

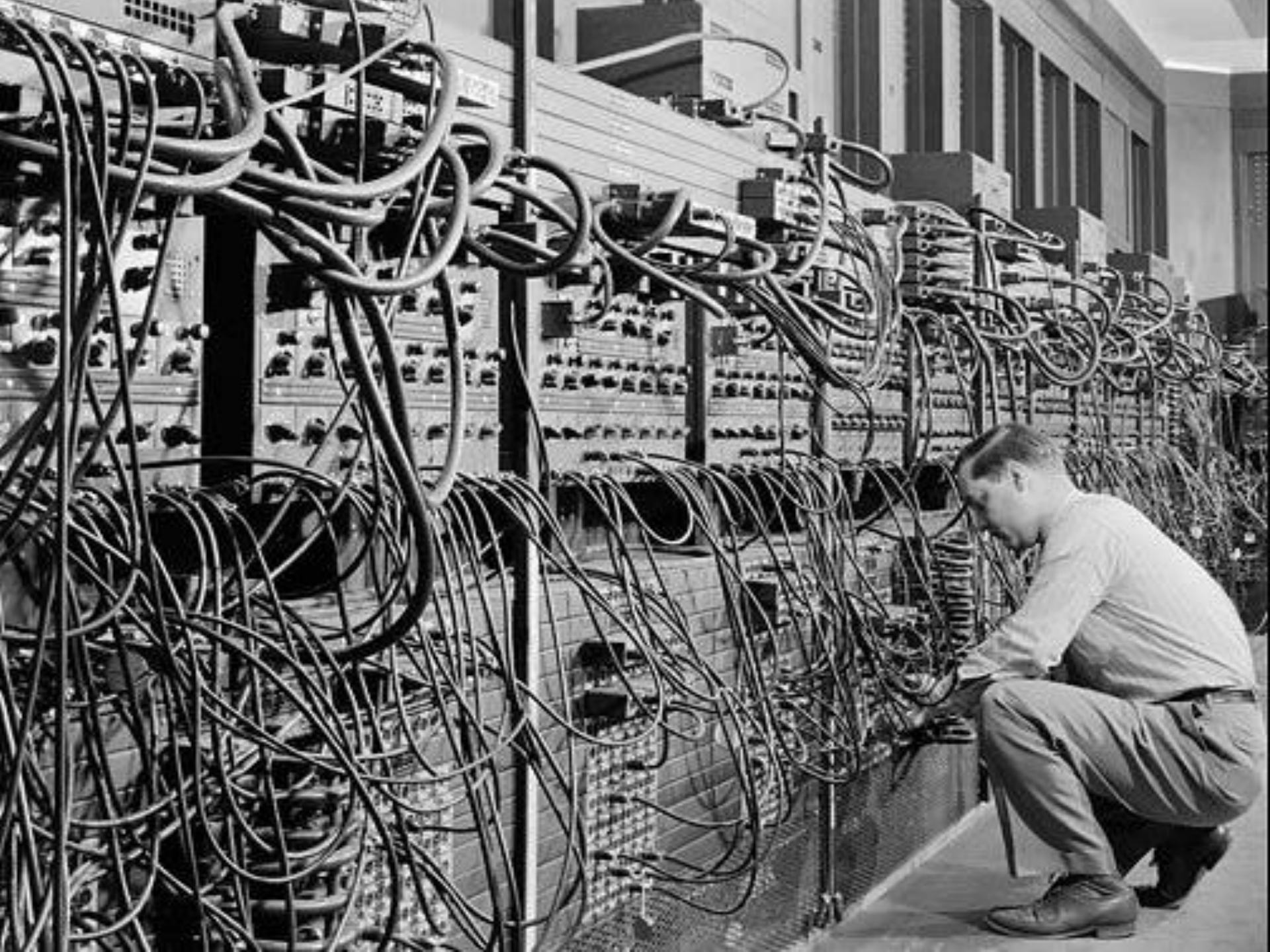
计算概论A 程序设计部分

感性认识C++程序

李 戈

北京大学 信息科学技术学院 软件研究所

lige@sei.pku.edu.cn





什么是程序？

■ 计算机也是机器！

- ◆ 必须“设置”好才能运行；

 - ENIAC采用“手工插线”的方式“编程”；

- ◆ “编程序” = 给计算机设置好运行的步骤；

■ 程序

- ◆ 人们用来告诉计算机应该做什么的东西；



清华大学

问题：怎么告诉它呢？

- ◆ **告诉计算机一些什么东西，它才能运行？**
- ◆ **以什么形式告诉它，它才能够明白？**

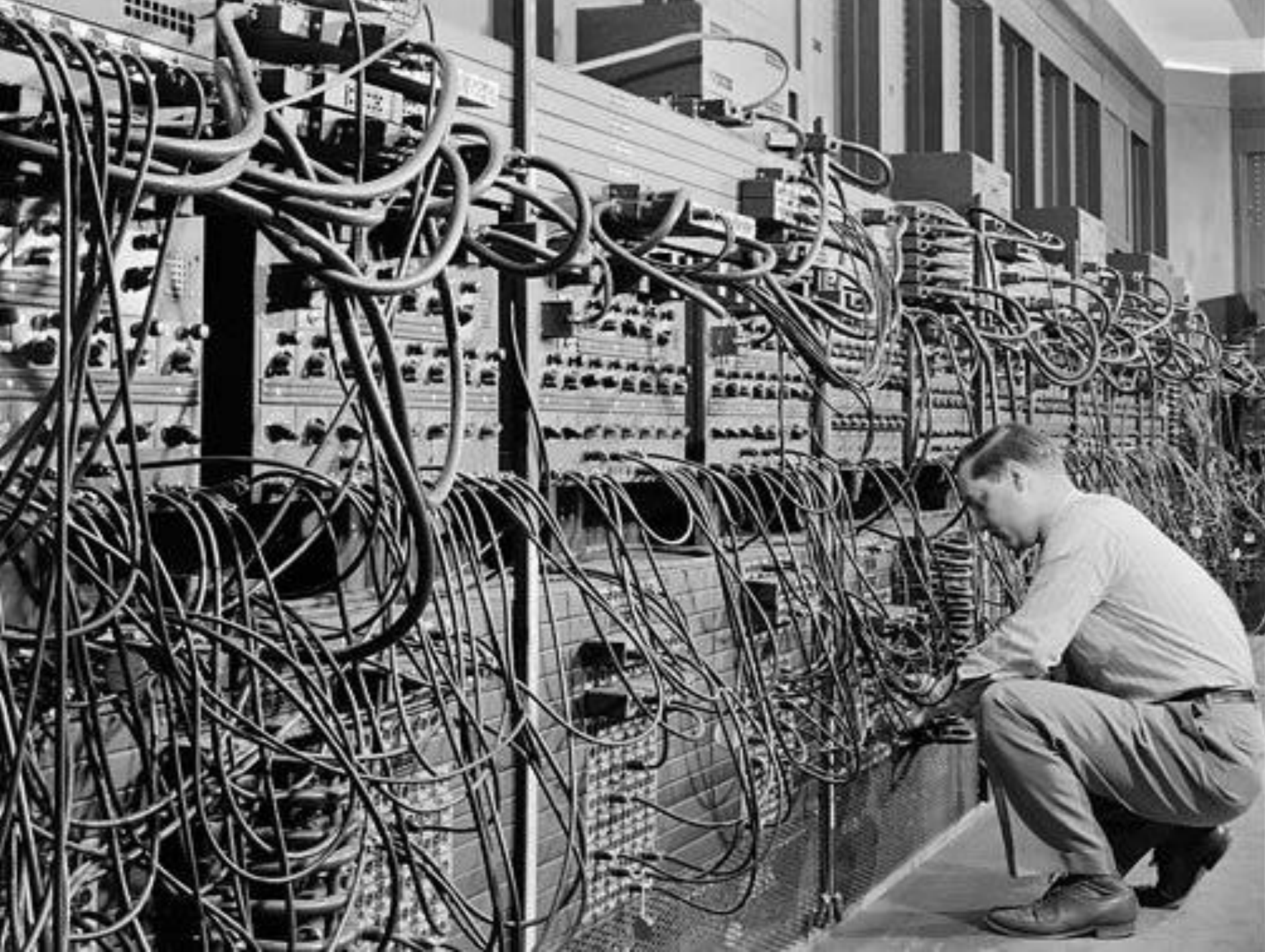
计算概论A 程序设计部分

感性认识C++程序

李 戈

北京大学 信息科学技术学院 软件研究所

lige@sei.pku.edu.cn





什么是程序？

■ 计算机也是机器！

- ◆ 必须“设置”好才能运行；

 - ENIAC采用“手工插线”的方式“编程”；

- ◆ “编程序” = 给计算机设置好运行的步骤；

■ 程序

- ◆ 人们用来告诉计算机应该做什么的东西；



北京大学

问题：怎么告诉它呢？

- ◆ **告诉计算机一些什么东西，它才能运行？**
- ◆ **以什么形式告诉它，它才能够明白？**



问题一

■ 告诉计算机一些什么东西，它才能运行？

◆ 给你一个数列：

78, 56, 69, 31, 36, 67, 31, 47, 69, 34, 45, 74, 61, 82, 43,
41, 76, 79, 81, 66, 54, 50, 76, 51, 53, 28, 74, 39, 45, 61,
52, 41, 43, 75, 78, 84, 72, 51, 43, 64, 75, 81, 69, 55, 74



北京大学



你怎么做的？

■ 哪个数字最大？

- ◆ 把某一个数字取出来，当作一个临时的“特别数字”记住，并假设这个数字最大；
- ◆ 拿这个临时的“特别数字”与其他数字相比较；
- ◆ 如果有其他数字比临时的“特别数字”更大，就把“特别的数字”换成这个更大的数字；
- ◆ 重复上述过程直到把所有的数字都比较完毕；
- ◆ 那么大脑中这个“特别数字”就记录了最大的数字；



北京大學



我的大脑这样运行...

■ 把自己的大脑当作计算机：

