赵老师 微信

C++

1.1 第一个C++程序

```
#include<iostream>
using namespace std;
int main() {

    cout << "Hello world" << endl;

    return 0;
}
```

解释：cout 输入语句 在黑屏上输出。

1.3 数据的输入

作用：用于从键盘获取数据

关键字：cin

语法： `cin >> 变量`

示例：

```
#include<iostream>
using namespace std;
int main() {
    //变量的定义
    //语法：数据类型  变量名 = 初始值
    int a;
    int b;
    cin>>a;
    cin>>b;
    cout << a+b << endl;
    return 0;
}
```

1.4 问题：简易计算器

参考程序：

示例：

```
#include<iostream>
using namespace std;
int main()
{
    int a;
    int b;
    cout<<"请输入第一个数";
    cin>>a;
    cout<<"请输入第二个数";
    cin>>b;

    cout<<"a+b="<<a+b<<endl;
    cout<<"a-b="<<a-b<<endl;
    cout<<"a*b="<<a*b<<endl;
    cout<<"a/b="<<a/b<<endl;

    return 0;
}
```

> 注意 ： 除法 整数 除以 整数 还是一个整数，去掉小数部分的整数。

有几种小盒子　　常用变量类型

int　 存储 整数 的小盒子　　`int a = 10;`

long long　 存储 很大整数 的小盒子　　`long long a = 100000000000;`

float 存储 小数 的小盒子　　`float a = 1.5;`

double 存储 很大小数 的小盒子　　`double a = 123456789.12345;`

char 存储 字符 的小盒子　　`char ch = 'a';`

> 注意1：在显示字符型变量时，用单引号将字符括起来，不要用双引号

> 注意2：单引号内只能有一个字符，不可以是字符串

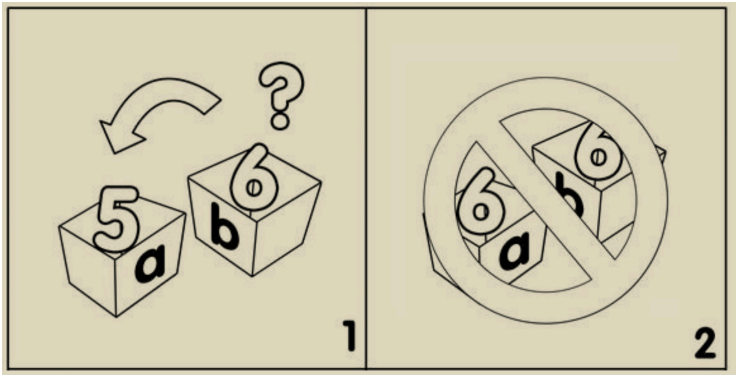
示例：

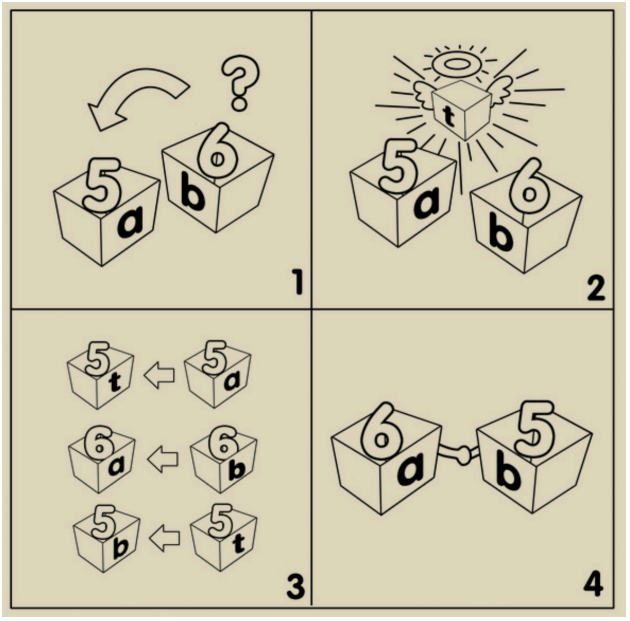
```
#include<iostream>
using namespace std;
int main() {

    char ch = 'a';
    cout << ch << endl;

    //ch = "abcde"; //错误，不可以用双引号
    //ch = 'abcde'; //错误，单引号内只能引用一个字符

    return 0;
}
```

逻辑挑战 1：交换小盒子中的数

下面的程序是否能实现交换两个数：

```
#include<iostream>
using namespace std;
int main() {
    int a = 5, b = 10;
    a=b;
    b=a;
    cout<<a<<" "<<b;

    return 0;
}
```





