：根据键盘上输入的表示星期几的数字，对应输出英文名称。

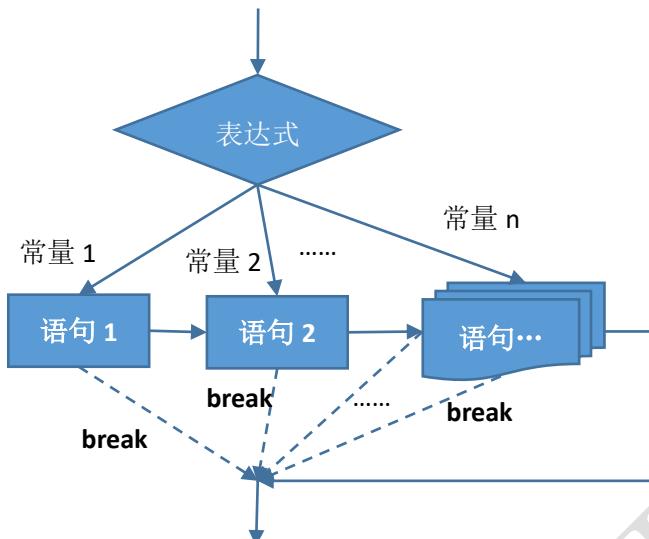
```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     int weekday;
5     cin>>weekday;
6     switch(weekday){
7         case 1:cout<<"Monday"<<endl;break;
8         case 2:cout<<"Tuesday"<<endl;break;
9         case 3:cout<<"Wednesday"<<endl;break;
10        case 4:cout<<"Thursday"<<endl;break;
11        case 5:cout<<"Friday"<<endl;break;
12        case 6:cout<<"Saturday"<<endl;break;
13        case 7:cout<<"Sunday"<<endl;break;
14        default:cout<<"Input Error!"<<endl;break;
15    }
16    return 0;
17 }
```

【运行结果】输入样例#：1；输出样例#：Monday



【知识点 005】4- 多分支选择结构模型



### 【知识点 006】4.5- switch-case 多分支选择——好多的选项

**【固定格式】** switch ( 表达式 ) {  
 case 常量 1 : 语句序列 1; break;  
 case 常量 2 : 语句序列 2; break;  
 .....  
 case 常量 n : 语句序列 n; break;  
 [default : 语句序列 n+1; ]  
 }

\*注：default: 用于解释“未尽事宜”，只能使用一次，可以位于结构中的任何位置，同时也可被省略。

\*注：每一个 case 或 default 分支都提供了一个“入口”，它们使用 break 语句来跳出 switch 结构，如果没有 break 语句，继续执行下面所有分支的语句。



### B. 【随堂练】006-恩格尔系数

问题描述：恩格尔系数是德国统计学家恩格尔在 19 世纪提出的放映一



个国家和地区居民生活水平状况的定律，计算公式为：

$$N = \frac{\text{人均食物支出金额}}{\text{人均总支出金额}} \times 100\% \quad (\text{运算中四舍五入取整})$$

联合国根据恩格尔系数的大小，对世界各国的生活水平有一个划分标准，即一个国家平均家庭恩格尔系数大于等于 60% 为贫穷；50% ~ 60% 为温饱；40% ~ 50% 为小康；30% ~ 40% 为相对富裕；20% ~ 30% 为富裕；20% 以下为极其富裕。

1000 3000  
相对富裕

**【运行结果】输入输出样例#：**



### 【关联题 006】

- 1) 小安买奖品：班长延小安准备将班费  $x$  ( $x > 8$ ) 元钱，用于购买若干奖品奖励给平常服务班级的同学。已知商店里有 3 种奖品，它们的单价分别为 6 元、5 元和 4 元。小安想买尽量多的奖品，同时不想有剩余钱。请编程帮小安制订奖品的方案。

70  
6元笔：1支；5元笔0支；4元笔16支

- 2) 年月天数：输入年份、月份，输出年、月和该月的天数。

2000 2  
2000年2月的天数为29

- 3) 数表读数：柯南又被卷进阿笠博士的游戏，他有一个 9\*9 靶型数表，每个格里都有一个 1 到 5 之间的整数，表示格子的价值，而且，价值的分布很有规律，如下图示。博士任意指定数表中的某行某列格子，柯南说出它的价值。请你为柯南设计一个程序，代他回答博士的问题。

<b>1</b>								
<b>1</b>	<b>2</b>	<b>1</b>						
<b>1</b>	<b>2</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>2</b>	<b>1</b>
<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>4</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>
<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>
<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>4</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>
<b>1</b>	<b>2</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>3</b>	<b>2</b>	<b>1</b>
<b>1</b>	<b>2</b>	<b>1</b>						
<b>1</b>								

输入：i 和 j，用一个空格隔开

输出：第 i 行第 j 列格子的价值

说明：i 和 j 是 1 到 9 之间的整数，表示格子的行列标记

【运行结果】输入样例#：3 4；输出样例#：3

4) P1422 小玉家的电费



## 2.3. 循环结构：重复，重复，再重复

### 2.3.1. for 循环：重要的事情做 N 遍

“给这份爱加上一个期限,我希望是一万年。” —— 《大话西游》



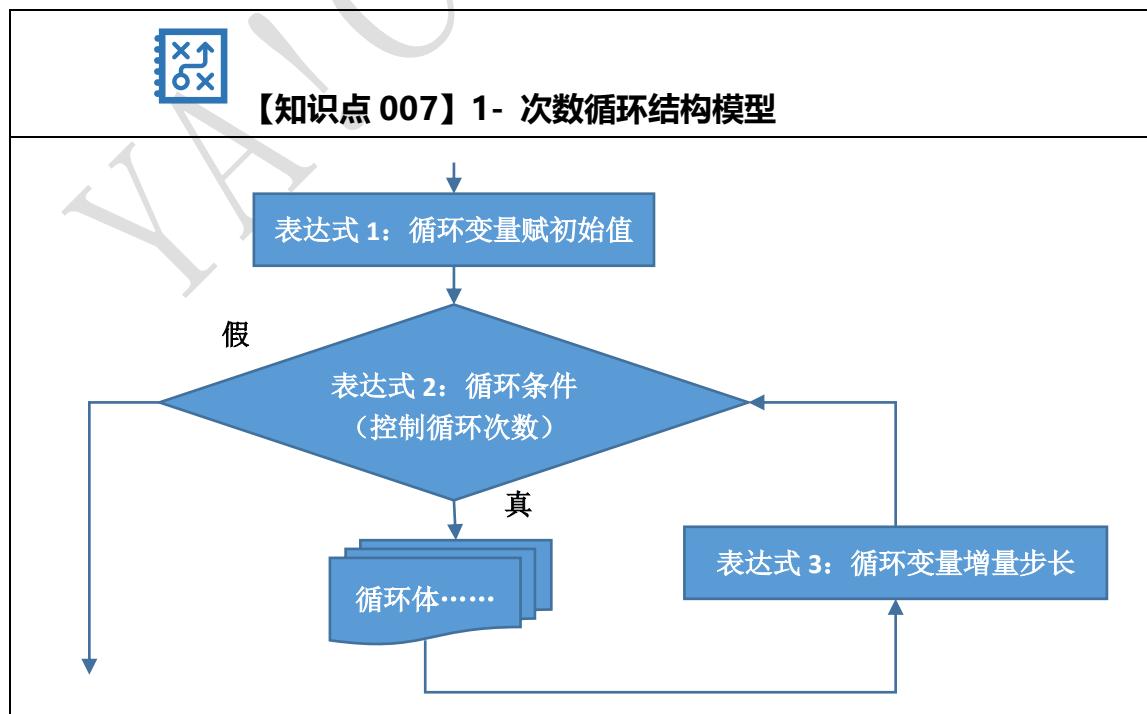
#### A. 【007-阶乘 n!】

问题描述: n 的阶乘公式:  $n! = 1 \times 2 \times 3 \times \dots \times n$ 。

```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     long long s,n;
5     cin>>n;
6     s=1;
7     for(int i=1;i<=n;i++)
8         s=s*i;
9     cout<<s<<endl;
10    return 0;
11 }
```

【运行结果】输入样例#：20；输出样例#：2432902008176640000





### 【知识点 007】1.5- for 循环

**【固定格式】** `for (初始化表达式 1; 条件表达式 2; 变化表达式 3) 语句 1;`

**【固定格式】** `for (初始化表达式 1; 条件表达式 2; 变化表达式 3) {  
 语句 1; .....;  
 语句 n; //循环体;  
}`

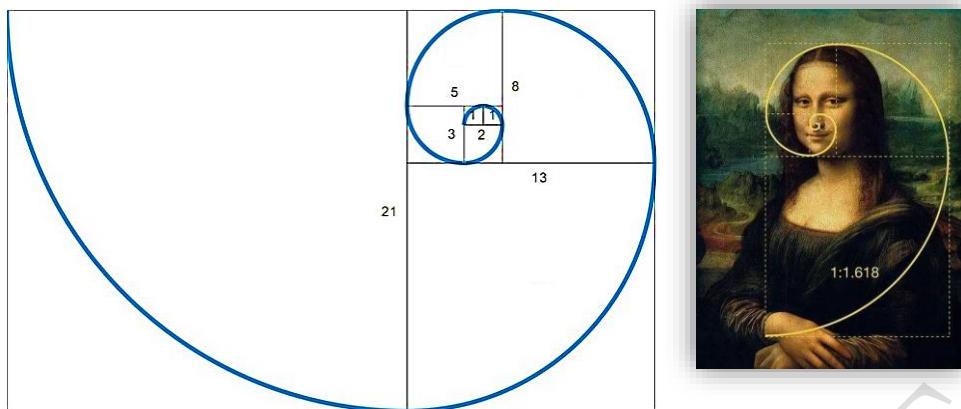
#### B. 【随堂练】007-斐波那契数列

问题描述：

斐波那契数列 (Fibonacci sequence)：又称黄金分割数列、因数学家列昂纳多·斐波那契 (Leonardoda Fibonacci) 用兔子繁殖为例子引入，故也称为“兔子数列”，指的是这样一个数列：0、1、1、2、3、5、8、13、21、34、.....在数学上，斐波纳契数列各项以递推的方法定义： $F(0)=0$ ,  $F(1)=1$ ,  $F(2)=1$ ,  $F(n)=F(n-1)+F(n-2)$  ( $n \geq 2$ ,  $n \in \mathbb{N}^*$ ) 即，最前两项值为 0 和 1，之后的每一项的值是其前两项的相加。斐波纳契数列在现代物理、准晶体结构、化学、艺术等领域，都有直接的应用。

请输出斐波那契数列的第  $n=0$  到 20 各项的数值。(注：第  $n=20$  项：6765)





【运行结果】0, 1, 1, 2, 3, 5, .....4181, 6765,



### C. 【随堂想 007】

(1) 2018 年 NOIP 普及真题-阅读程序 2.

```

1 #include <cstdio>
2 using namespace std;
3 int main() {
4     int x;
5     scanf("%d", &x);
6     int res = 0;
7     for (int i = 0; i < x; ++i) {
8         if (i * i % x == 1) {
9             ++res;
10        }
11    }
12    printf("%d", res);
13    return 0;
14 }
```

输入: 15      输出:

(2) 2013 年 NOIP 普及真题-阅读程序 2.

```

1 #include <iostream>
2 using namespace std;
3 int main(){
4     int a, b, u, i, num;
5     cin>>a>>b>>u; num = 0;
6     for (i = a; i <= b; i++)
7         if ((i % u) == 0)
8             num++;
9     cout<<num<<endl;
10    return 0;
11 }

```

输入: 1 100 15      输出:

(3) 2009 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 using namespace std;
3 const int c=2009;
4 int main(){
5     int n,p,s,i,j,t;
6     cin >> n >> p;
7     s=0;t=1;
8     for(i=1;i<=n;i++){
9         t=t*p%c;
10        for(j=1;j<=i;j++)
11            s=(s+t)%c;
12    }
13    cout << s << endl;
14    return 0;
15 }

```

输入: 11 2      输出:



**【关联题 007】**

- 1) 百以内奇数和: 求  $1+3+5+\dots+99$  的和。【运行结果】输出#: 2500



2) 百内整除计数: 输入 N, 输出 1 ~ 100 间能被 N 整除的数, 并统计个数。

【运行结果】输入输出样例#:

7
7 14 21 28 35 42 49 56 63 70 77 84 91 98
14

3) 趣味运算:

a) 输入 n, 计算  $1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{4} + \frac{1}{5} \dots \dots \frac{1}{n}$

【运行结果】输入输出样例 1#: 50  
0.683247

输入输出样例 2#: 100  
0.688172

b) 计算  $1 \times 2 \times 3 \times \dots \times 50 + 51 + 52 + \dots + 100$  的结果。

【运行结果】输出#: 3.04141e+064

c) 计算  $1 \times 3 + 2 \times 4 + 3 \times 5 + \dots + 10 \times 12$  的结果。【运行结果】输出#:

495

4) 小球下落: 一个小球从 200 米高处自由下落, 每次落地反弹至原高度的一半, 然后在下落, 编程计算小球从开始下落到第十次落地时共经过了多少路程。【运行结果】输出#: 599.219

5) P1035 级数求和

6) P1980 计数问题

### 2.3.2. while 循环: 当重复次数是个谜

上邪，我欲与君相知，长命无绝衰。

山无陵，江水为竭，冬雷震震，夏雨雪，天地合，乃敢与君绝。

——《乐府诗集·上邪》汉·佚名



问题描述：利用辗转相除法，求两个正整数 m、n 的最大公约数。

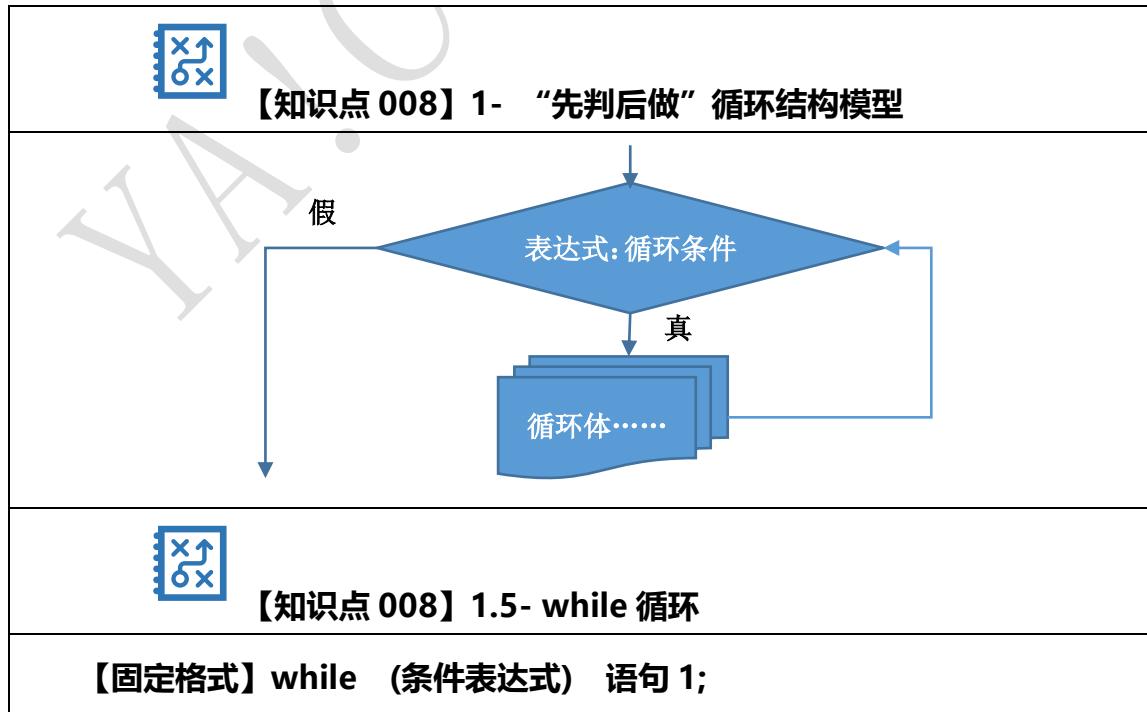
```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     int m,n,r;
5     cin>>m>>n;
6     r=m%n;
7     while(r!=0){
8         m=n;
9         n=r;
10        r=m%n;
11    }
12    cout<<"最大公约数"<<n;
13    return 0;
14 }
```

【运行结果】输入输出样例#:  
34 119  
最大公约数17

辗转相除法，又名欧几里德算法 (Euclidean algorithm)，是求最大公约数的一种方法。它的过程是：用较大数除以较小数，再用出现的余数（第一余数）去除除数，再用出现的第二余数去除第一余数，如此反复，直到最后余数是 0 为止。

如果是求两个数的最大公约数，那么，最后的除数就是这两个数的最大公约数。



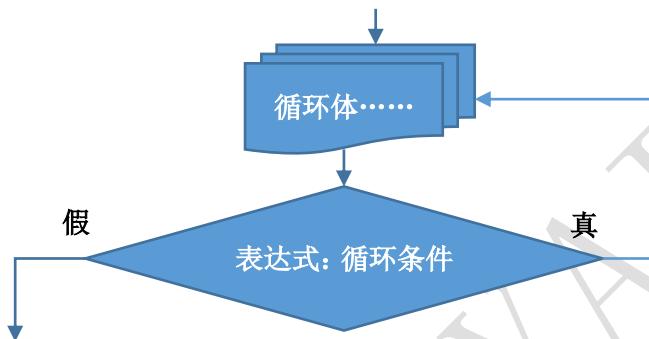
**【固定格式】while (条件表达式) {**

语句 1; .....; 语句 n; //循环体;

}



**【知识点 008】2- “先做后判” 循环结构模型**



**【知识点 008】2.5- do-while 循环**

**【固定格式】do**

语句 1;

while (条件表达式);

**【固定格式】do{**

语句 1; .....; 语句 n; //循环体;

} while (条件表达式);



**B. 【随堂练】008-二进制位数**

问题描述：输入任意十进制数，输出其对应二进制的数位数。或用“do-while”循环结构实现。

**【运行结果】输入样式 1#：45；输出样式 1#：6**

输入样例 2#：27；输出样例 2#：5

将十进制数45转换成二进制数

2	45	余数	结果
2	22	1	
2	11	0	
2	5	1	
2	2	1	
2	1	0	
	0	1	

结果： $(45)_{10} = (101101)_2$



### 【知识点 008】3- 循环嵌套

形如：

```
for (初始化表达式 1; 条件表达式 2; 变化表达式 3) {  
    while (条件表达式) {  
        .....;  
    }  
}
```

【固定格式】**for (初始化表达式 1; 条件表达式 2; 变化表达式 3) {**

**语句 1; .....; 语句 n; //含有 for 或 while 或 do-while 结构**  
}

【固定格式】**while (条件表达式) {**

**语句 1; .....; 语句 n; //含有 for 或 while 或 do-while 结构**  
}

【固定格式】**do{**

**语句 1; .....; 语句 n; //含有 for 或 while 或 do-while 结构**  
} **while (条件表达式);**

\*注：建议善于利用缩进和 {}，提高程序的易读性——示意相互间的嵌套关系。

\*注：“循环条件表达式”的细节处理应注意实际的循环边界。

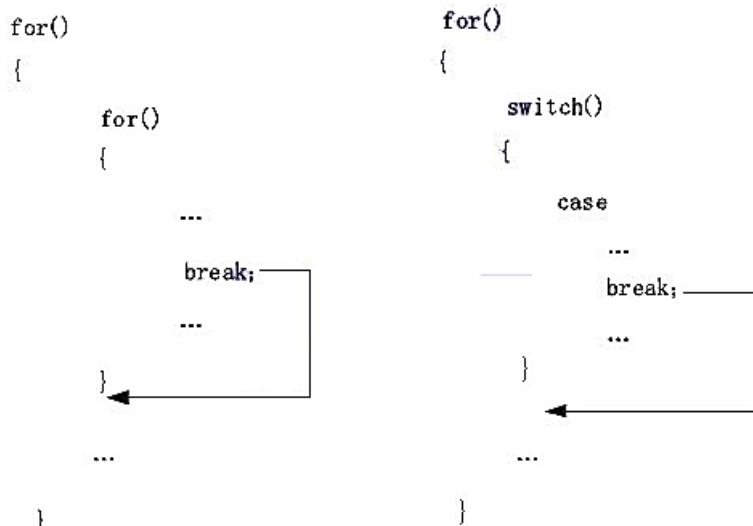


### 【知识点 008】4- 无效循环 vs 死循环 和 break vs continue

**无效循环：**循环条件永远无法达到，因而永远无法进入循环体。

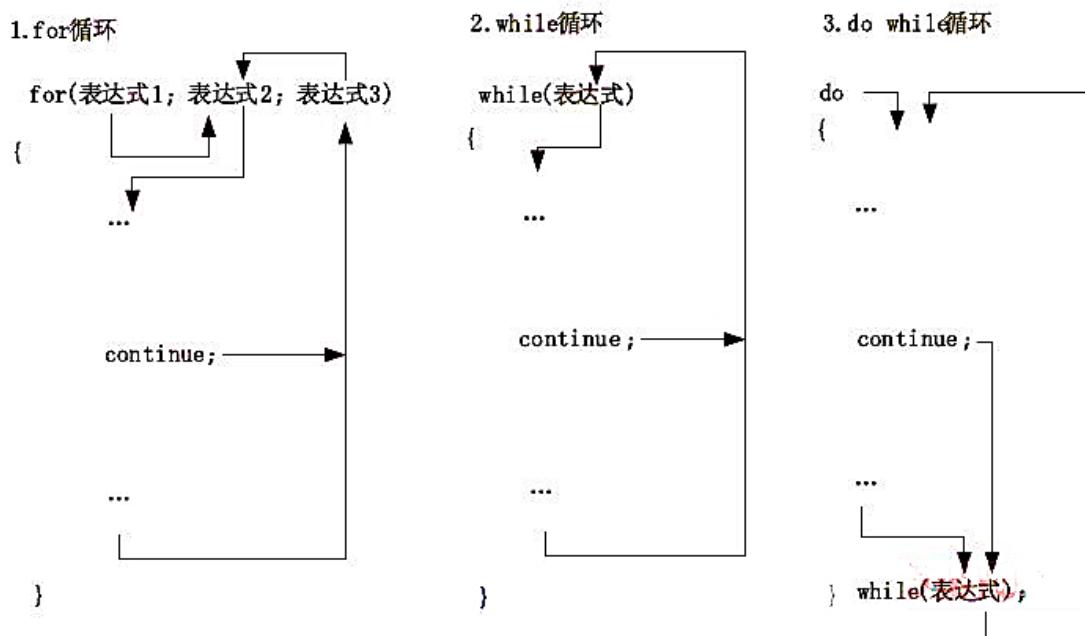
**死循环：**循环条件永远成立（或为非零常数），因而永远无法自动退出循环体。

(1) **break:** 强行完全终止本层循环，专治“各种停不下来”！



注：**break** 仅对循环和 switch 有效，对 if 无效。

(2) **continue:** 仅强行跳出（中断）本轮循环，进入下一轮。



注：**continue** 仅对循环有效。

C.  【随堂想 008】

(1) 2016 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int max, min, sum, count = 0;
5     int tmp;
6     cin >> tmp;
7     if (tmp == 0)
8         return 0;
9     max = min = sum = tmp; count++;
10    while (tmp != 0) {
11        cin >> tmp;
12        if (tmp != 0) {
13            sum += tmp;
14            count++;
15            if (tmp > max)
16                max = tmp;
17            if (tmp < min)
18                min = tmp;
19        }
20    }
21    cout << max << "," << min << "," << sum / count << endl;
22    return 0;
23 }

```

输入： 1 2 3 4 5 6 0 7

输出：

## (2) 2016 年 NOIP 普及真题-阅读程序 2.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int i = 100, x = 0, y = 0;
5     while (i > 0) {
6         i--;
7         x = i % 8;
8         if (x == 1) y++;
9     }
10    cout << y << endl;
11    return 0;
12 }

```

输出：

## (3) 2011 年 NOIP 普及真题-阅读程序 1.