- ◆ 在大脑中开辟一片“存储空间”存放输入的数字；
- ◆ 使用了一个另一片“存储空间”存放“特别数字”；
- ◆ 按照某种规律“反复”选定“输入存储空间中的数字”与“特别数字”比较；
- ◆ 每次比较时，判断“选定的数字”是否大于“特别数字”；
- ◆ 如果大于，重新“刷新”“特别数字”；
- ◆ 如果“特别数字”与其他数字都进行了比较，说出“特别数字”；





把这个过程稍做整理.....

■ 更清楚的描述方式:

- ◆ 在你的大脑里开辟一片存储空间存放输入的数字;
- ◆ 开辟另一个片存储空间存放“特别数字”;
- ◆ 从存储空间中的第一个数字开始, 直到最后一个数, 重复以下操作:
 - 比较“存储空间中的数字”与“特别数字”;
 - 如果“存储空间中的数字”大于“特别数字”;
 - 那么, 将“特别数字”换成“存储空间中的数字”;
- ◆ 说出“特别数字”;



```
#include<iostream>
using namespace std;
int main( )
{
    int number[45] = {78, 56, 69, 31, 36, 67, 31, 47, 69, 34, 45, 74, 61,
        82, 43, 41, 76, 79, 81, 66, 54, 50, 76, 51, 53, 28, 74, 39, 45, 61, 52, 41,
        43, 75, 78, 84, 72, 51, 43, 64, 75, 81, 69, 55, 74};
    int max = 0;
    int i = 0;
    for(i = 0; i < 45; i++)
    {
        if(number[i] > max)
            max = number[i];
    }
    cout<<"The Maximal Number is:"<<max;
    return 0;
}
```



■ 提个要求：

如果你要 创造 一门 “程序设计语言”

你将如何回答以下的问题？



北京大学

问题1：

**是不是 无论我们在程序里写什么
“单词”，计算机都能明白？**

问题2：

**是不是 无论我们在程序里写什么
“数” 和 “计算符号”，计算机都
能明白？**

问题3：

世界上可能要用“程序来表达的逻辑”纷繁复杂，程序设计语言里面得有多少种“句式”才能够用？

问题1：

是不是 无论我们在程序里写什么“单词”，计算机都能明白？

◆ NO！

◆ 编程语言定义了一些有特定含义的“关键字”，计算机“只能明白”这些“词”的含义。



计算机能够认识的单词

| | | | | |
|----------|---------|----------|----------|--------|
| auto | break | case | char | const |
| continue | default | do | double | else |
| enum | extern | float | for | goto |
| if | int | long | register | return |
| short | signed | sizeof | static | struct |
| switch | typedef | unsigned | union | void |
| volatile | while | bool | catch | class |



```
#include<iostream>
using namespace std;
int main()
{
    enum day{Mon, Tue, Wed, Thu, Fri, Sat, Sun};
    day workDay;
    double times, wages, hourlyRate, hours;
    cout<<"Enter the hourly wages rate."<<endl;
    cin>>hourlyRate;
    cout<<"Enter hours worked daily\n";
    for (workDay=mon; workDay<=sun; workDay++)
    {
        cin>>hours; //输入周一到周日的工作时间;
        switch(workDay)
        {
            case sat: times=1.5*hours; break;
            case sun: times=2.0*hours; break;
            default: times=hours; }
        wages = wages + times*hourlyRate;
    }
    cout<<"The wages for the week are "<<wages;
    return 0;
}
```