完整实现代码：

```cpp
#include<iostream>
using namespace std;
int main() {
    int a = 5, b = 10, t;
    t = b;
    b = a;
    a = t;
    cout<<a<<" "<<b;

    return 0;
}
```

另一种实现方式：

```cpp
#include<iostream>
using namespace std;
int main() {
    int a = 5, b = 10;
    a=b-a;
    b=b-a;
    a=b+a;
    cout<<a<<" "<<b;

    return 0;
}
```

注释

作用：在代码中加一些说明和解释，方便自己或其他程序员程序员阅读代码

两种格式：

1. **单行注释**：`// 描述信息`
2. **多行注释**：`/* 描述信息 */`

> 提示：编译器在编译代码时，会忽略注释的内容

单行注释示例：快捷键：ctrl + / 注释，再按一次 取消注释

```
#include<iostream>
using namespace std;
int main() {
    int a;
    a=1;
    //a=2;
    //a=3;
    //a=4;
    //a=5;
    cout<<a;// 输出变量 a 的值

    return 0;
}
```

多行注释示例：

```
#include<iostream>
using namespace std;
int main() {
    int a;
    a=1;
    /*
    a=2;
    a=3;
    a=4;
    a=5;
    */
    cout<<a;

    return 0;
}
```

1.3 比较运算符

作用：用于表达式的比较，并返回一个真值或假值

比较运算符有以下符号：

运算符	术语	示例	结果
==	相等	4 == 3	0
!=	不等于	4 != 3	1
<	小于	4 < 3	0
>	大于	4 > 3	1
<=	小于等于	4 <= 3	0
>=	大于等于	4 >= 1	1

示例：

```
#include <iostream>
using namespace std;
int main() {

    int a = 10;
    int b = 20;

    cout << (a == b) << endl; // 0

    cout << (a ≠ b) << endl; // 1

    cout << (a > b) << endl; // 0

    cout << (a < b) << endl; // 1

    cout << (a ≥ b) << endl; // 0

    cout << (a ≤ b) << endl; // 1
    return 0;
}
```

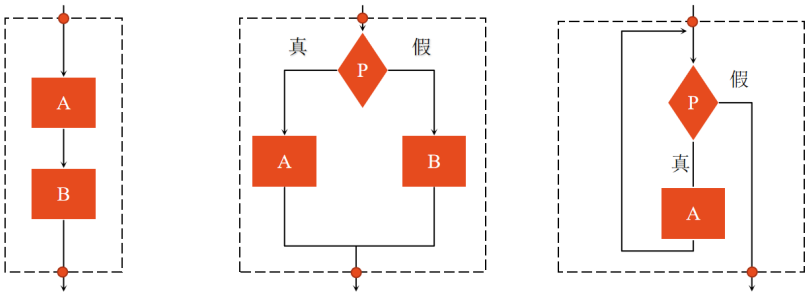
> 注意：C和C++ 语言的比较运算中，“真”用数字“1”来表示，“假”用数字“0”来表示。

2.0 程序流程结构

C/C++支持最基本的三种程序运行结构：==顺序结构、选择结构、循环结构==

- 顺序结构：程序按顺序执行，不发生跳转
- 选择结构：依据条件是否满足，有选择的执行相应功能
- 循环结构：依据条件是否满足，循环多次执行某段代码

三种基本结构



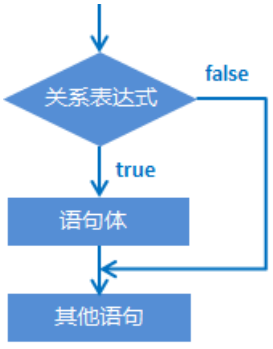
顺序结构　选择结构　循环结构

2.1 选择结构

2.2 if语句

作用：执行满足条件的语句

单行格式if语句：`if(条件){ 条件满足执行的语句 }`

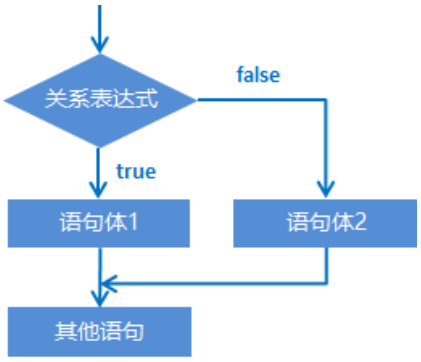


示例：

```
#include<iostream>
using namespace std;
int main() {
    int score=0;
    cout << "请输入一个分数: " << endl;
    cin >> score;

    if (score >= 488)
    {
        cout << "恭喜你考上了一本大学！！！" << endl;
    }

    return 0;
}
```