```

1 #include<iostream>
2 using namespace std;
3 int main()
4 {
5     int i,n,m,ans;
6     cin>>n>>m;
7     i=n;
8     ans=0;
9     while(i<=m){
10        ans+=i;
11        i++;
12    }
13    cout<<ans<<endl;
14    return 0;
15 }

```

输入： 10 20 输出：



### 【关联题 008】

- 1) 读数判符号: 读一组实数, 遇 0 结束, 输出与第一数符号相同的所有数。

【运行结果】输入样例#: -1 2 3 -4 5 6 -7 -8 -9 0

输出样例#: -4 -7 -8 -9

- 2) 折纸: 假设纸的长度足够长, 厚度为 0.1 毫米, 对折一次厚度增加 1 倍,

现在对折纸张, 直到总厚度超过珠穆朗玛峰的高度 (8848.13 米) 为止,

编程求纸张对折的次数。【运行结果】输出#: 27

- 3) 特征数

- a) 两位同构数: 指一个两位数, 它平方数的后两位, 恰好等于它本身,

如:  $25^2 = 625$ , 编程输出所有的两位同构数。

【运行结果】输出#: 25, 76

- b) 水仙花数: 指一个三位数, 它的三个位置上数字的立方和, 等于它

本身, 如:  $153 = 1^3 + 5^3 + 3^3$ , 编程输出所有水仙花数。

【运行结果】输出#: 153, 370, 371, 407

- c) 玫瑰花数: 指一个四位数, 它的四个位置上数字的 4 次方和, 等于

它本身, 如:  $1634 = 1^4 + 6^4 + 3^4 + 4^4$ , 编程输出所有玫瑰花数。

【运行结果】输出#: 1634, 8280, 9474

- d) 完全数: 指一个数的所有小于本身的正约数之和等于它本身, 如:

28 的约数为 1、2、4、7 和 14,  $28=1+2+4+7+14$ , 编程输出 3 ~

1000 之间所有的完全数。【运行结果】输出#: 6, 28, 496

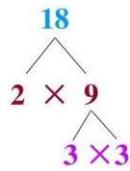
- 4) 分解质因数: 一个正整数 n, 把它分解成质因数 (即所有因数均为质数)

相乘的形式。例如:  $19=1\times 19$ ,  $36=1\times 2\times 2\times 3\times 3$ 。【提示】设因数为 i,

从 2 开始, 当  $i \leq n$  时, 重复以下操作: 先让  $n$  重复被  $i$  除, 若能被整除, 则用商代替  $n$ ,  $i$  为  $n$  的一个因子; 若不能整除, 则将  $i$  增大。

**【运行结果】** 输入样例 1#: 9 ; 输出样例 1#: 19=1\*19

输入样例 2#: 36 ; 输出样例 2#: 36=1\*2\*2\*3\*3



$$18 = 2 \times 3 \times 3$$

- 5) 邮箱密码: 延小安把自己的邮箱密码给忘了, 印象中密码是一个 5 位数, 中间一位(百位)是 1, 并且小安有两个纪念日是 8 月 1 日和 9 月 1 日, 因此常喜欢把 81 和 91 的共同倍数作为密码。请编程帮小安找回密码。

**【运行结果】** 输出#: 22113

- 6) 素数判断: 输入一个数, 判断该数是否为素数。

**【运行结果】** 输入输出样例#: 13  
是素数

- 7) 打印图形:



**【运行结果】** 输出#:

- 8) 切割钢筋: 有一根长 600cm 的钢筋, 需要截成长度为 69cm、39cm、29cm 三种规格的短料, 在三种规格都至少截 1 个单位的前提下, 编程求余料最少的截取方案。

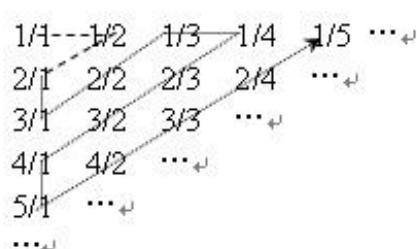
截取方案:  
69cm 钢筋: 6 根;  
39cm 钢筋: 4 根;  
29cm 钢筋: 1 根;

**【运行结果】** 输出#: 最少余材: 1cm。



9) Cantor 表: 现代数学的著名证明之一是 Georg Cantor 证明了有理数是可枚举的。他用下面这张表来证明这一命题。

1/1	1/2	1/3	1/4	1/5	...
2/1	2/2	2/3	2/4	2/5	...
3/1	3/2	3/3	3/4	3/5	...
4/1	4/2	4/3	4/4	4/5	...
5/1	5/2	5/3	5/4	5/5	...
...	...	...	...	...	...



输入整数 n ( $1 \leq n \leq 10000000$ )；输出对应表中的第 n 项

【运行结果】输入样例#：n=7 ；输出样例#：1/4

10) 出租车计费：在 YA 市出租车计费方案是：起步价 3 公里之内 8 元；超过 3 公里后，15.1 公里内，每 550 米计费 1 元；而超过 15.1 公里之后，每 370 米计费 1 元。这样，乘车到里程越远，花费越高。例如，行驶 30 公里，需付费 70 元。一位打车达人分享经验说，若中途下车（即中停结帐）再重新打车，会减少花费。求最少付费的中停结帐位置和总花费。

输入：总里程数

输出：中停结账地点（距起点路程） 和 最少付费值

30  
路程最少付费：59.64元  
中途下车1次

【运行结果】输入样例#：中途第1次下车点15.10公里

11) P1423 小玉在游泳

12) P1424 小鱼的航程(改进版)

# 第3章 数组和字符串

## 3.1. 数型数组：数以类聚

### 3.1.1. 一维数组：将“数”排队！

若将“数”连成“串”，编号。



问题描述：按从大到小逆序输出斐波那契数列（该数列详情参见 B007）

的前 40 项。（参考升级：输入 n，输出逆序数列的第 n 项数。）

```

1 #include<iostream>
2 #include<iomanip>
3 using namespace std;
4 int main(){
5     int fibo[40]={0,1};
6     int i,n;
7     //生成斐波那契数列
8     for(i=2;i<40;i++){
9         fibo[i]=fibo[i-1]+fibo[i-2];
10    }
11    //逆序输出，调整输出样式
12    for(i=39;i>=0;i--){
13        cout<<setw(10)<<fibo[i];
14        if(i%5==0) cout<<endl;
15    }
16    return 0;
17 }
```

【运行结果】输出#：

63245986	39088169	24157817	14930352	9227465
5702887	3524578	2178309	1346269	832040
514229	317811	196418	121393	75025
46368	28657	17711	10946	6765
4181	2584	1597	987	610
377	233	144	89	55
34	21	13	8	5
3	2	1	1	0

参考升级输入样例 1#：20；输出样例 1#：6765

参考升级输入样例 2#：12；输出样例 1#：317811





## 【知识点 009】一维数组

**数组 (Array)**, 是有序、同类型的变量元素的序列或集合。它们被存储在内存中一段连续的存储空间内。每一数组都有一个唯一的名称，通过在名称后面加索引号 (index, 也称, 下标) 的方式可以引用它的某一个指定元素。

**一维数组**, 是用单项索引号对变量进行编号的数组。例如,

索引	0	1	2	3	4
int Array[5]	12	34	56	78	90

### (1) 定义和初始化

#### 【固定格式】

**类型关键字 一维数组名[元素总数] 【={初值 0, 初值 1, ..... , 初值 n}】 ;**

在定义数组时, 若未对数组赋初始值, 那么各元素的值将是内存中的不定值。

若定义数组的同时对其赋初始值, 可以整体赋值, 也可部分赋值。部分赋值时, 其余元素自动赋为 0 (或空)。若花括号{}内为空, 则数组整体赋为 0 (或空)。

### (2) 赋值和调用

对数组的整体赋值, 只在定义数组时进行。调用时, 只能单个元素逐一赋值。

在调用特定数组元素时, 其下标 (索引号) 从 0 开始, 即一个元素总数为 n 个的数组, 它各个元素的下标范围为从 0 到 n-1。若下标值在这个范围以外的情况, 视为数组下标越界。

#### 【固定格式】一维数组名 [下标]

\*注: 数组变量的下标, 可以用**常值、变量或表达式**等方式来表示。

\*注: 一维数组定义或作为函数形式参数描述时, 允许[]内为空。若[]内为空, 则必须为数组整体赋初始值, 且数组元素的总数将由初始值的个数决定。



## B. 【随堂练】009-本年第几天

问题描述： 输入年、月、日，输出这天是本年的第几天。

【运行结果】输入样例 1#： 2019 5 26 ；输出样例 1#： 146

输入样例 2#： 2000 3 3；输出样例 2#： 63



## C. 【随堂想 009】

(1) 2016 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int a[6] = {1, 2, 3, 4, 5, 6};
5     int pi = 0; int pj = 5; int t, i;
6     while (pi < pj) {
7         t = a[pi];
8         a[pi] = a[pj];
9         a[pj] = t;
10        pi++;
11        pj--;
12    }
13    for (i = 0; i < 6; i++)
14        cout << a[i] << ",";
15    cout << endl;
16    return 0;
17 }
```

输出：

(2) 2013 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 using namespace std;
3 int main(){
4     const int SIZE = 100;
5     int n, f, i, left, right, middle, a[SIZE];
6     cin>>n>>f;
7     for (i = 1; i <= n; i++)
8         cin>>a[i]; left = 1;
9     right = n;
10    do {
11        middle = (left+right) / 2;
12        if (f <= a[middle])
13            right = middle;
14        else
15            left = middle + 1;
16    } while (left < right);
17    cout<<left<<endl;
18    return 0;
19 }
```



输入: 12 17

2 4 6 9 11 15 17 18 19 20 21 25

输出:

(3) 2009年NOIP普及真题-阅读程序2.

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int a[3], b[3];
6     int i, j, tmp;
7     for (i=0; i<3; i++)
8         cin >> b[i];
9     for (i=0; i<3; i++)
10    {
11        a[i]=0;
12        for (j=0; j<=i; j++)
13        {
14            a[i]+=b[j];
15            b[a[i]%3]+=a[j];
16        }
17        tmp=1;
18        for (i=0; i<3; i++)
19        {
20            a[i]%=10;
21            b[i]%=10;
22            tmp*=a[i]+b[i];
23        }
24        cout << tmp << endl;
25    }
26 }
```

输入: 2 3 5

输出:

(4) 2008年NOIP普及真题-阅读程序1.

```

1 #include<iostream>
2 using namespace std;
3 int main()
4 {
5     int i, a, b, c, d, f[4];
6     for(i = 0; i < 4; i++)
7         cin >> f[i];
8     a = f[0] + f[1] + f[2] + f[3];
9     a = a / f[0];
10    b = f[0] + f[2] + f[3];
11    b = b / a;
12    c = (b * f[1] + a) / f[2];
13    d = f[(b / c) % 4];
14    if(f[(a + b + c + d) % 4] > f[2])
15        cout << a + b << endl;
16    else
17        cout << c + d << endl;
18 }
```

输入: 9 19 29 39

输出:



### 【关联题 009】

1) 循环数列：一列数 3, 7, 2, 5, 1, 3, 7, 2, 5, 1.....中，第 1992

个数是几？这 1992 个数的总和是多少？【运行结果】输出#：7 7174

2) 十数排序：输入 10 个正整数，把这十个数按由小到大的顺序排列。

【运行结果】输入样例#：0 9 1 8 2 7 36 45 123 45；

输出样例#：0 1 2 7 8 9 36 45 45 123

3) 百门开关：旅馆里有 100 个房间，从 1 到 100 编了号，第一个服务员

把所有的房间门都打开了，第二个服务员把所有编号是 2 的倍数的房间

“反向处理”，第三个服务员把所有编号是 3 的倍数的房间 “反向处

理”，……，以此类推。问第 100 个服务员来过后，哪几扇门是打开的？

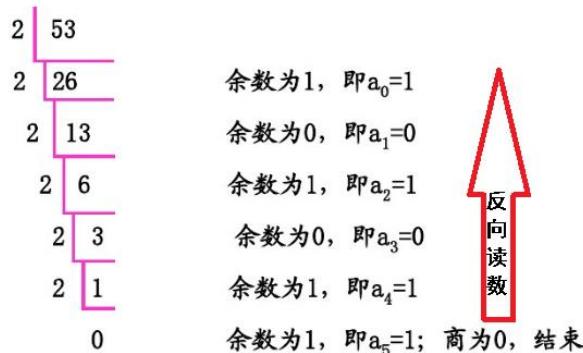
(注：“反向处理”是：原来开着的门关上，原来关着的门打开)

【运行结果】输出#：1 4 9 16 25 36 49 64 81 100

4) 十转二进制：输入一个不大于 32767 的正整数 N，将它转换成对应的二

进制数。

➤ 整数部分：除以2取余，商零即止。



最后结果为： $(53)_{10} = (a_5a_4a_3a_2a_1a_0)_2 = (110101)_2$

【运行结果】输入样例 1#：53；输出样例 1#：110101

输入样例 2#：100；输出样例 2#：1100100



- 5) 约瑟夫问题: M 个人围成一圈, 从第一个人开始报数, 数到 N 的人出圈;  
再由下一个人开始报数, 数到 N 的人出圈; .....。输出依次出圈人的编号。 $M, N$  均由键盘输入。

【运行结果】输入样例#： 10 3；

输出样例#： 3 6 9 2 7 1 8 5 10 4

- 6) P1427 小鱼的数字游戏  
7) P1428 小鱼比可爱  
8) P1047 校门外的树

### 3.1.2. 二维/多维数组：将“数”成表！

当汇聚很多规格相同的“数串”，列表。

#### A. 【010-矩阵相加】

问题描述：输入两个  $3 \times 3$  矩阵，输出它们的相加矩阵。例如，

$$\begin{array}{ccc} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{array} + \begin{array}{ccc} 9 & 9 & 9 \\ 8 & 8 & 8 \\ 7 & 7 & 7 \end{array} = \begin{array}{ccc} 10 & 11 & 12 \\ 10 & 11 & 12 \\ 10 & 11 & 12 \end{array}$$

```

1 #include<iostream>
2 using namespace std;
3 int main(){
4     int a[3][3],b[3][3],c[3][3];
5     int i,j;
6     for(i=0;i<=2;i++)
7         for(j=0;j<=2;j++)
8             cin>>a[i][j];
9     for(i=0;i<=2;i++)
10        for(j=0;j<=2;j++)
11            cin>>b[i][j];
12    for(i=0;i<=2;i++){
13        for(j=0;j<=2;j++){
14            c[i][j]=a[i][j]+b[i][j];
15            cout<<c[i][j]<<" ";
16        }
17        cout<<endl;
18    }
19    return 0;
20 }
```



## 【知识点 010】二维数组与多维数组

**二维数组**，是用两项索引号对变量进行编号的数组。例如，

int Array[3][5]	0	1	2	3	4
0	1	2	3	4	5
1	12	34	56	78	90
2	123	456	789	543	210

### (1) 定义和初始化

#### 【固定格式】

**类型关键字 二维数组名[行数][列数] 【={ {初值 00, 初值 01, ..... , 初值 0m}, {初值 10, 初值 11, ..... , 初值 1m}, ..... , {初值 n0, 初值 n1, ..... , 初值 nm} }】；**