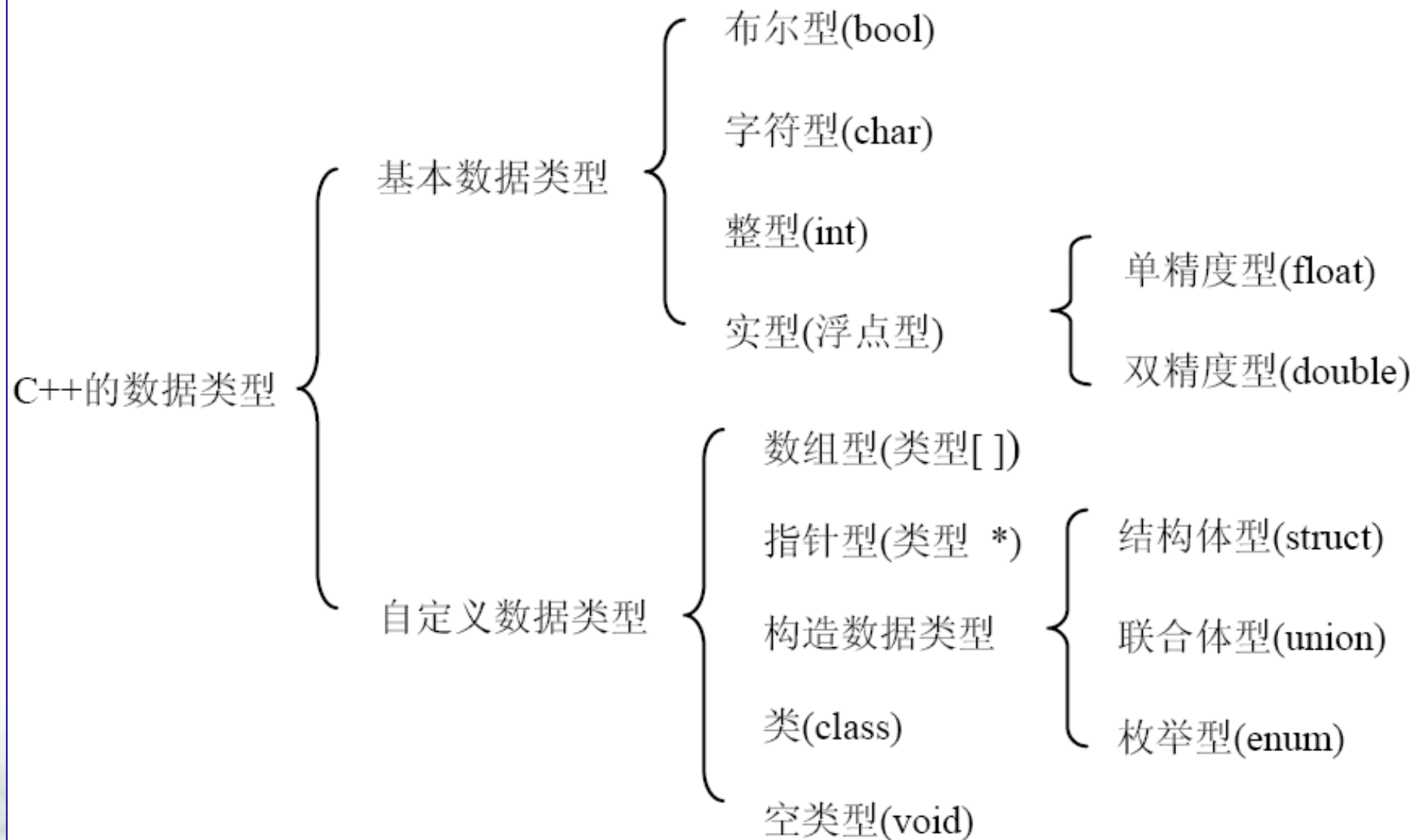

问题2：

是不是 无论我们在程序里写什么“数”和“计算符号”，计算机都能明白？

◆ NO！

◆ 计算机只能“看懂”某些类型的数据，这些“数据的类型”和相应的“操作符号”也是定义好的。

计算机能够看懂的数据的类型



计算机能够理解的运算的种类

■ C++语言的运算符

- ◆ 求字节数运算符: **sizeof**
- ◆ 下标运算符 **[]**
- ◆ 赋值运算符 **=**
- ◆ 算术运算符 **+ - * / %**
- ◆ 关系运算符 **< > == >= <= !=**
- ◆ 逻辑运算符 **! && ||**
- ◆ 条件运算符 **? :**
- ◆ 逗号运算符 **,**
- ◆ 位运算符 **>> ~ | ^ &**