> 注意：if条件表达式后不要加分号

if else 语句：`if(条件){ 条件满足执行的语句 }else{ 条件不满足执行的语句 }`



示例：

```
#include<iostream>
using namespace std;
int main() {
```

```
    int score = 0;
    cout << "请输入考试分数: " << endl;
    cin >> score;
    if (score ≥ 488)
        {
            cout << "恭喜你考上了一本大学！！！" << endl;
        }
    else
        {
            cout << "未考上一本大学" << endl;
        }

    return 0;
}
```

如果语句是一句话，可以没有大括号：

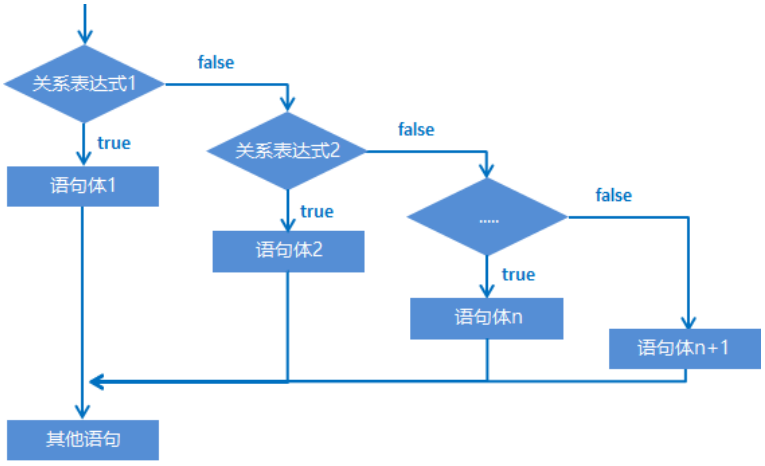
```
#include<iostream>
using namespace std;
int main() {
    int score = 0;
    cout << "请输入考试分数: " << endl;
    cin >> score;
    if (score ≥ 60)
        cout << "恭喜你及格了！！！" << endl;
    else
        cout << "很可惜，你没有及格!" << endl;

    return 0;
}
```

{ }大括号的用法：两句或者两句以上，一定要用大括号括起来，使两个语句绑定在一起。

```
#include<iostream>
using namespace std;
int main() {
    int score = 0;
    cout << "请输入考试分数: " << endl;
    cin >> score;
```

```
    if (score ≥ 60)
        {
            cout << "恭喜你及格了！！！" << endl;
            cout << "放假可以出去玩啦！" << endl;
        }
    else
        {
            cout << "很遗憾，你没有及格！" << endl;
            cout << "放假在家好好学习，不能出去！" << endl;
         }


    return 0;
}
```

多条件的if语句：`if(条件1){ 条件1满足执行的语句 }else if(条件2){条件2满足执行的语句} ... else{ 都不满足执行的语句}`



示例：

```
#include<iostream>
using namespace std;
int main() {
    int score = 0;
    cout << "请输入考试分数: " << endl;
    cin >> score;

    if (score ≥ 488)
        {
```

```
            cout << "恭喜你：考上了一本大学" << endl;
        }
    else if (score ≥ 415)
            {
                cout << "恭喜你：考上了二本大学" << endl;
            }
            else if (score ≥ 200)
                {
                    cout << "恭喜你：考上了三本大学" << endl;
                }
                else
                {
                    cout << "恭喜你：考上了专科" << endl;
                }

    return 0;
}
```

课堂练习： 对学生成绩进行分级

100 ~ 90 包括90分 A

90 ~ 80 包括80分 B

80 ~ 70 包括70分 C

70 ~ 60 包括60分 D

60分以下 F

```
#include<iostream>
using namespace std;
int main() {

    int score = 0;
    cout << "请输入考试分数：" << endl;
    cin >> score;

    if (score ≥ 90)
        {
            cout << "A" << endl;
        }
    else if (score ≥80)
            {
                cout << "B" << endl;
            }
        else if (score ≥ 70)
```

```
                {
                    cout << "C" << endl;
                }
            else if (score ≥ 60)
                    {
                        cout << "D" << endl;
                    }
                else
                    {
                        cout << "F" << endl;
                    }

    return 0;

}
```