在定义数组时，若未对数组赋初始值，那么各元素的值将是内存中的不定值。若定义数组的同时对其赋初始值，可以整体赋值，也可部分赋值。部分赋值时，其余元素自动赋为 0 (或空)。若花括号{}内为空，则数组整体赋为 0 (或空)。

#### 赋值和调用

对数组的整体赋值，只在定义数组时进行。调用时，只能单个元素逐一赋值。在调用特定数组元素时，其行、列下标（索引号）均从 0 开始，即一个元素总数为 n 个的数组，它各个元素的下标范围为从 0 到 n-1。若下标值在这个范围以外的情况，视为数组下标越界。

#### 【固定格式】二维数组名 [行标] [列标]

\*注：数组变量的下标，可以用常值、变量或表达式等方式来表示。



\*注：二维数组定义或作为函数形式参数描述时，只允许第一个[]内为空，后面[]内的数值用来确定增加维度的深度。若[]内为空，则必须为数组整体赋初始值，且数组元素的总数将由初始值的个数决定。

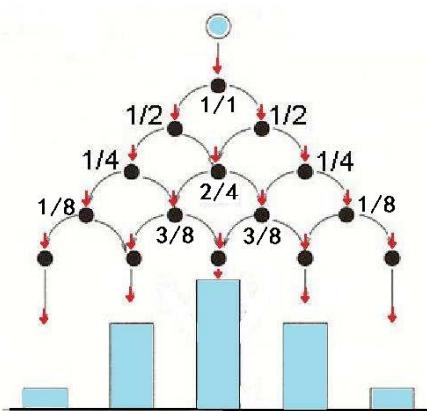
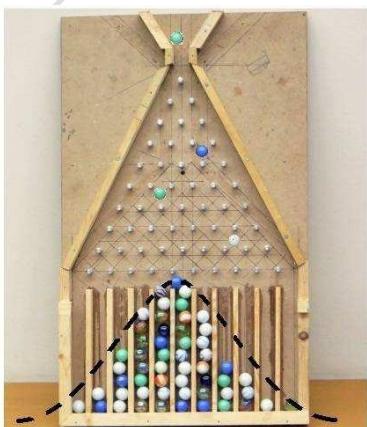
**多维数组**，是以两项以上索引号对变量进行编号的数组，数组的索引号有几项就称为几维数组。

多维数组实际是一个抽象概念，它等价于将各项索引的乘积存入一维数组。例如，int array[3][5]等价于 int array[15]， $3 \times 5 = 15$ 。在定义、赋值与调用等也与一、二维数组类似。



## B. 【随堂练】010-杨辉三角

问题描述：输入一个不大于 20 的正整数 n，输出 n 行的杨辉三角。



## 【运行结果】

10											
1											
1	1										
1	2	1									
1	3	3	1								
1	4	6	4	1							
1	5	10	10	5	1						
1	6	15	20	15	6	1					
1	7	21	35	35	21	7	1				
1	8	28	56	70	56	28	8	1			
1	9	36	84	126	126	84	36	9	1		
1	10	45	120	210	252	210	120	45	10		1



### 【关联题 00n】

- 1) 投票问题：某小团队，共有 15 人，要评选人群成员，每个人都是候选人，且每人可以投三票，在全员完成投票后，从大到小排序出每人的得票数。请编程模拟此投票过程。
  - 2) P2141 珠心算测验

## 3.2. 字符串：词以群分

字符串数组：若将“符号”串成“串”，成词。



问题描述：输入你的姓名，屏幕打印你给 Grace 的生日卡。

```

1 #include<iostream>
2 #include<cstring>
3 using namespace std;
4 int main(){
5     char my_name[20],your_name[20];
6     cin.getline(your_name,20,'\'n');
7     strcpy(my_name,"Dear Grace");
8     cout<<"*****\n";
9     cout<<"To: "<<my_name<<endl;
10    cout<<"Happy Birthday To You ! \n";
11    cout<<"\t From: "<<your_name<<endl;
12    cout<<"*****\n";
13    return 0;
14 }
```

```

Yan An
*****
To: Dear Grace
Happy Birthday To You !
From: Yan An
*****
```

【运行结果】



### 【知识点 011】1-字符串数组

**字符串数组**，是可以存储字符型数据的数组，它也根据下标项的个数也可分为一维或多维数组。在定义、赋值与调用等也与数型的一、二维数组类似。例如，

索引	0	1	2	3	4	5	6	7	8
char c_ar[9]	H	E	L	L	O	\0			

#### (1) 定义和初始化

```
【固定格式】char 数组名[元素总数]【={ '字符 0' , '字符 1' , ..... , '\0' }】;
```

```
char 数组名[元素总数]【= “字符串”】;
```

当用字符串来对字符数组进行初始化时（固定格式二），在字符串的有效内容结尾处会自动加一个空字符（'\0' 或 0）来表示字符结束，也需占用一个数字元素的位置。因此，字符数组被用于存储短于总长度的一组字符，每个数组元素存储的单个字符。

\*注：当需要存储多组字符串时，可以使用二维字符数组来表示。

## (2) **cin.getline (字符数组名，存储长度【，结束字符】)**

字符数组名：输入的字符内容，将被存入的地方。

存储长度：限制存储的最大容量。

结束字符：用来判断用户输入结束的字符，当它缺省时默认是换行符（'\n'）。

### **它与 cin>>的区别：**

任何的空白符（空格、Tab、换行、回车）均为输入分隔，只接收单独的词，而不能输入完整句。不能指定存储容量，使程序不稳定。

输入关键词	变量分隔符	对应输出
<b>cin&gt;&gt;字符数组名</b>	空格、Tab、换行、回车	<b>cout&lt;&lt;</b>
<b>cin.getline ( 字符数组名， 长度， 结束字符 )</b>	换行、或自定义	-
<b>scanf ( "%s" ， 字符数组名)</b>	空格	<b>printf ( )</b>
<b>gets (字符数组名)</b>	回车	<b>puts ( )</b>
<b>getchar (字符数组名[下标])</b>	输入单个字符	<b>putchar ( )</b>



**(3) 常用字符类型转换函数 <cstdlib>**

- **atoi (字符数组名):** 转为整型 int
- **atol (字符数组名):** 转为长整型 long
- **atof (字符数组名):** 转为浮点型 float

**(4) 常用操作函数<cstring>**

- **memset (字符数组 string[ ] , 值 , 字节长度 ):**  
按位对 string 数组的全字节进行赋指定值，它是对较大的数组进行清零操作的一种最快方法，例如， memset( str , 0 , sizeof(str) )。
- **strcat ( 字符数组 dest[ ] , 字符数组 src[ ] ):**  
将 src 内容附加到 dest 的末尾，返回数组 dest。
- **strcmp ( 字符数组 string1[ ] , 字符数组 string2[ ] ):**  
比较两个字符数组 string1 和 string2; string1=string2, 返回 0;  
string1>string2, 返回正数; string1<string2, 返回负数。
- **strcpy ( 字符数组 dest[ ] , 字符数组 src[ ] ):**  
将 src 内容拷贝到 dest，返回数组 dest。
- **strlen ( 字符数组 string[ ] ):**  
返回一组字符的实际长度。终止符'\0'不计在内。
- **strlwr ( 字符数组 string[ ] ):**  
将字符串中大写字母转换为小写字母，但 linux 系统下不支持该函数。  
(另注， tolower(单字符)。)
- **strupr ( 字符数组 string[ ] ):**  
将字符串中小写字母转换为大写字母，但 linux 系统下不支持该函数。

(另注, toupper(单字符)。)

\*注: 字符之间的大小比较实际是比较该字符对应的 ASCII 码值的大小。

因此, 字符的大写转小写也可写为: 字符+32 或字符+( 'a' - 'A' ), 字符的小写转大写也可写为: 字符-32 或字符-( 'a' - 'A' )。

### (5) "0"、'0'、'\0'、0 区别

符号 表示含义	int( )	sizeof( )
"0" 字符串 0, 末尾隐含有字符串结束符 '\0' '	-	2
'0' 字符 0	48	1
'\0' 空字符	0	1
0 空字符, 或数字 0	0	4

这一原理常被用于数字字符转化为数字的实际数值。

**字符转数**可写作: 数型变量=一个数字字符- '0' , 数型变量=一个数字字符-48; **数转字符**可写作: 字符变量=一个数字+ '0' , 字符变量=一个数字+48。

\*注: 这一写法, 对 string 类型变量也同样适用。

### (6) sscanf( ) 和 sprintf( )

我们经常涉及到数字与字符串之间的转换, 这里介绍头文件<cstdio>中的 sprintf 和 sscanf 函数, 它们类似于 scanf 和 printf, 但不同点是从字符串\*str 用于输入输出。

**sprintf 函数原型为 int sprintf(char \*str, const char \*format, ...)**。作用是格式化字符串, 具体功能如下: (1) 将数字变量转换为字符串。(2) 得到整型变量的 16 进制和 8 进制字符串。(3) 连接多个字符串。

```
char str[256] = { 0 };
```



```

int data = 1024;

sprintf(str,"%d",data); //将 data 转换为字符串

cout<<str<<endl;

sprintf(str,"0x%X",data); //获取 data 的十六进制

cout<<str<<endl;

sprintf(str,"0%o",data); //获取 data 的八进制

cout<<str<<endl;

const char *s1 = "Hello";

const char *s2 = "World";

sprintf(str,"%s %s",s1,s2); //连接字符串 s1 和 s2

cout<<str<<endl;

```

**sscanf函数原型为 int sscanf(const char \*str, const char \*format, ...).**

将参数 str 的字符串根据参数 format 字符串来转换并格式化数据，转换后的结果存于对应的参数内。具体功能如下：(a) 根据格式从字符串中提取数据。如从字符串中取出整数、浮点数和字符串等。(b) 取指定长度的字符串。(c) 取到指定字符为止的字符串。(d) 取仅包含指定字符集的字符串。(e) 取到指定字符集为止的字符串。

**sscanf 可以支持格式字符%[]:**

- (a) - : 表示范围，如：%[1-9]表示只读取 1-9 这几个数字，%[a-z]表示只读取 a-z 小写字母，%[A-Z]只读取 A-Z 大写字母
- (b) ^ : 表示不取，如：%[^1]表示读取除'1'以外的所有字符，%^[/]表示除/以外的所有字符，%^ [注意^后面有一个空格!] 表示读取空格前是所有字符。

(c), : 范围用", "相连接, 如%[1-9,a-z]表示同时取 1-9 数字和 a-z 小写字母。

解析网址的例子如下所示:

```
const char s[256] = "http://www.baidu.com:1234";
char protocol[32] = { 0 };
char host[128] = { 0 };
char port[8] = { 0 };
sscanf(s,"%[:];//%[:]:%[1-9]",protocol,host,port);
printf("protocol: %s\n",protocol);
printf("host: %s\n",host);
printf("port: %s\n",port);
```

string 类型：“字符串”的专属容器。

### A. 【011-2. 单词分行】

问题描述：输入一段字符串，将其中的单词分行输出。

```
1 #include<iostream>
2 #include<string>
3 using namespace std;
4 int main(){
5     string str;
6     getline(cin,str);
7     int len=0;
8     len=str.size();
9     for(int i=0;i<len;i++){
10         int pos=0;
11         pos=str.find(" ");
12         str[pos]='\n';
13     }
14     cout<<str;
15 }
16 }
```



```
I am a teacher, you are a student.  
I  
am  
a  
teacher,  
you  
are  
a  
student.
```

【运行结果】



### 【知识点 011】2-string 类型

**string** 类型，是 C++ 在 STL 里扩展的可以存储字符型数据的类型。

该型变量在定义、赋值或调用时类似普通变量，但若要调用其中单个字符时用法类似一维数组。而它的一维数组可以用来存储多组字符串，但若要调用其中单个字符时用法类似二维数组。

#### (1) 定义和初始化

【固定格式】

**string** 字串变量名【= “字符串”】；

**string** 字串变量名【(“字符串”）】；

**string** 字串数组名[元素总数]【={“字符串 0”,……,“字符串 n”}】；

#### (2) 调用、赋值整体字符串

【固定格式】字串变量名=“字符串”；

#### (3) 调用、赋值单个字符

【固定格式】

字串变量名[下标]

字串变量名[元素号][下标]

字串变量名.at(下标)

字串变量名[元素号].at(下标)

输入关键词	变量分隔符	对应输出
<code>cin&gt;&gt;字符串变量名</code>	空格、Tab、换行、回车	<code>cout&lt;&lt;</code>
<code>getline ( cin , 字串变量名 )</code>	换行	-

#### (4) 常用 string 成员函数<string>

##### 【固定格式】字符串变量 . 成员函数 ( )

- `.size ( )`或 `.length ( )`: 求字符串长度。
- `.empty ( )`: 判断字符串是否为空。
- `.insert( 位置 i , "插入串" )`: 在指定位置插入字符串。
- `.replace( 位置 i , 替换长度 , "替换串" )`: 将替换指定位置字符串。
- `.substr ( 开始位置 i , 取串长度 )`: 取字符串的子串。
- `.erase ( 位置 i , 删除长度 )`: 删除指定位置字符串，输出剩余部分。
- `.find ( "子串" )`: 查找子串，找到，返回子串首次出现的位置；反之，  
返回 `string::npos` (值常为-1)。