- ◆ 指针运算符 ***, &**
- ◆ 强制类型转换运算符: **(类型)**
- ◆ 分量运算符 **. →**



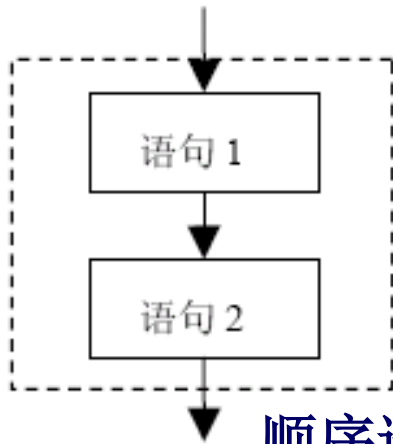
问题3：

世界上可能要用“程序来表达的逻辑”纷繁复杂，程序设计语言里面得有多少种“句式”才能够用？

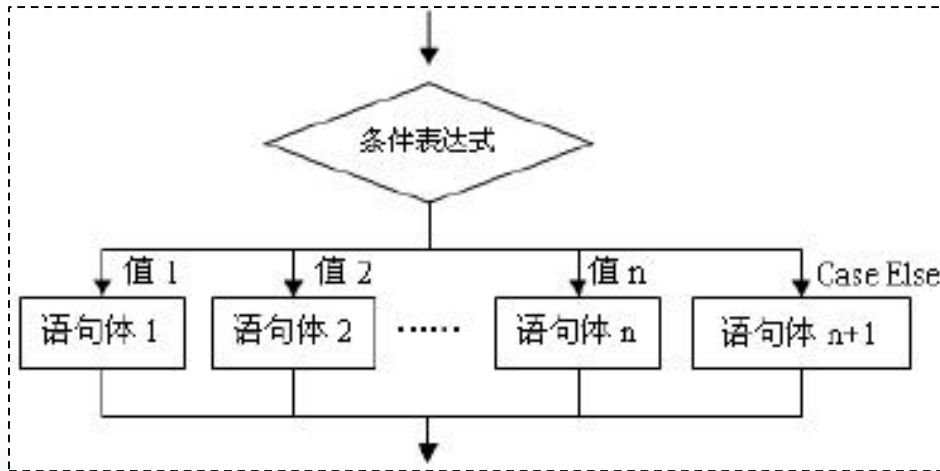
◆ 不多！三种而已！

- C. Bohm & G. Jacopini, "Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules," Communications of the ACM, vol9(5) May 1966, pp 366-371.