嵌套if语句：在if语句中，可以嵌套使用if语句，达到更精确的条件判断

示例：

```
#include<iostream>
using namespace std;
int main() {
    int score = 0;
    cout << "请输入考试分数: " << endl;
    cin >> score;

    if (score ≥ 488)
    {
        if (score ≥ 600)
            {
                cout << "恭喜你: 考上了清北复交" << endl;
            }
```

```
        else
            {
                cout << "恭喜你：考上一般本科" << endl;
            }
    }
    else
    {
        cout << "未考上了一本大学" << endl;
    }

    return 0;
}
```

嵌套

```
#include<iostream>
using namespace std;
int main() {
    int a,b,c;
    cin>>a>>b>>c;
    if(a≥b)
        if(a≥c)
            cout≪a;
        else
            cout≪c;
    else
        if(b≥c)
            cout≪b;
        else
            cout≪c;
    return 0;
}
```

else 和他上面最近的一个 if 结合。 缩进格式的重要性！

```
if()
    if()  语句1
else
    if()  语句2
else      语句3
```

```
if ()
   if () 语句 1
   else if () 语句 2
        else  语句 3
```

2.3 三目运算符

作用： 通过三目运算符实现简单的判断

语法： `表达式1 ？ 表达式2 ： 表达式3`

```
if (a>b)
    max=a;
else
    max=b;
```

```
max=(a>b) ? a : b;
```

或

```
a>b ? (max=a) : (max=b);  //表达式2和表达式3是赋值表达式
```

解释：

如果表达式1的值为真，执行表达式2，并返回表达式2的结果；

如果表达式1的值为假，执行表达式3，并返回表达式3的结果。

```
#include<iostream>
using namespace std;
int main() {

    int a = 10;
    int b = 20;

    (a > b ? a : b) = 100;

    cout << "a = " << a << endl;
    cout << "b = " << b << endl;

    return 0;
}
```

> 总结：和if语句比较，三目运算符优点是短小整洁，缺点是如果用嵌套，结构不清晰

示例2：

```
#include <iostream>
using namespace std;
int main()
{
    int a, b;
    cin >> a >> b;
    a > b ? cout << a : cout << b;
    return 0;
}
```

题目： 输入一个整数，这个数如果是奇数则输出“奇数”，否则输出“偶数”。

（使用三目运算符完成）

参考程序：

示例3：

```cpp
#include <iostream>
using namespace std;
int main()
{
    int a;
    cin >> a;
    a%2==1 ? cout << "奇数" << endl: cout << "偶数" << endl;
    return 0;
}
```

> 比较的结果： C++ 只认识 0 和 非 0， 0 即是假 是 错， 非 0 即是 真 是 对

1.0

```cpp
#include<iostream>
using namespace std;
int main() {
    if (1>2)
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

2.0

```cpp
#include<iostream>
using namespace std;
int main() {
    if (1)
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

3.0

```
#include<iostream>
using namespace std;
int main() {
    if (-5)
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

4.0

```
#include<iostream>
using namespace std;
int main() {
    if (0)
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

5.0

```
#include<iostream>
using namespace std;
int main() {
    if ('A')
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

6.0

```
#include<iostream>
using namespace std;
int main() {
    if (3.1415926)
        cout<<"yes";
    else
        cout<<"no";

    return 0;

}
```

2.4 switch语句

作用：执行多条件分支语句

语法：switch 多分支选择关键字 。() 里面是一个变量，变量对应case 后面的数字，匹配就执行数字后面的语句，否则不执行。break 意思是破坏循环，直接跳出switch语句。

```
switch(表达式)

{
```

```
    case 结果1: 执行语句;break;

    case 结果2: 执行语句;break;

     ...

    default:执行语句;break;

}
```

题目：根据从键盘上输入的数字，输出星期几。

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
      int weekday;
      cin>>weekday;
      switch(weekday)
      {
           case 1: cout<<"星期一"<<endl; break;
           case 2: cout<<"星期二"<<endl; break;
           case 3: cout<<"星期三"<<endl; break;
           case 4: cout<<"星期四"<<endl; break;
           case 5: cout<<"星期五"<<endl; break;
           case 6: cout<<"星期六"<<endl; break;
           case 7: cout<<"星期日"<<endl; break;
                }
     return 0;
}
```