##### B. 【随堂练】011-1.字符比较统计

问题描述：先输入一个字符串，再输入一个字符，然后分别统计出字符串中大于、等于、小于该字符的个数。

【运行结果】

abcdefg	h	i	a	b	c	d	e	f	g
d									
10	2	6							



##### 【随堂练】011-2.查找子串

问题描述：利用 `string` 类型实现，先输入 5 个字符串，再输入一个待查字符串，然后统计出含待查字符串的个数。



```

输入5个串
abc
bcd
edf
ere
dfedf
输入待查串
edf
输出个数
【运行结果】2

```



### C. 【随堂想 011】

(1) 2018 年 NOIP 普及真题-阅读程序 1.

```

1 #include <cstdio>
2 using namespace std;
3 char st[100];
4 int main() {
5     scanf("%s", st);
6     for (int i = 0; st[i]; ++i) {
7         if ('A' <= st[i] && st[i] <= 'Z')
8             st[i] += 1;
9     }
10    printf("%s\n", st);
11    return 0;
12 }

```

输入: QuanGuoLianSai

输出:

(2) 2017 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     int t[256]; string s; int i;
5     cin >> s;
6     for (i = 0; i < 256; i++)
7         t[i] = 0;
8     for (i = 0; i < s.length(); i++)
9         t[s[i]]++;
10    for (i = 0; i < s.length(); i++)
11        if (t[s[i]] == 1) {
12            cout << s[i] << endl;
13            return 0;
14        }
15    cout << "no" << endl; return 0;
16 }

```

输入: xyzxyw

输出:

(3) 2017 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 using namespace std;
3 int main() {
4     string ch;int a[200];int b[200];
5     int n, i, t, res; cin >> ch;
6     n = ch.length();
7     for (i = 0; i<200; i++)b[i] = 0;
8     for (i = 1; i <= n; i++){
9         a[i] = ch[i - 1] - '0';
10        b[i] = b[i - 1] + a[i];
11    }
12    res = b[n];
13    t = 0;
14    for (i = n; i > 0; i--) {
15        if (a[i] == 0) t++;
16        if (b[i-1]+t<res) res=b[i-1]+t;
17    }
18    cout << res << endl; return 0;
19 }
```

输入: 1001101011001101101011110001

输出: 

## (4) 2015 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int main(){
5     string str;
6     int i;
7     int count; count = 0;
8     getline(cin, str);
9     for(i = 0; i < str.length(); i++)
10        if(str[i]>='a'&&str[i]<='z')
11            count++;
12     cout << "It has " << count << " lowercases" << endl;
13     return 0;
14 }
```

输入: NOI2016 will be held in Mian Yang.

输出: 

## (5) 2014 年 NOIP 普及真题-阅读程序 3.

```

1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int main(){
5     string st;
6     int i, len;
7     getline(cin, st);
8     len = st.size();
9     for(i = 0; i < len; i++)
10        if(st[i] >='a'&&st[i]<='z')
11            st[i] = st[i] - 'a' + 'A';
12     cout << st << endl;
13     return 0;
14 }
```



输入: Hello, my name is Lostmonkey.

输出: 

## (6) 2011 年 NOIP 普及真题-阅读程序 2.

```

1 #include<iostream>
2 #include<string>
3 using namespace std;
4 int main(){
5     string map= "2223334445556667778889999";
6     string tel;
7     int i;
8     cin>>tel;
9     for(i=0;i<tel.length();i++)
10        if((tel[i]>='0')&&(tel[i]<='9'))
11            cout<<tel[i];
12        else
13            if((tel[i]>='A')&&(tel[i]<='Z'))
14                cout<<map[tel[i]-'A'];
15    cout<<endl;
16    return 0;
17 }
```

输入: CCF-NOIP-2011

输出: 

## (7) 2011 年 NOIP 普及真题-阅读程序 2.

```

1 #include <iostream>
2 #include <string>
3 using namespace std;
4 int main(){
5     string s;
6     char m1, m2;
7     int i;
8     getline(cin, s);
9     m1 = ' ';
10    m2 = ' ';
11    for (i=0; i<s.length(); i++)
12        if (s[i] > m1) {
13            m2 = m1;
14            m1 = s[i];
15        }
16        else if (s[i] > m2)
17            m2 = s[i];
18    cout<<int(m1)<<' '<<int(m2)<<endl;
19    return 0;
20 }
```

输入: Expo 2010 Shanghai China

输出:

提示：

字符	空格	'0'	'A'	'a'
ASCII 码	32	48	65	97



### 【关联题 011】

- 1) 字符串排序：首先输入一个 M，表示输入 M 个字符串，然后对这 M 个字符串进行选择排序，按从小到大的顺序输出字符串

【运行结果】输入样例#：5

输出样例#：

abcd

a

ab

abd

ae

a

ab

abcd

abd

ae

- 2) 回文判断：输入一个符号数量不超过 100 个的字符串，以“.”为结束。

判断该字符串是否构成回文。

回文：指一串字符无论从左往右，还是从右往左读，都一样。例如，12321，abccba，aa 等。提示：先将要判断的字符串读入数组，并计算长度，然后首尾字符比较，不断向中间靠拢。

- 3) 数拼音：输入一个 0 到 99 之间的数，输出该数拼音的念法。

【运行结果】输入样例#：35；输出样例#：SAN SHI WU

- 4) 字符运算器：输入一串两个数字夹着运算符号的字符串，做运算。

【运行结果】输入样例 1#：123+123；输出样例 1#：246



输入样例 2#: 1/2; 输出样例 1#: 0.5

5) 高精度加法: 输入两个最大位数不超过 127 的长正整数 A 和 B, 求 A+B。

提示: 位数达 127 位的长正整数已超过数据类型所能表示的极限, 因此,

利用数组来存储长正整数的每一位的数字, 然后模拟数学上的加法运算。

【运行结果】 输入样例 1#: 2 99; 输出样例 1#: 101

输入样例 2#: 18446744073709551616

18446744073709551616

输出样例 2#: 36893488147419103232

6) P1055 ISBN 号码

7) P1200 [USACO1.1] 你的飞碟在这儿 Your Ride Is Here.

8) P1914 小书童——密码

9) P1308 统计单词数

10) P1553 数字反转 (升级版)

# 第4章 函数

## 4.1. 反值函数和 void 函数：模块 DIY

自制个性化解题函数！



问题描述：输入一个正整数 n，依次求出 n 的阶乘和 n+3 的阶乘。

n 的阶乘公式： $n! = 1 \times 2 \times 3 \times \dots \times n$ 。

```

1 #include<iostream>
2 using namespace std;
3 long long f(int x){
4     long long res=1;
5     for(int i=1;i<=x;i++)
6         res*=i;
7     return res;
8 }
9 int main(){
10    int n;
11    long long ans1, ans2;
12    cin>>n;
13    ans1=f(n);
14    ans2=f(n+3);
15    cout<<ans1<< " "<<ans2;
16    return 0;
17 }
```

【运行结果】输入样例#：5 ；输出样例#：120 40320



### 【知识点 012】0-函数：模块化编程思维

**函数：**模块化描述问题的解法，返回问题解决的结果。

运用函数的主要意义：**代码重用，问题分解。** C++中包含的函数种类：

- (1) 主函数 main( )：有且只有一个；



- (2) 系统函数：例如，`getchar()`、`abs()`，必须要加上相关头文件；
- (3) 自定义函数。



### 【知识点 012】1-返值函数

**返值函数：**函数在调用完毕后，将返回一个确定数据类型的数值。

- (1) 定义返值函数：

#### 【固定格式】

```
返值类型 函数名(【类型 形参名 1，类型 形参名 2，.....】) {
    .....函.....数.....体..... ;
    return 表达式;
}
```

- (2) 调用返值函数：

#### 【固定格式】函数名(【实参名 1，实参名 2，.....】) ;

A.

#### 【012-2.void 函数】

```

1 #include<iostream>
2 using namespace std;
3 void f(int a, float b){
4     float r;
5     r=a*b;
6     cout<<"I'm a function!"<<endl;
7     cout<<"The result is "<<r<<endl;
8 }
9 int main(){
10    for(int i=1;i<=10;i++)
11        f(i,0.5);
12    return 0;
13 }
```

```
I'm a function!
The result is 0.5
I'm a function!
The result is 1
I'm a function!
The result is 1.5
I'm a function!
The result is 2
I'm a function!
The result is 2.5
I'm a function!
The result is 3
I'm a function!
The result is 3.5
I'm a function!
The result is 4
I'm a function!
The result is 4.5
I'm a function!
The result is 5
```

**【运行结果】****【知识点 012】2-void 函数**

**void 函数:** 函数只执行一系列操作过程，而无返回数值。

(1) 定义 void 函数:

**【固定格式】**

```
void 函数名(【类型 形参名 1, 类型 形参名 2, .....】) {
    .....函.....数.....体..... ;
}
```

(2) 调用 void 函数:

**【固定格式】函数名(【实参名 1, 实参名 2, .....】) ;****【知识点 012】3-形式参数和实际参数**

**形式参数:** 在定义函数时，将由外部传入函数内，用于参与函数运算过程的，并不存在的**虚拟变量**。而对于一些无参函数，形式参数并非必需要素，形参列表可以省略，则括号( )内为空。

**实际参数:** 在调用函数时，将值传入函数参与运算的，真实存在的**实际变量**。

**实参和形参可以重名。**





## B. 【随堂练】0012-组合数 $C(n,m)$

问题描述：依次输入两个非负整数  $n$  和  $m$  ( $n > m$ )，求从  $n$  个不同元素中选出  $m$  个元素一共有多少种组合  $C_n^m$ ，用“函数”方法求解组合数  $C_n^m$ 。

组合数公式： $C_n^m = n! / m! * (n - m)!$ 。 (! 为阶乘符号)

【运行结果】输入样例#：10 5；输出样例#：252



## C. 【随堂想 012】

(1) 2010 年 NOIP 普及真题-阅读程序 2.

```

1 #include <iostream>
2 using namespace std;
3 int rSum(int j){
4     int sum = 0;
5     while (j != 0) {
6         sum = sum * 10 + (j % 10);
7         j = j / 10;
8     }
9     return sum;
10}
11 int main(){
12     int n, m, i;
13     cin >> n >> m;
14     for (i = n; i < m; i++)
15         if (i == rSum(i))
16             cout << i << ' ';
17     return 0;
18}

```

输入：90 120

输出：



### 【知识点 012】5-函数原型和函数重载

#### (1) 函数重载

定义同名函数时，注意它们的参数列表中的内容务必不同。因此，据此特征

可以构造多个功能完全不同的同名函数。而在函数调用时会根据传入的实参列表情况，对函数进行相应的重载。

### 【程序范例】函数重载-除法

```

1 #include<iostream>
2 using namespace std;
3 int divide(int a,int b){
4     return a/b;
5 }
6 float divide(float a,float b){
7     return a/b;
8 }
9 int main(){
10    int x=5,y=2;
11    float n=5.0,m=2.0;
12    cout<<divide(x,y)<<"\n";
13    cout<<divide(n,m)<<"\n";
14    return 0;
15 }
```

输出：

### (2) 函数原型

运用函数时要求：先定义，再调用。**函数定义中不可嵌套对其他函数的定义，但可以嵌套对其他函数的调用**，而被嵌套调用的函数，必须是之前定义过的函数。

由于函数定义必定有先后顺序，但若有两个函数需要互相调用对方，则可以在定义函数前先定义一个“**函数原型**”，即对函数做一个简短、重要的声明，以便让编译器了解函数的参数和返回性质。

**【固定格式】\*注：定义函数原型时，句尾的分号；不可缺省。**

**函数类型 函数名(【类型 形参名1，类型 形参名2，……】)；**

**函数类型 函数名(【类型 ，类型 ，……】)；**

### 【程序范例】函数原型-奇偶数



```

1 #include<iostream>
2 using namespace std;
3 void odd(int a); //或写为: void odd(int);
4 void even(int a); //或写为: void even(int);
5 int main(){
6     int i;
7     do{
8         cout<<"Type a number: (0 to exit)">>i;
9         odd(i);
10    }while(i!=0);
11    return 0;
12 }
13 }
14 void odd(int a){
15     if((a%2)!=0) cout<<"Number is odd.\n";
16     else even(a);
17 }
18 void even(int a){
19     if((a%2)==0) cout<<"Number is even.\n";
20     else odd(a);
21 }

```

输入: 2 输出:

输入: 1 输出:

输入: 0 输出:

输入: -3 输出:



## 【知识点 012】6-变量的引用范围：局部变量和全局变量

### 【程序范例】全局部变量-加