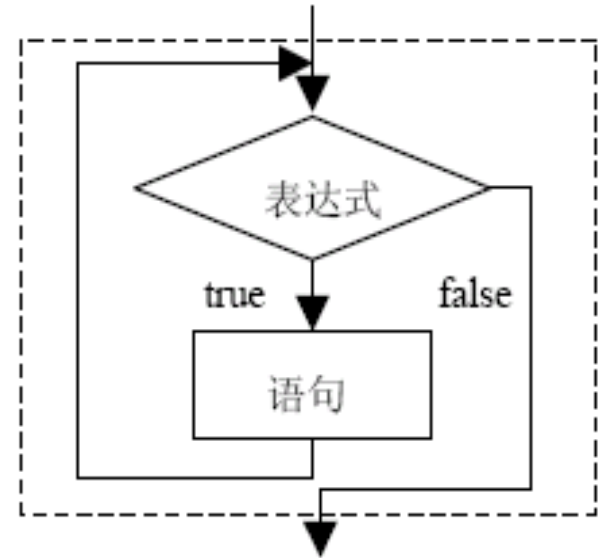
如何表达纷繁复杂的计算逻辑？



顺序语句



分支语句



循环语句



北京大学

我们要学的编程语言

30几个关键字

十几种基本数据类型 + 30几个运算符

三种基本的逻辑语句



接下来

**先把最关键的告诉你，让你开始
写程序.....**



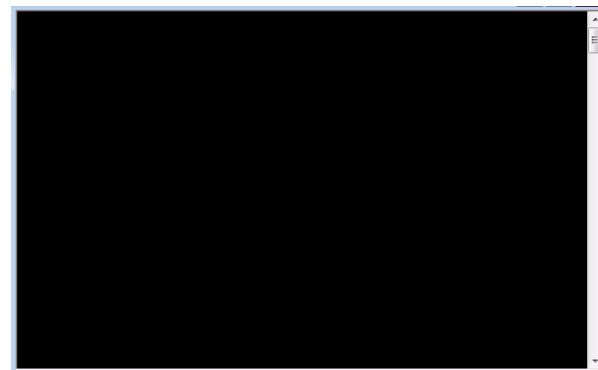
清华大学

最简单的程序

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

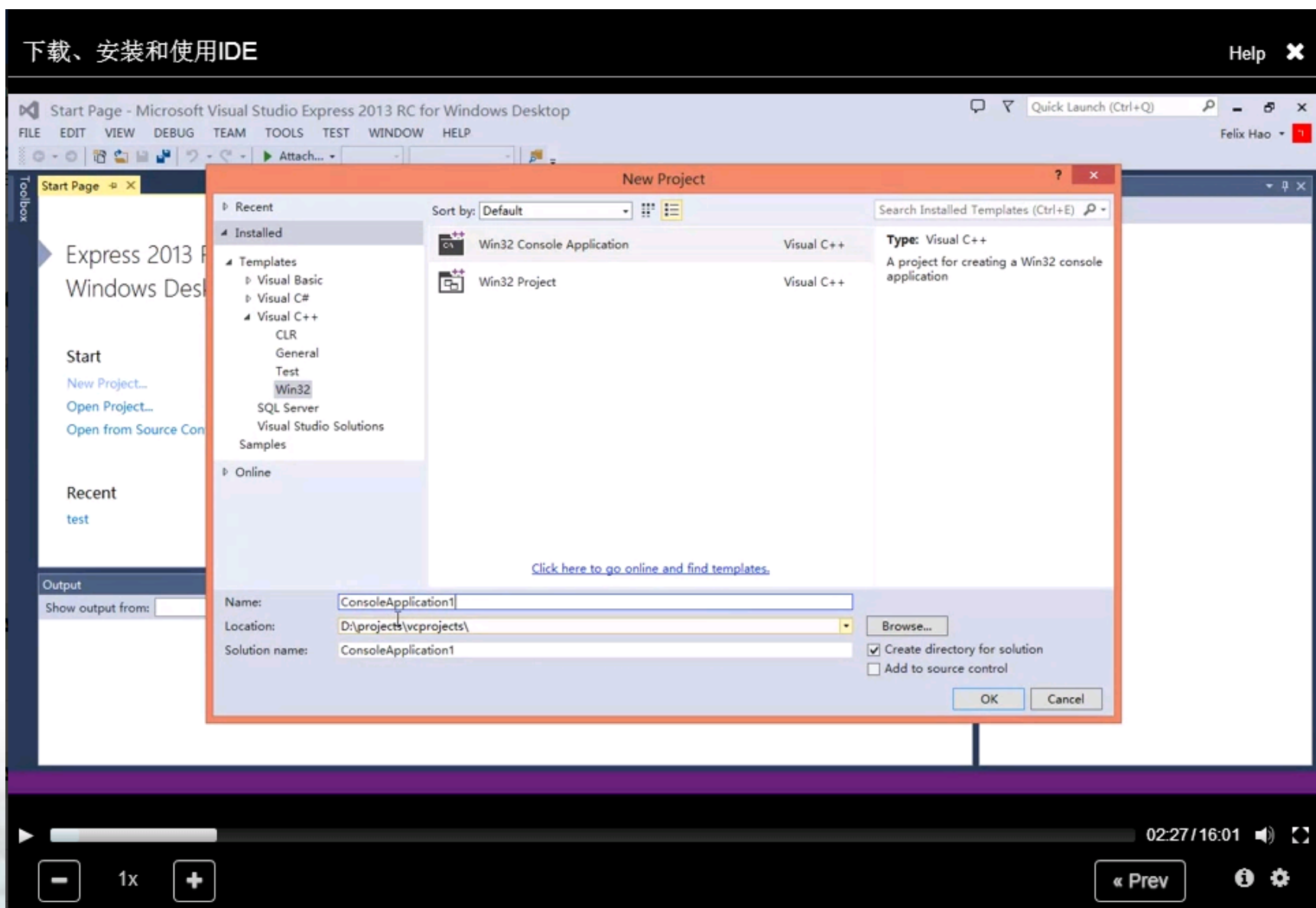
    return 0;