课堂练习：假如一个月的第一天是星期一，输入一个日期，判断星期几。

参考程序：

```
#include<iostream>
```

```
using namespace std;
int main()
{
      int weekday;
      cin>>weekday;
      switch(weekday%7)
      {
           case 1: cout<<"星期一"<<endl; break;
           case 2: cout<<"星期二"<<endl; break;
           case 3: cout<<"星期三"<<endl; break;
           case 4: cout<<"星期四"<<endl; break;
           case 5: cout<<"星期五"<<endl; break;
           case 6: cout<<"星期六"<<endl; break;
           case 0: cout<<"星期日"<<endl; break;
                }
     return 0;
}
```

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
      int weekday;
      weekday=3;
      switch(weekday)
      {
           case 1: cout<<"星期一"<<endl;
           case 2: cout<<"星期二"<<endl;
           case 3: cout<<"星期三"<<endl;
           case 4: cout<<"星期四"<<endl;
           case 5: cout<<"星期五"<<endl;
           case 6: cout<<"星期六"<<endl; break;
           case 7: cout<<"星期日"<<endl; break;
                }
     return 0;
}
```

1.2 switch语句练习-春夏秋冬（应用）

- 需求：一年有12个月，分属于春夏秋冬4个季节，键盘录入一个月份，请用程序实现判断该月份属于哪个季节，并输出。
- 运行结果：

```
春：3、4、5
夏：6、7、8
秋：9、10、11
冬：1、2、12
```

- 示例代码：

```
#include<iostream>
using namespace std;
int main()
{
	int month;
	cin>>month;
	switch(month) {
		case 1:
		case 2:
		case 12:
			cout<<"冬季"<<endl;
			break;
		case 3:
		case 4:
		case 5:
			cout<<"春季"<<endl;
			break;
		case 6:
		case 7:
		case 8:
			cout<<"夏季"<<endl;
			break;
		case 9:
		case 10:
		case 11:
			cout<<"秋季"<<endl;
			break;
		default:
			cout<<"你输入的月份有误"<<endl;
	}
    return 0;
}
```

- 注意：如果switch中得case，没有对应break的话，则会出现case穿透的现象。

题目： 对学生成绩进行分级

100 ~ 90 包括90分 A

90 ~ 80 包括80分 B

80 ~ 70 包括70分 C

70 ~ 60 包括60分 D

60分以下 F

参考程序：

```
#include <iostream>
using namespace std;
int main()
{
    int score;
    cin >> score;
    switch(score/10)
    {
        case 10:
        case 9:
            cout << "等级A";
            break;
        case 8:
            cout << "等级B";
            break;
        case 7:
            cout << "等级C";
            break;
        case 6:
            cout << "等级D";
            break;
        default:
            cout << "等级F";
            break;
    }
    return 0;
}
```

> 注意1：switch语句中表达式类型只能是整型或者字符型

> 注意2：case里如果没有break，那么程序会一直向下执行
>
> 注意3：case语句后的各常量表达式的值不能相同，否则会出现错误。
>
> 注意4：各case和default子句的先后顺序可以变动，这不会影响程序执行结果。
>
> 注意6：default子句可以省略，default后面的语句末尾可以不必写break。

> 总结：与if语句比，对于多条件判断时，switch的结构清晰，缺点是switch不可以判断区间

课堂练习：一个最简单的计算器，支持+, -, *, / 四种运算。仅需考虑输入输出为整数的情况，数据和运算结果不会超过int表示的范围。

输入样例：7 * 9

输出样例：63

参考程序：

```
#include <iostream>
using namespace std;
int main()
{
    int a, b;
    char c;
    cin >> a >> c >> b;

    switch(c)
    {
        case '+':
            cout << "a + b = " << a+b;
            break;
        case '-':
```

```
            cout << "a - b = " << a-b;
            break;
        case '/':
            cout << "a / b = " << a/b;
            break;
        case '*':
            cout << "a * b = " << a*b;
    }
    return 0;
}
```

表 3-1　运算符总结

名　称	作　用	名　称	作　用
+	加	>=	大于等于
–	减	<=	小于等于
*	乘	!=	不等于
/	除	&&	与
>	大于	||	或
<	小于	!	非
==	等于		

作业：　　判断x年是否为闰年，闰年分两种情况——整百和非整百，两种条件只需满足其一，（非整百且是4的倍数）或（整百且是400的倍数）

!= 不等于符号　==等于符号

&& 并且　|| 或者

参考程序：

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cout<<"请输入判断的年份" ;
    cin>>a;
    if((a%100≠0&&a%4==0) || (a%400==0))
        cout<<"闰年" ;
    else
        cout<<"不是闰年" ;
```

```
return 0;
}
```