```

1 #include<iostream>
2 using namespace std;
3 int a=2,b=4,c;
4 int plusab(float a,float b){
5     float c;
6     c=a+b;
7     cout<<c<<endl;
8 }
9 int main(){
10     float x,y;
11     cin>>x>>y;
12     plusab(x,y);
13     cout<<c<<endl;
14     for(int i=a;i<=b;i++){
15         c+=i;
16     }
17     cout<<c<<endl;
18 }
19

```

\*试验并纠错

输入: 1.2 3.4      输出:

**局部变量:** 在主函数或自定义函数内部定义的变量, 它的有效引用范围从定义该变量的位置到它所处的函数或结构代码块内。

**全局变量:** 在函数外部定义的变量, 它的有效引用范围从定义该变量的位置到之后的任意位置。

全局变量在引用范围内一直占用内存空间, 它在定义时若未赋值, 其默认值为 0。当全局变量和局部变量重名时, 在局部变量的引用范围内, 局部变量优先级高于全局变量, 全局变量将被自动屏蔽。



### 【知识点 012】7-参数传递: 传值和传址

**参数传递:** 在程序运行过程中, 实际参数就会将参数值传递给相应的形式参数, 然后在函数中实现对数据处理和返回的过程, 方法有按值传递参数, 按地址传递参数和按数组传递参数。

#### (1) 传值调用

将实际参数值的副本复制给形式参数, 函数执行后, 不改变实际参数的值。

#### (2) 传址调用

将实际参数的存储地址传递给形式参数, 函数执行后, 实际参数的值会受到函数过程的影响而发生改变。

#### 【固定格式】

类型 函数名(类型 &形参名 1, 类型 &形参名 2, .....){

.....函.....数.....体.....;

}



### 【程序范例】参数传递-前后数

```

1 #include<iostream>
2 using namespace std;
3 void prevnext(int x,int &prev,int &next){
4     prev=x-1;
5     next=x+1;
6     x=0;
7 }
8 int main(){
9     int x=100,y,z;
10    prevnext(x,y,z);
11    cout<<x<<" Previous+"<<y<<, Next+"<<z;
12    return 0;
13 }
```

输出:

### (3) 数组调用

当数组作为参数，在函数的实参和形参之间传递时，采用的是传址调用。而对数组参数的传址调用，在定义函数时无需使用&符号。

#### ① 一维数组参数

##### 【固定格式】

类型 函数名 ( 类型 形参数组名 [ ] ) {

.....函.....数.....体..... ;

}

函数名 ( 实参数组名 );

#### ② 多维数组参数

##### 【固定格式】

类型 函数名 ( 类型 形参数组名 [ ][列深度] ) {

.....函.....数.....体..... ;

}

函数名 ( 实参数组名 [行深度] );

## (2) 2010 年 NOIP 普及真题-阅读程序 1.

```

1 #include <iostream>
2 using namespace std;
3 void swap(int & a, int & b){
4     int t;
5     t = a;
6     a = b;
7     b = t;
8 }
9 int main(){
10    int a1, a2, a3, x;
11    cin>>a1>>a2>>a3;
12    if (a1 > a2) swap(a1, a2);
13    if (a2 > a3) swap(a2, a3);
14    if (a1 > a2) swap(a1, a2);
15    cin>>x;
16    if (x < a2)
17        if (x < a1)cout<<x<<' '<<a1<<' '<<a2<<' '<<a3<<endl;
18        else cout<<a1<<' '<<x<<' '<<a2<<' '<<a3<<endl;
19    else
20        if (x < a3) cout<<a1<<' '<<a2<<' '<<x<<' '<<a3<<endl;
21        else cout<<a1<<' '<<a2<<' '<<a3<<' '<<x<<endl;
22    return 0;
23 }
```

输入: 91 2 20

输出:

77



## 【关联题 012】

- 1) P1149 火柴棒等式
- 2) P1217 [USACO1.5]回文质数 Prime Palindromes



## 4.2. 递归函数

从前有座山，山里有座庙，庙里有和尚，老和尚对小和尚说：“

从前有座山，山里有座庙，庙里有和尚，老和尚对小和尚说：

从前有座山，山里有座庙，庙里有和尚，老和尚对小和尚说：.....”



### A. 【013-递归求 n!】

问题描述：请用“递归函数”编程求  $n!$  的阶乘公式： $n! = 1 \times 2 \times 3 \times \dots$

$\boxed{\times n.}$

```

1 #include<iostream>
2 using namespace std;
3 long long fac(int a){
4     if(a==1) return 1;
5     else return a*fac(a-1);
6 }
7 int main(){
8     int n;
9     cout<<"Type a number:";
10    cin>>n;
11    cout<<n<<"!"<< "="<<fac(n)<<endl;
12    return 0;
13 }
```

Type a number:10

【运行结果】 10!=3628800



### 【知识点 013】递归函数

**递归函数：**函数直接或间接地调用自己的方法叫“函数的递归调用”，这样的函数称为“递归函数”。

问题可以用“递归函数”解决的两个前提要素：

(1) **递归关系：**原问题可被转化为局部解法相同，但规模（参数值）不同

的新问题；新问题可表示为确定的，运用了原问题解法的，新运算关系。

**(2) 递归边界：**有明确的递归结束条件。

**例如：求 n! 问题的递归要素转换**

$$n! = f(n) = \begin{cases} 1, & n = 1 \text{——递归边界} \\ n \times f(n-1), & n > 1 \text{——递归关系} \end{cases}$$



### B. 【随堂练】013-递归求 Fibo

问题描述：请用“递归函数”编程求 Fibonacci (斐波那契) 数列：在数学上，斐波纳契数列被以递推的方法定义： $F(0)=0$ ,  $F(1)=1$ ,  $F(2)=1$ ,  $F(n)=F(n-1)+F(n-2)$  ( $n \geq 3$ ,  $n \in N^*$ ) 即，最前两项值为 0 和 1，之后的每一项的值是其前两项的相加。输出该数列的前 n 项。

【运行结果】输入样例#：10 ； 输出样例#：0 1 1 2 3 5 8 13 21 34



### C. 【随堂想 013】

(1) 2008 年 NOIP 普及真题-阅读程序 2.

```

1 #include<iostream>
2 using namespace std;
3 void foo(int a, int b, int c){
4     if(a > b)
5         foo(c, a, b);
6     else
7         cout<<a<<' '<<b<<' '<<c<<endl;
8 }
9 int main(){
10    int a, b, c;
11    cin >> a >> b >> c;
12    foo(a, b, c);
13    return 0;
14 }
```

输入：3 1 2

输出：



## (2) 2009年NOIP普及真题-阅读程序1.

```

1 #include<iostream>
2 using namespace std;
3 int a,b;
4 int work(int a,int b){
5     if (a%b)
6         return work(b,a%b);
7     return b;
8 }
9 int main(){
10    cin >> a >> b;
11    cout << work(a,b) << endl;
12    return 0;
13 }
```

输入: 20 12

输出:

## (3) 2014年NOIP普及真题-阅读程序2.

```

1 #include <iostream>
2 using namespace std;
3 int fun(int n){
4     if(n == 1)
5         return 1;
6     if(n == 2)
7         return 2;
8     return fun(n - 2) - fun(n - 1);
9 }
10 int main(){
11     int n;
12     cin >> n;
13     cout << fun(n) << endl;
14     return 0;
15 }
```

输入: 7

输出:



## 【关联题 013】

1) P1028 数的计算

2) P1036 选数

# 第5章 结构体和数据的文件存储

## 5.1. 结构体

造“物种”——自定义类型。



```

1 #include<iostream>
2 #include<cstring>
3 #include<cstdlib>
4 using namespace std;
5 struct movies_t{
6     char title[50];
7     int year;
8 }mine,yours;
9 void printmovie(movies_t movie);
10 int main(){
11     char buffer[50];
12     strcpy(mine.title,"Zootopia");
13     mine.year=2016;
14     cout<<"Enter title:";
15     cin.getline(yours.title,50);
16     cout<<"Enter year:";
17     cin.getline(buffer,50);
18     yours.year=atoi(buffer);
19     cout<<"My favourite movie is:\n";
20     printmovie(mine);
21     cout<<"And yours:\n";
22     printmovie(yours);
23     return 0;
24 }
25 void printmovie(movies_t movie){
26     cout<<movie.title;
27     cout<<"("<<movie.year<<")\n";
28 }
```





## 【知识点 014】结构体

在处理数据时，根据问题实体的特征，常常需要把不同类型的数据视为一个整体。因此，提供了构造自定义类型的机制——结构体（struct）。

### (1) 结构体的定义

**【固定格式】** struct 类型名 {

    数据类型 1 成员名 1, 成员名 2, .....;

    数据类型 2 成员名 3;

.....;

//以上为该结构体的成员列表

} ;

### (2) 结构体变量的定义

**【固定格式】** 【struct】 结构体类型名 变量名列表;

**【固定格式】** struct 类型名 {

.....;

**成员列表**

.....;

} 变量名列表 ;

### (3) 结构体变量成员的访问

**【固定格式】** 结构体变量名 . 成员名

(4) 结构体变量的特点：变量可以整体操作；变量成员的访问方便且清晰；

结构体变量的初始化和数组的初始化类似。

例如：

```
struct tStudent {
    string name;
    int c, m, e;
} ;
tStudent m_st = { "Yan xiao an" , 84, 95, 90} ;
```



### B. 【随堂练】014-生日

问题描述：延小安想调查学校 OI 组每个同学的生日，并按照从大到小的顺序排序。但小安最近作业很多，没有时间，所以请你帮她排序。输入格式有 2 行，第 1 行为 OI 组总人数 n；第 2 行至第 n+1 行分别是每人的姓名 s、出生年 y、月 m、日 d。输出格式有 n 行，即 n 个生日从大到小同学的姓名。(如果有两个同学生日相同,输入靠后的同学先输出)

数据规模：1<n<100, length(s)<20

【运行结果】输入样例#：

```
3
Yangchu 1992 4 23
Qiujingya 1993 10 13
Luowen 1991 8 1
```

输入样例#：

```
Luowen
Yangchu
```



Qiujingya



## 【随堂想 014】

- (1) 2015 年 NOIP 普及真题-阅读程序 2.

```
1 #include<iostream>
2 using namespace std;
3 struct point{
4     int x;
5     int y;
6 };
7 int main(){
8     int a, b, c;
9     struct EX
10    {
11         int a; int b; point c;
12     }e;
13     e.a = 1;
14     e.b = 2;
15     e.c.x = e.a + e.b;
16     e.c.y = e.a * e.b;
17     cout<<e.c.x<<', '<<e.c.y<< endl;
18     return 0;
19 }
```

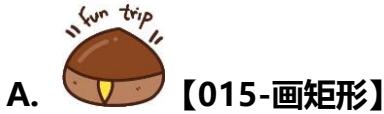
【运行结果】

## 【关联题 014】

- 1) P2095 营养膳食

## 5.2. 文件的输入输出

有“出处”，亦有“归属”。

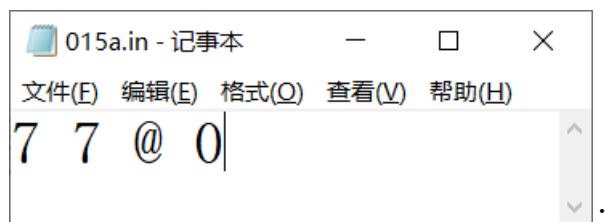


问题描述：根据参数，画出矩形。输入四个参数：前两个参数为整数，依次代表矩形的高和宽（高不少于 3 行不多于 10 行，宽不少于 5 列不多于 10 列）；第三个参数是一个字符，表示用来画图的矩形符号；第四个参数为 1 或 0，0 代表空心，1 代表实心。

请用文件的方式实现程序测试数据的输入、输出。

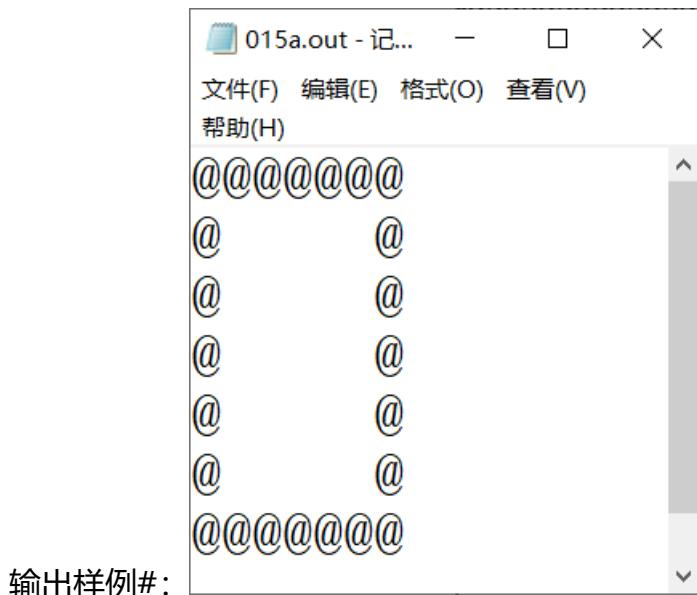
```

1 #include<iostream>
2 #include<cstdio> //调用freopen()函数需要
3 using namespace std;
4 int main(){
5     freopen("015a.in","r",stdin);
6     //只读方式打开015a.in文件用于输入数据
7     freopen("015a.out","w",stdout);
8     //只写方式打开015a.out文件用于输出数据
9     int h, w;    char c; bool f;
10    cin >>h >>w >>c >>f;
11    for( int i=1; i<=h; i++ ){
12        for( int j=1; j<=w; j++ ){
13            if( f==1 ) cout <<c;
14            else
15                if( i==1 || i==h || j==1 || j==w ) cout <<c;
16                else cout <<' ';
17        }
18        cout <<endl;
19    }
20    fclose(stdin); //关闭文件
21    fclose(stdout); //关闭文件
22    return 0;
23 }
```



【运行结果】输入样例#:





### 【知识点 015】fopen 的文件输入、输出

以往，我们在运行测试数据时，往往通过运行窗口来输入数据，并输出数据。但测试数据的输入效率较低，同时，结果数据会随着窗口的关闭而丢失。因此，利用文件来输入、输出数据，可以提高数据的输入输出效率，并有助于管理测试数据和保存结果数据。它的基本原理如下图所示：



关键词：fopen( )、fclose( )函数

#### 【固定格式】