}
```



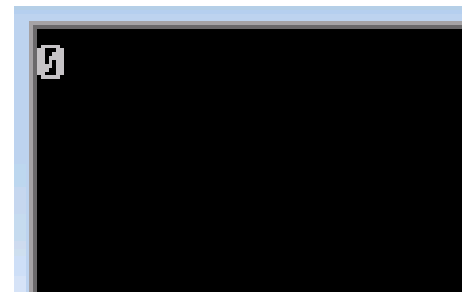
➤ 【说明】编程工具的安装

✓ 下载、安装和使用IDE



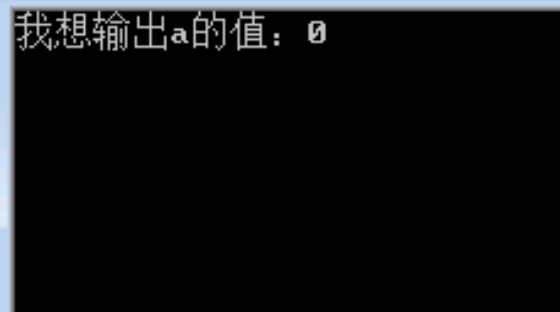
很简单的程序 (1)

```
#include <iostream>
using namespace std;
int main()
{
    int a = 0;
    cout << a << endl;
    return 0;
}
```



很简单的程序 (2)

```
#include <iostream>
using namespace std;
int main()
{
    int a = 0;
    cout<<"我想输出a的值: "<<a<<endl;
    return 0;
}
```



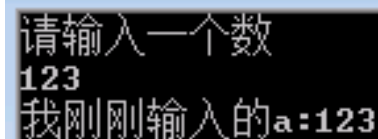
我想输出a的值: 0



北京大學

很简单的程序 (3)

```
#include <iostream>
using namespace std;
int main()
{
    int a = 0;
    cout << "请输入一个数" << endl;
    cin >> a;
    cout << "我刚刚输入的a:" << a << endl;
    return 0;
}
```

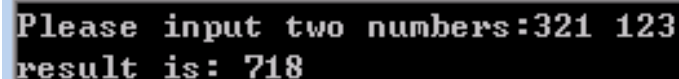


请输入一个数
123
我刚刚输入的a:123



简单的程序（1）——顺序结构

```
#include <iostream>
using namespace std;
int main()
{
    int a = 0, b = 0, result = 0;
    cout << "Please input two numbers:";
    cin >> a >> b;
    result = 3*a-2*b+1;
    cout << "result is : " << result << endl;
    return 0;
}
```

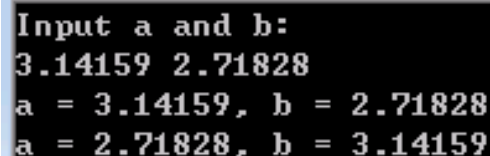


Please input two numbers:321 123
result is: 718



简单的程序 (2) —— 顺序结构

```
#include<iostream>
using namespace std;
int main()
{
    float a = 0, b = 0, temp = 0;
    cout << "Input a and b:"<<endl;
    cin >> a >> b;
    cout << "a = "<< a << ", b = " << b<<endl;
    temp = a; a = b; b = temp;
    cout << "a = " << a << ", b = " << b<<endl;
    return 0;
}
```



```
Input a and b:
3.14159 2.71828
a = 3.14159, b = 2.71828
a = 2.71828, b = 3.14159
```



简单的程序 (3) —— 分支结构