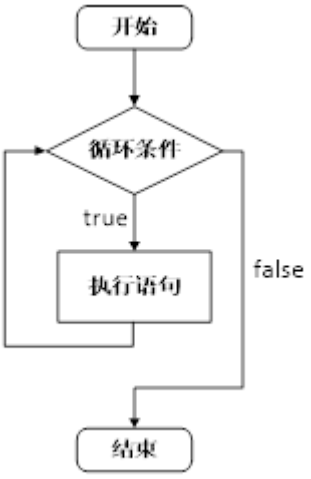
3.0 循环结构

3.1 while循环语句

作用：满足循环条件，执行循环语句

语法： `while(循环条件){ 循环语句 }`

解释：==只要循环条件的结果为真，就执行循环语句==



示例：

```cpp
#include<iostream>
using namespace std;
int main() {

    int num = 0;
    while (num < 10)
    {
        cout << "num = " << num << endl;
        num=num+1;
    }

    return 0;
}
```

> 注意：在执行循环语句时候，程序必须提供跳出循环的出口，否则出现死循环

题目：1+2+3+.....+100=?

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int a=1,sum=0;
    while(a≤100)
    {
        sum=sum+a;
        a=a+1;
    }
    cout≪sum;
    return 0;
}
```

课堂练习：1+3+……+99=?　　所有奇数相加。

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int a=1,sum=0;
    while(a≤100)
    {
        sum=sum+a;
        a=a+2;
    }
    cout≪sum;
    return 0;
}
```

课堂练习：2+4+……+100=?　　所有偶数相加。

参考程序：

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a=2,sum=0;
    while(a≤100)
    {
        sum=sum+a;
        a+=2;
    }
    cout≪sum;
    return 0;
}
```

课堂练习：2+4+……+100=?　　所有偶数相加（另一种方法）。

参考程序：

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a=2,sum=0;
    while(a≤100)
    {
       if(a%2==0)
         sum+=a;
       a++;
    }
    cout≪sum;
    return 0;
}
```

> a++;　　a=a+1;
>
> sum+=a;　　sum=sum+a;

输入一个整数 n(1<=n<=9)，求 n 的阶乘

参考程序：

```
#include<iostream>
using namespace std;
int main() {
    int a,i,n;
    a=1;
    i=1;
    cin>>n;
    while(i≤n)
    {
        a=a*i;
        i++;
    }
    cout<<a;
    return 0;
}
```

题目：请问以下程序的输出结果是多少?

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int a=1,sum=0;
    while(a≤100);
    {
        sum=sum+a;
        a++;
    }
    cout≪sum;
    return 0;
}
```

> 注意：;分号 就是一条语句，叫做空语句，什么也不干，但是起到语句的作用。

课堂练习：从1开始数到数字100， 如果数字个位含有3，或者该数字是3的倍数，我们打印鼓掌，其余数字直接打印输出。

参考程序：

```
#include<iostream>
using namespace std;
int main()
```

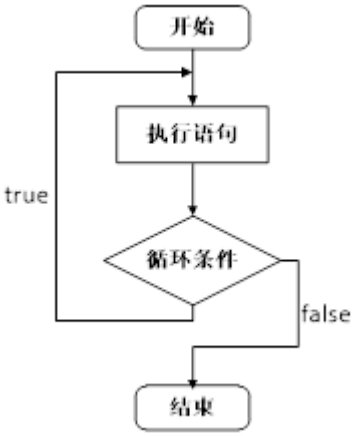
```
{
    int a=1;
    while(a≤100)
    {
        if(a%3==0||a%10==3)
            cout<<"鼓掌"<<endl;
        else
            cout<<a<<endl;
    a++;
    }
    return 0;
}
```

3.2 do...while循环语句

作用： 满足循环条件，执行循环语句

语法： `do{ 循环语句 } while(循环条件);`

注意：与while的区别在于==do...while会先执行一次循环语句==，再判断循环条件



示例：

```cpp
#include<iostream>
using namespace std;
int main() {
    int num = 0;
    do
    {
        cout << num << endl;
        num++;
    } while (num < 10);

    return 0;
}
```

> 总结：与while循环区别在于，do...while先执行一次循环语句，再判断循环条件

课堂练习：1+2+3+.....+100=?

参考程序：

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a=1,sum=0;
    do
    {
        sum=sum+a;
        a++;
    }while(a≤100);
    cout≪sum;
    return 0;
}
```