```
#include<头文件>

#include<cstdio> //调用 fopen() 函数需要

using namespace std;

int main(){
```

```
freopen ( "输入文件" , "r" , stdin ) ;  
  
//只读方式打开输入文件用于输入数据  
  
Freopen ( "输出文件" , "w" , stdout ) ;  
  
//只写方式打开输出文件用于输出数据  
  
..... ....;  
  
.....原始程序.....;  
  
..... ....;  
  
fclose ( stdin ) ; //关闭文件  
  
fclose ( stdout ) ; //关闭文件  
  
return 0;  
  
}
```

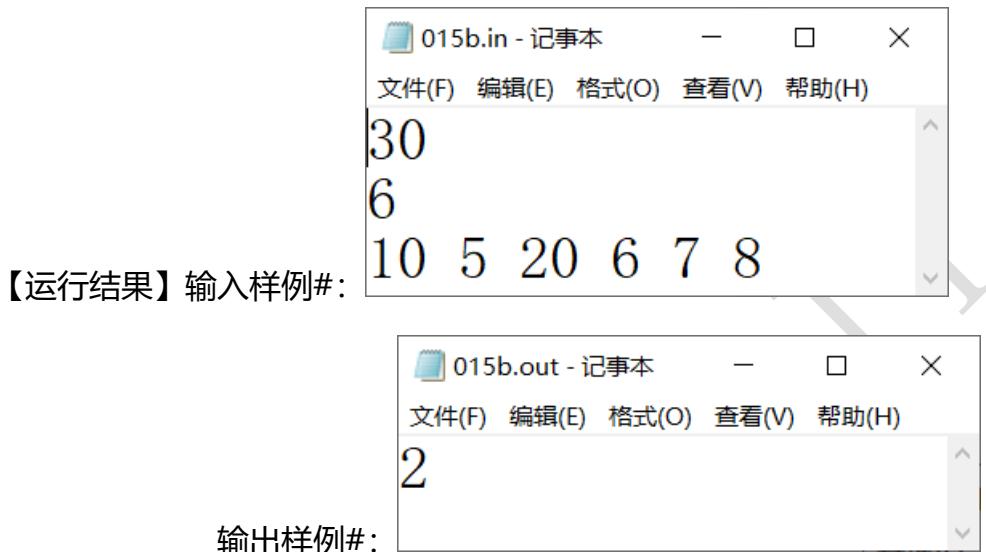
## B. 【随堂练】015-药房管理

问题描述：随着信息技术的蓬勃发展，医疗信息化已经成为医院建设中必不可少的一部分。计算机可以很好地辅助医院管理医生信息、病人信息、药品信息等海量数据，使工作人员能够从这些机械的工作中解放出来，将更多精力投入真正的医疗过程中，从而极大地提高了医院整体的工作效率。对药品的管理是其中的一项重要内容。现在药房的管理员希望使用计算机来帮助他管理。假设对于任意一种药品，每天开始工作时的库存总量已知，并且一天之内不会通过进货的方式增加。每天会有很多病人前来取药，每个病人希望取走不同数量的药品。如果病人需要的数量超过了当时的库存量，药房会拒绝该病人的请求。管理员希望



知道每天会有多少病人没有取上药。

请用文件的方式实现程序测试数据的输入、输出。(输入文件 015b.in,  
输出文件 015b.out)



### C. 【随堂想 015】

- (1) 复制以下程序到 Dev C++, 编译纠错, 使其可正常运行。

```
//纠错版“画矩形”

#include<iostream>

using namespace std;

int main(){

    freopen("1097.in",r,stdin);

    freopen(1097.out,'r',stdin);

    int h, w;    char c; bool f;

    cin >>h >>w >>c >>f;

    for( int i=1; i<=h; i++ ){

        cout << c << endl;
    }
}
```

```
for( int j=1; j<=w; j++ ){

    if( f==1 ) cout <<c;

    else

        if( i==1 || i==h || j==1 || j==w ) cout <<c;

        else cout <<' ';

    }

    cout <<endl;

}

fclose(stdout);

fclose(stdout);

return 0;

}
```



## ~我是彩蛋~

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 long long program_solved=0, knowledge_learned=0;
4
5 void study_hard(){
6     knowledge_learned++;
7     program_solved++;
8 }
9
10 int main(){
11     long long level=0;
12     while(365){
13         study_hard();
14         level++;
15         if(level==100){
16             cout<<"First Prize of Self-Growth ~"<<endl;
17             break;
18         }
19         else cout<<"We learned more everyday ."<<endl;
20     }
21     return 0;
22 }
```

# 附录

## 1. 运算符优先级

### 运算符优先级

优先级从上到下依次递减，最上面具有最高的优先级，逗号操作符具有最低的优先级。

相同优先级中，按结合顺序计算。大多数运算是从左至右计算。

只有三个优先级是从右至左结合的，它们是单目运算符、条件运算符、赋值运算符。

基本的优先级需要记住：

指针最优，单目运算优于双目运算。如正负号。

先乘除（模），后加减。

先算术运算，后移位运算，最后位运算。请特别注意： $1 \ll 3 + 2 \& 7$ 等价于 $(1 \ll (3 + 2))\& 7$ .

逻辑运算最后计算。

`++`优先级要大于`++`

优先级	运算符	名称或含义	使用形式	结合方向	说明
1	后置 <code>++</code>	后置自增运算符	变量名 <code>++</code>	左到右	
	后置 <code>--</code>	后置自减运算符	变量名 <code>--</code>		
	<code>[]</code>	数组下标	数组名[整型表达式]		
	<code>()</code>	圆括号	(表达式) / 函数名(形参表)		
	<code>.</code>	成员选择(对象)	对象.成员名		
	<code>-&gt;</code>	成员选择(指针)	对象指针->成员名		
2	<code>-</code>	负号运算符	<code>-</code> 表达式	左到右	单目运算符
	<code>(类型)</code>	强制类型转换	(数据类型)表达式		
	前置 <code>++</code>	前置自增运算符	<code>++</code> 变量名		单目运算符
	前置 <code>--</code>	前置自减运算符	<code>--</code> 变量名		单目运算符
	<code>*</code>	取值运算符	<code>*</code> 指针表达式		单目运算符
	<code>&amp;</code>	取地址运算符	<code>&amp;</code> 左值表达式		单目运算符
	<code>!</code>	逻辑非运算符	<code>!</code> 表达式		单目运算符
	<code>~</code>	按位取反运算符	<code>~</code> 表达式		单目运算符
3	<code>sizeof</code>	长度运算符	<code>sizeof</code> 表达式 / <code>sizeof</code> (类型)	左到右	
	<code>/</code>	除	表达式 / 表达式		双目运算符
	<code>*</code>	乘	表达式 * 表达式		双目运算符
	<code>%</code>	余数(取模)	整型表达式 % 整型表达式		双目运算符



4	+	加	表达式+表达式	左到右	双目运算符
	-	减	表达式-表达式		双目运算符
5	$\ll$	左移	表达式 $\ll$ 表达式	左到右	双目运算符
	$\gg$	右移	表达式 $\gg$ 表达式		双目运算符
6	>	大于	表达式>表达式	左到右	双目运算符
	$\geq$	大于等于	表达式 $\geq$ 表达式		双目运算符
	<	小于	表达式<表达式		双目运算符
	$\leq$	小于等于	表达式 $\leq$ 表达式		双目运算符
7	$\equiv$	等于	表达式 $\equiv$ 表达式	左到右	双目运算符
	$\neq$	不等于	表达式 $\neq$ 表达式		双目运算符
8	&	按位与	整型表达式&整型表达式	左到右	双目运算符
9	$\wedge$	按位异或	整型表达式 $\wedge$ 整型表达式	左到右	双目运算符
10		按位或	整型表达式 整型表达式	左到右	双目运算符
11	$\&\&$	逻辑与	表达式 $\&\&$ 表达式	左到右	双目运算符
12	$\ \ $	逻辑或	表达式 $\ \ $ 表达式	左到右	双目运算符
13	?:	条件运算符	表达式1? 表达式2: 表达式3	左到右	三目运算符
14	=	赋值运算符	变量=表达式	右到左	
	$/=$	除后赋值	变量/=表达式		
	$*=$	乘后赋值	变量*=表达式		
	$%=$	取模后赋值	变量%=表达式		
	$+=$	加后赋值	变量+=表达式		
	$-=$	减后赋值	变量-=表达式		
	$\ll=$	左移后赋值	变量 $\ll=$ 表达式		
	$\gg=$	右移后赋值	变量 $\gg=$ 表达式		
	$\&=$	按位与后赋值	变量 $\&=$ 表达式		
	$\wedge=$	按位异或后赋值	变量 $\wedge=$ 表达式		
15	,	逗号运算符	表达式,表达式,...	左到右	从左向右顺序

## 2. 数据类型

类型关键字	描述	字节	数值范围
<b>char</b>	字符 character	1	有符号(signed) char: $-2^7 \sim (2^7-1)$ 即 $-128 \sim 127$ 无符号 unsigned char: $0 \sim (2^8-1)$ 即 $0 \sim 255$
<b>bool</b>	布尔	1	假 false 0 或 真 true 1 (或非 0 数)
<b>short (int)</b>	短整型	2	有符号(signed) short: $-2^{15} \sim (2^{15}-1)$ 即 $-32768 \sim 32767$ 无符号 unsigned short: $0 \sim (2^{16}-1)$ 即 $0 \sim 65535$
<b>(long) int</b>	(长)整型 integer	4	有符号(signed) int: $-2^{31} \sim (2^{31}-1)$ 即 $-2147483648 \sim 2147483647$ 无符号 unsigned int: $0 \sim (2^{32}-1)$ 即 $0 \sim 4294967295$
<b>long long</b>	超长整型	8	有符号(signed) int $-2^{63} \sim (2^{63}-1)$ 无符号 unsigned int $0 \sim (2^{64}-1)$
<b>float</b>	单精度 浮点	4	约 $-3.4e+38 \sim 3.4e+38$ , 6 ~ 8 位有效数
<b>double</b>	双精度 浮点	8	约 $-1.7e+308 \sim 1.7e+308$ , 15 ~ 16 位有效数
<b>long</b>	长双精度	-	占字节数、取值范围和有效位数因编译器而异。
<b>double</b>	浮点	16	约 $-1.2e+4932 \sim 1.2e+4932$ , 18 ~ 19 位有效数

\*注: 1 字节=8 位 (1Byte=8bit), 各浮点型输出时只能显示 6 位数。



## 【扩展阅读】计算机内存中浮点型数据的存储结构

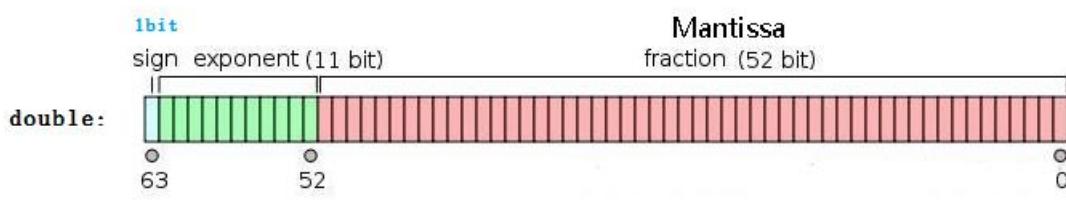
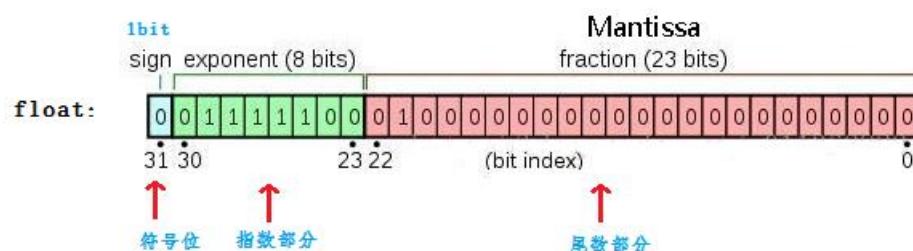
### 一、内存表示

一个十进制数N可以写成 $N = M \times 10^e$ , 一个R进制数N可以写成 $N = M \times R^e$ , (其中: M 尾数, e 指数, R 基数)。阶码 E: 表达指数部分, 用整数形式表示, 指明小数点在数据中的位置, 决定浮点数的表示范围。尾数 M: 用定点小数表示, 给出有效数字的位数决定了浮点数的表示精度。例如:

对于十进制数:  $156.78 = 15.678 \times 10^1 = 1.5678 \times 10^2 = 0.15678 \times 10^3 = M \times R^E$

对于二进制数:

$$\begin{aligned} 1011.1101 &= 0.10111101 \times 2^{+4} \\ &= 10.111101 \times 2^{+2} = 1.0111101 \times 2^{+3} \quad (\text{规格化表示法}) \\ &= 1.0111101 \times 2^{+11} \quad (11 \text{ 表示二进制的 } 3) \quad (\text{规格化表示法}) \\ &= M \times R^E \end{aligned}$$