```
#include<iostream>
using namespace std;
int main()
{
    int x = 0, y = 0;
    cin >> x >> y;
    if (x > y)
        cout << "Max number is: " << x << endl;
    else
        cout << "Max number is: " << y << endl;
    return 0;
}
```



```
123 321
Max number is: 321
```

简单的程序（4）——分支结构

```
#include<iostream>
using namespace std;
int main()
{
    char a = 'A';
    cout << "猜猜我是哪个字母:" << endl;
    cin >> a;
    if (a != 'G')
        cout << "你猜错了！" << endl;
    else if (a == 'G')
        cout << "被你猜中了！" << endl;
    return 0;
}
```

猜猜我是哪个字母:
a
你猜错了！

猜猜我是哪个字母:
G
被你猜中了！

简单的程序 (5) —— 循环结构

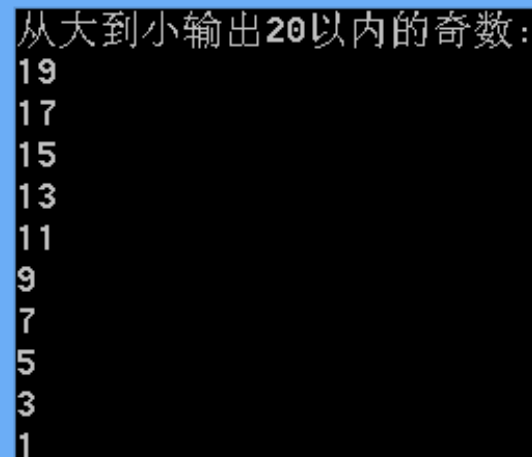
```
#include<iostream>
using namespace std;
int main()
{
    int i = 0;
    cout << "20以内的奇数:" << endl;
    for (i = 0; i < 20; i++)
    {
        if (i % 2 != 0)
            cout << i << endl;
    }
    return 0;
}
```

20以内的奇数:

1
3
5
7
9
11
13
15
17
19

程序 (6) —— 循环结构

```
#include<iostream>
using namespace std;
int main()
{
    int i = 0;
    cout << "从大到小输出20以内的奇数:" << endl;
    for (i = 20; i >= 0; i--)
    {
        if (i % 2 != 0)
            cout << i << endl;
    }
    return 0;
}
```



从大到小输出20以内的奇数:

19
17
15
13
11
9
7
5
3
1

简单的程序 (7) —— 数组

```
#include<iostream>
using namespace std;
int main()
{
    int i = 0;

    char a[10] = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j' };

    cout << “字母表中序号为奇数的前五个字母:” << endl;
    for (i = 0; i < 10; i=i+2)
    {
        cout << a[i] << endl;
    }
    return 0;
}
```

输出字母表中序数为奇数的五个字母:

a
c
e
g
i



```
char a[10] = { 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j' };
```

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|

| | | | | | | | | | |
|------|------|------|------|------|------|------|------|------|------|
| a[0] | a[1] | a[2] | a[3] | a[4] | a[5] | a[6] | a[7] | a[8] | a[9] |
|------|------|------|------|------|------|------|------|------|------|

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|----|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|----|

```
int a[10] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
```



程序 (8) —— 综合

```
#include<iostream>
using namespace std;
int main()
{
    char a = ' '; //用于存放用户输入的字母
    cout << "猜猜我是哪个字母,最多猜5次哦:" << endl;
    int i = 0; //用于记录猜过多少次了
    for (i = 0; i < 5; i++)
    {
        cin >> a;
        if (a == 'G') //如果猜中
        {
            cout << "被你猜中了! " << endl;
            break; //终止循环
        }
        else //如果没有猜中
            cout << "你猜错了! 接着猜吧! " << endl;
    }
    return 0;
}
```

什么样的程序是“好程序”

我们重视：

- ✓ “我的程序运行结果正确，解决了问题！”
- ✓ “我的程序更容易被别人看懂！”
- ✓ “我的程序结构更清楚，更美！”

我们不重视：

- ✗ “我少使用了几个变量！”
- ✗ “我的程序行数比较少！”
- ✗ “我的程序运行起来快了一些！”



荀子（约公元前313 - 前238）

不积跬步
无以至千里
不积小流
无以成江海

——《劝学》