3.3 for循环语句

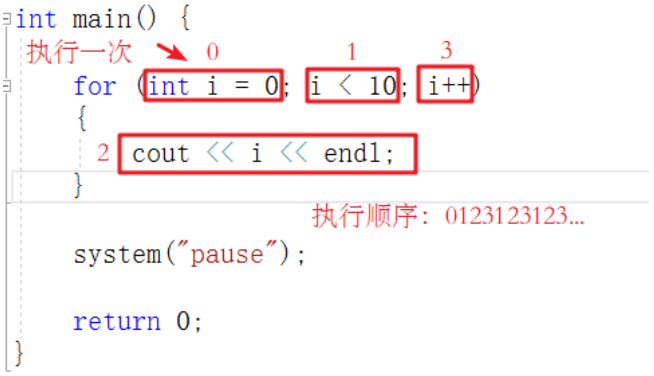
作用： 满足循环条件，执行循环语句

语法： `for(起始表达式;条件表达式;末尾循环体) { 循环语句; }`

示例：

```
#include <iostream>
using namespace std;
int main() {

    for (int i = 0; i < 10; i++)
    {
        cout << i << endl;
    }

    return 0;
}
```

详解：



> 注意：for循环中的表达式，要用分号进行分隔

> 总结：while , do...while, for都是开发中常用的循环语句，for循环结构比较清晰，比较常用

课堂练习：1+2+3+.....+100=?

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int sum=0;
    for(int a=1;a≤100;a++)
    {
       sum=sum+a;
    }
    cout≪sum;
    return 0;
}
```

课堂练习：1+2+3+.....+100=?

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int sum=0;
    int a=1;
    for(;;)
    {
        if(a>100)break;
        sum=sum+a;
        a++;
```

```
    }
    cout<<sum;
    return 0;
}
```

课堂练习：在控制台打印一排十个星号。

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int i;
    for(i=1;i≤10;i++)
    {
       cout<<"*";
    }
    return 0;
 }
```

3.2.4 嵌套循环

作用： 在循环体中再嵌套一层循环，解决一些实际问题

循环的嵌套

01

```
while()
{   ⋮
    while()
    {…}
}
```
内层循环

02

```
do
{   ⋮
    do
    {…}
    while();
}while();
```
内层循环

03

```
for( ; ; )
{
    for( ; ; )
    {…}
}
```
内层循环

04

```
while()
{   ⋮
    do
    {…}
    while();
    ⋮
}
```
内层循环

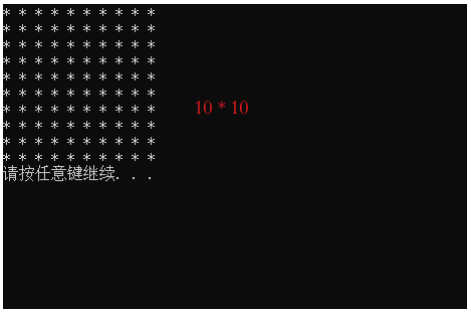
05

```
for( ; ; )
{   ⋮
    while()
    {…}
    ⋮
}
```
内层循环

06

```
do
{   ⋮
    for( ; ; )
    {…}
}while();
```
内层循环

例如我们想在屏幕中打印如下图片，就需要利用嵌套循环

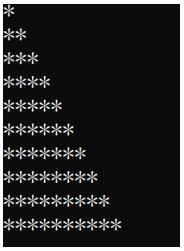


示例：

```cpp
#include<iostream>
using namespace std;
int main() {

    //外层循环执行1次，内层循环执行1轮
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < i; j++)
        {
```

```
            cout << "*" << " ";
        }
        cout << endl;
    }

    return 0;
}
```

课堂练习：在控制台打印十行十列三角形星号。



参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int i,j;
    for(i=1;i≤10;i++)
    {
        for(j=1;j≤i;j++)
           cout<<"*";
        cout<<endl;
    }
    return 0;
 }
```

课堂练习：打印乘法口诀表

```
1*1=1
2*1=2   2*2=4
3*1=3   3*2=6   3*3=9
4*1=4   4*2=8   4*3=12   4*4=16
5*1=5   5*2=10   5*3=15   5*4=20   5*5=25
6*1=6   6*2=12   6*3=18   6*4=24   6*5=30   6*6=36
7*1=7   7*2=14   7*3=21   7*4=28   7*5=35   7*6=42   7*7=49
8*1=8   8*2=16   8*3=24   8*4=32   8*5=40   8*6=48   8*7=56   8*8=64
9*1=9   9*2=18   9*3=27   9*4=36   9*5=45   9*6=54   9*7=63   9*8=72   9*9=81
```