**S :**

**1**表示负数  
**0**表示正数

**E :**  
含阶符的阶码,  
采用移码方式来  
表示正负指数

**M:**

尾数,小数点放  
在尾数域最前面

**真值与浮点机器数的相互转化**——根据 IEEE754 标准，规格化表示原则如下：

1、尾数 M 最高有效位为 1，隐藏，并且隐藏在小数点的左边（即： $1 \leq M < 2$ ）

2、32 位单精度浮点数规格化表示  $x = (-1)^s \times (1.M) \times 2^{E-127}$

$$e = E - 127 \text{ (即, } E = e + 127)$$

3、64 位双精度浮点数规格化表示  $x = (-1)^s \times (1.M) \times 2^{E-1023}$

$$e = E - 1023 \text{ (即, } E = e + 1023)$$

指数真值 e 用偏移码形式表示为阶码 E。

例 1：浮点机器数  $(41360000)_{16}$ , 求真值。

① 十六进制数展开成二进制数：0100 0001 0011 0110 0000 0000 0000

0000

0	100 0001 0	011 0110 0000 0000 0000
S(1 位)	阶码 E(8 位)	尾数 M(23 位)

② 指数  $e = \text{阶码 } E - 127 = 1000 0010 - 01111111 = 00000011 = (3)_{10}$

③ 包括隐藏位 1 的尾数  $1.M = 1.011011$

④  $X = (-1)^s \times 1.M \times 2^e = + (1.011011) \times 2^3 = + 1011.011 = (11.375)_{10}$

例 2：真值  $20.59375$ , 求 32 位单精度浮点数。

① 分别将整数和分数部分转换成二进制数。 $20.59375 = 10100.10011$

② 移动小数点，使其在第 1、2 位之间。

$$10100.10011 = 1.010010011 \times 2^4$$

$$e = 4; S = 0; E = 4 + 127 = 131 = 10000011; M = 010010011$$

③ 得到 32 位浮点数的二进制存储格式为：

S(1 位)	阶码 E(8 位)	尾数 M(23 位)
--------	-----------	------------



0	10000011	010010011000000000000000
$0100\ 0001\ 1010\ 0100\ 1100\ 0000\ 0000\ 0000 = (41A4C000)_{16}$		
<b>二、表示范围</b>		
<u>32位单精度浮点数表示范围:</u>		
$E=1\ (0000\ 0001) \sim 254\ (1111\ 1110)$ ; E全为1是用来通知异常情况。		
$e=-126 \sim +127$		
表达的数据范围(绝对值) :		
最小值: $e=-126, M=0\ (1.M=1)$		
十进制表达: $2^{-126} \approx 1.18 \times 10^{-38}$		
最大值: $e=127, M=11\dots1\ (23\ 个1)$		
$1.M=1.11\dots1\ (23\ 个1) = 2 - 2^{-23}$		
十进制表达: $(2 - 2^{-23}) \times 2^{127} \approx 2 \times 2^{127} \approx 3.40 \times 10^{38}$		
<u>64位单精度浮点数表示范围:</u>		
$E=1 \sim 2046$ ; E全为1是用来通知异常情况。		
$e=-1022 \sim +1023$		
表达的数据范围(绝对值) :		
最小值: $e=-1022, M=0\ (1.M=1)$		
十进制表达: $2^{-1022} \approx 2.23 \times 10^{-308}$		
最大值: $e=1023, M=11\dots1\ (52\ 个1)$		
$1.M=1.11\dots1\ (52\ 个1) = 2 - 2^{-52}$		
十进制表达: $(2 - 2^{-52}) \times 2^{1023} \approx 2 \times 2^{1023} \approx 1.79 \times 10^{308}$		

### 3. ASCII 码表

高四位		ASCII控制字符								ASCII打印字符								ASCII打印字符														
		0000				0001				0010				0011				0100				0101				0110						
十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl	代码	转义字符	字符解释	十进制	字符	Ctrl
0000 0 0	\0	^@	NUL	\0	空字符	16	▶	^P	DLE	数据链路转义	32		48	0	64	@	80	P	96	`	112	P										
0001 1 1	☺	^A	SOH		标题开始	17	◀	^Q	DC1	设备控制 1	33	!	49	1	65	A	81	Q	97	a	113	q										
0010 2 2	⊕	^B	STX		正文开始	18	↑	^R	DC2	设备控制 2	34	"	50	2	66	B	82	R	98	b	114	r										
0011 3 3	♥	^C	ETX		正文结束	19	!!	^S	DC3	设备控制 3	35	#	51	3	67	C	83	S	99	c	115	s										
0100 4 4	◆	^D	EOT		传输结束	20	¶	^T	DC4	设备控制 4	36	\$	52	4	68	D	84	T	100	d	116	t										
0101 5 5	♣	^E	ENQ		查询	21	§	^U	NAK	否定应答	37	%	53	5	69	E	85	U	101	e	117	u										
0110 6 6	♦	^F	ACK		肯定应答	22	—	^V	SYN	同步空闲	38	&	54	6	70	F	86	V	102	f	118	v										
0111 7 7	•	^G	BEL	ba	响铃	23	↓	^W	ETB	传输块结束	39	‘	55	7	71	G	87	W	103	g	119	w										
1000 8 8	█	^H	BS	lb	退格	24	↑	^X	CAN	取消	40	(	56	8	72	H	88	X	104	h	120	x										
1001 9 9	○	^I	HT	lt	横向制表	25	↓	^Y	EM	介质结束	41	)	57	9	73	I	89	Y	105	i	121	y										
1010 A 10	█	^J	LF	ln	换行	26	→	^Z	SUB	替代	42	*	58	:	74	J	90	Z	106	j	122	z										
1011 B 11	♂	^K	VT	lv	纵向制表	27	←	^I	ESC	le	溢出	43	+	59	;	75	K	91	l	107	k	123	{									
1100 C 12	♀	^L	FF	lf	换页	28	█	^S	文件分隔符	44	,	60	<	76	L	92	\	108	l	124												
1101 D 13	♪	^M	CR	rv	回车	29	↔	^A	GS	组分隔符	45	-	61	=	77	M	93	]	109	m	125	}										
1110 E 14	♫	^N	SO		移出	30	▲	^A	RS	记录分隔符	46	.	62	>	78	N	94	^	110	n	126	~										
1111 F 15	♫	^O	SI		移入	31	▼	^A	US	单元分隔符	47	/	63	?	79	O	95	_	111	o	127	□	^Backspace 代码 : DEL									

注：表中的ASCII字符可以用“Alt + 小键盘上的数字键”方法输入。

2013/08/08



## 4. scanf () 与 printf () 的格式说明符

%d	用来输入和输出int
%ld	用来输入和输出long int
%lld	用来输入和输出long long int
%hd	用来输入和输出short
%i	用来输入和输出有符号十进制整数
%u	用来输入和输出无符号十进制整数
%lu	用来输入和输出无符号十进制长整数
%llu	用来输入和输出无符号十进制长长整数
%hu	用来输入和输出无符号短十进制整数
%o	用来输入和输出八进制整数
%lo	用来输入和输出长八进制整数
%ho	用来输入和输出短八进制整数
%#o	用来输出八进制整数，数字前有0
%x	用来输入和输出十六制整数，字母小写
%#x	用来输出十六制整数，字母小写，数字前有0x
%lx	用来输入和输出长十六制整数，字母小写
%X	用来输入和输出十六制整数，字母大写
%#X	用来输出十六制整数，字母大写，数字前有0X
%lX	用来输入和输出长十六制整数，字母大写

%c	用来输入和输出单个字符
%s	用来输入和输出一串字符串 输入时遇空格，制表符或换行符结束 输出时连格式说明符一起输出 <code>printf ("%s", "%d%f", a, b);</code> ;输出 %d%f
%f	用来输入和输出float，输出double
%lf	用来输入和输出double (double输出用%f和%lf都可以)
%Lf	用来输入和输出long double
%e	用来输入和输出指数，字母小写
%le	用来输入和输出长指数，字母小写
%E	用来输入和输出指数，字母大写
%lE	用来输入和输出长指数，字母大写
%g	用来输入和输出指数或float (输出最短的一种)，字母小写
%lg	用来输入和输出长指数或double (输出最短的一种)，字母小写
%G	用来输入和输出指数或float (输出最短的一种)，字母大写
%lG	用来输入和输出长指数或double (输出最短的一种)，字母大写

## scanf () 独有格式说明符

%* (所有类型) , 如%*d	用来输入一个数，字符或字符串而不赋值 (跳过无关输入) 如 <code>scanf("%d%*c%d", &amp;a, &amp;b);</code> 这样就可以只将1+2中的1和2赋值给a和b。
%m (所有类型) , 其中m为常数	限定输入范围，如 <code>scanf ("%4d", &amp;a)</code> 时输入123456，只把1234赋值给a
, (逗号)	无实际用处，仅用于美观。如 <code>scanf ("%d,%d,%d", &amp;a, &amp;b, &amp;c);</code>
- (横杠) : (冒号)	方便日期等输入，但不赋值 <code>scanf ("%-d-%d-%d", &amp;a, &amp;b, &amp;c);</code> ;需输入2018-11-20 <code>scanf ("%d: %d: %d": &amp;a, &amp;b, &amp;c);</code> ;需输入2018:11:20
所有字符串，符号 (包括空格) 数字 (不与输入数相接)	任何所写的东西都必须如横杠一般先输入 (不赋值)，不然系统报错 <code>scanf ("%d 456 %d", &amp;a, &amp;b);</code> 需输入 1 456 7 (1和7之间有456 (前后各一个空格)) 结果为a=1 b=7



## printf () 独有格式说明符

%m.nd % -m.nd (m和n为常数)	m用于在d位数小于m时补空格(右对齐) d位数大于m时忽略 如%5d, 输出123, _123 (123前面两个空格)  .n用于在d位数小于n时补0(右对齐) d位数大于n时忽略 如%.5d, 输出123, 00123 (123前面两个0)  %-m.nd则为左对齐
%m.nf %m.nlf %m.nLf %-m.nf %-m.nlf %-m.nLf	m用于在小数位数小于m时补空格(右对齐) 小数位数大于m时忽略 (小数点算一位) 如%6f 需输出3.14 结果为_3.14 (3.14前面两个空格)  .n用于控制小数位数 小数部分长度大于n则四舍五入 小数部分长度小于n则补0 如%.6f 需输出3.14 结果为3.140000 如%6f 需输出3.1415926 结果为3.141593  %-m.nf %-m.nlf %-m.nLf 则为左对齐
%m.ns % -m.ns	m用于在字符串位数小于m时补空格(右对齐) 字符串位数大于m时忽略 如%5s, 输出abc, _abc (abc前面两个空格)  .n用于控制字符串位数 长度大于n则仅输出前n位 字符串长度小于n时忽略 如%.6s 需输出abcdefg 结果为abcdef 如%6f 需输出abc 结果为abc  %-m.ns则为左对齐
%mc % -mc	m限制char的输出长度 当m>1时, 在左方补m-1个空格 %-mc则为左对齐
%m.ne % -m.ne %m.nE % -m.nE	m用于控制指数长度, 在QT中, 指数部分占五位(如 e+001) 位数小于m时左方补空格 位数大于m时忽略 如printf("%15.5e",a); 设a为123.456789 结果为 _1.23457e+002 (三个空格)  .n用于控制小数长度 小数部分长度大于n则四舍五入 小数部分长度小于n则补0 如printf("%15.5e",a); 设a为123.456789 结果为 _1.23457e+002 (三个空格)  %-m.ne %-m.nE则为左对齐
%* (整型) 如%*d %-* (整型)	在输出项中规定整型数据的宽度, 少于限制补空格, 大于忽略 如printf (" %*d" ,a,b) ; a=5 b=123 结果为 _123 (前面有两个空格)  %-* (整型) 左对齐
%0* (整型)	在输出项中规定整型数据的宽度, 少于限制补0, 大于忽略 如printf (" %0*d" ,a,b) ; a=5 b=123 结果为 00123
注意	printf () 中的运算是从右至左, 而输出是从左至右 如a=1, printf (" %d %d %d" ,a++,a++,a++) ; 结果为3 2 1

## 5. 转义字符表

转义字符	意义	ASCII 码值 (十进制)
\a	响铃(BEL)	007
\b	退格(BS) , 将当前位置移到前一列	008
\f	换页(FF), 将当前位置移到下页开头	012
\n	换行(LF) , 将当前位置移到下一行开头	010
\r	回车(CR) , 将当前位置移到本行开头	013
\t	水平制表(HT) (跳到下一个 TAB 位置)	009
\v	垂直制表(VT)	011
\\"	代表一个反斜线字符"\\"	092
\'	代表一个单引号 (撇号) 字符	039
\"	代表一个双引号字符	034
\?	代表一个问号	063
\0	空字符(NUL)	000
\ddd	1 到 3 位八进制数所代表的任意字符	三位八进制
\xhh	1 到 2 位十六进制所代表的任意字符	十六进制

注意：区分，斜杠：“/”与 反斜杠：“\”，此处不可互换