参考程序：

```cpp
#include<iostream>
using namespace std;
int main()
{
    int i,j;
    for(i=1;i≤9;i++)
    {
        for(j=1;j≤i;j++)
        {
            cout≪i≪"*"≪j≪"="≪i*j≪"  ";
        }
        cout≪endl;
    }
    return 0;
}
```

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

参考程序：

```cpp
#include<iostream>
using namespace std;
int main()
{
    int i,j,a=1;
    for(i=1;i≤5;i++)
    {
        for(j=1;j≤i;j++)
```

```
            {
                cout<<a<<" ";
                a++;
            }
        cout<<endl;
    }
    return 0;
 }
```

课堂练习：打印斐波那契数列前20项。

参考程序：

```
#include<iostream>
using namespace std;
int main()
{
    int a,b,c,d=1;
    a=1;
    cout<<a<<endl;
    b=1;
    cout<<b<<endl;
    while(d<=18)
    {   c=a+b;
        cout<<c<<endl;
        a=b;
        b=c;

        d++;
    }
    return 0;
 }
```

课堂练习：判断一个正整数是否为质数

质数，又称为素数，指大于1的自然数，除了1和该整数自身外，无法被其他自然数整除

比1大但不是质数的数称为合数。1和0既非质数也非合数。20以内的质数有2、3、5、7、11、13、17和19。

```cpp
#include<iostream>
using namespace std;
int main()
{
    int a, count, i;
    count=0;
    cin>>a;
    for(i=2; i<a; i++)
    {
        if(a%i==0)
            {
                count++;
                break;
            }
    }
    if(count==0)
        cout<<"质数";
    else
        cout<<"合数";
    return 0;
}
```

4.3 跳转语句

4.3.1 break语句

作用: 用于跳出==选择结构==或者==循环结构==

break使用的时机：

- 出现在switch条件语句中，作用是终止case并跳出switch
- 出现在循环语句中，作用是跳出当前的循环语句
- 出现在嵌套循环中，跳出最近的内层循环语句

示例1:

```cpp
#include<iostream>
using namespace std;
```

```cpp
int main() {
    //1、在switch 语句中使用break
    cout << "请选择您挑战副本的难度: " << endl;
    cout << "1、普通" << endl;
    cout << "2、中等" << endl;
    cout << "3、困难" << endl;

    int num = 0;

    cin >> num;

    switch (num)
    {
    case 1:
        cout << "您选择的是普通难度" << endl;
        break;
    case 2:
        cout << "您选择的是中等难度" << endl;
        break;
    case 3:
        cout << "您选择的是困难难度" << endl;
        break;
    }

    return 0;
}
```

示例2:

```cpp
#include <iostream>
using namespace std;
int main() {
    //2、在循环语句中用break
    for (int i = 0; i < 10; i++)
    {
        if (i == 5)
        {
            break; //跳出循环语句
        }
        cout << i << endl;
    }
```

```
    return 0;
}
```

示例3：

```
#include <iostream>
using namespace std;
int main() {
    //在嵌套循环语句中使用break，退出内层循环
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            if (j == 5)
            {
                break;
            }
            cout << "*" << " ";
        }
        cout << endl;
    }

    return 0;
}
```

示例4

```
#include <iostream>
using namespace std;
int main() {
    int i, j;
    for (i = 0; i < 10; i++)
    {
        for (j = 0; j < 10; j++)
        {
            cout << "*" << " ";
        }
        if (i == 5)
            { break;
            }
        cout << endl;
    }
```

```
    return 0;

}
```

4.3.2 continue语句

作用：在==循环语句==中，跳过本次循环中余下尚未执行的语句，继续执行下一次循环

示例2：

```
#include <iostream>
using namespace std;
int main() {
    for (int i = 0; i < 10; i++)
    {
        if (i == 5)
        {
            continue;
        }
        cout << i << endl;
    }
    return 0;
}
```

示例：

```
#include<iostream>
using namespace std;
int main() {

    for (int i = 0; i < 100; i++)
    {
        if (i % 2 == 0)
        {
            continue;
        }
        cout << i << endl;
```

```
    }
    return 0;
}
```

> 注意：continue并没有使整个循环终止，而break会跳出循环