1. **我的第一个C++程序**

流插入运算符<<和cout结合在一起使用，可向显示器屏幕输出数据。

格式：**cout<<表达式;**

功能：它把表达式的值输出到屏幕上，该表达式可以是各种基本类型的常量、变量或者由它们组成的表达式。输出时，程序根据表达式的类型和数值大小，采用不同的默认格式输出，大多数情况下可满足要求。若要输出多个数据，可以连续使用流插入运算符。

流读取运算符>>和cin结合在一起使用，可从键盘输入数据。

**格式：**

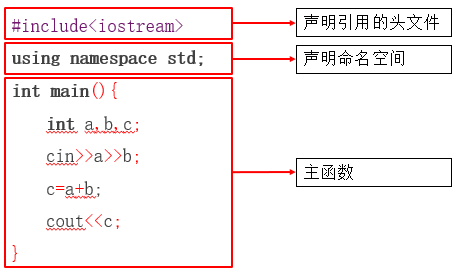
**cin>>变量1>>变量2>>变量3·······;**

功能：是从键盘读取一个数据并将其赋给“变量”。

例子：声明一个整型变量age然后等待用户从键盘输入到cin并将输入值存储在这个变量中。

int age;

cin >> age;



* 第一节课中，同学们完成了上面这个程序，现在请根据老师的问题，尝试修改程序，思考以下问题：
* 1、左边程序实现了两个数的加法，能否求两个数的**减法、乘法、除法**？
* 2、在测试程序时，你输入的最大的数是多大？输入的数据能够无限大？
* **3、在进行两个数的运算时，你是否只测试了整数，如果输入小数程序能正常工作吗，计算结果有什么差异？**

**完善程序：输入一个长方形的两条边，求出长方形的面积。**

#include<iostream>

**using namespace std;**

**int main**(){

**int** a,b,s; //a、b分别代表长方形边长，s代表面积。

cin>>a>>\_\_\_; //输入两条边长

s=\_\_\_\_\_\_;

cout<<s<<endl; //输出面积

}

**数据类型**

数据类型是程序的基础，它告诉我们数据的意义以及我们能在数据上执行的操作。·C++语言支持广泛的数据类型。它定义了几种基本内置类型（如整型、浮点数、字符等）。例如，我们用int 来声明整形，用float来声明浮点型……

整型：

类型名：int • 范围：[-2147483648 ~ 2147483647 ]之间的整数，包括两个端点。 • 整型是有序类型。整型占内存的4个字节。我们在计算整数时，需选取整形，如我们要统计一个班的人数、桌子数量等时，应该选取整型。

实型：

类型名：float,double 功能：说明了一个实数集合，也称浮点数 表示方法： float占4个字节，[1.17549e-038--3.40282e+038]。我们在计算一些包含小数的数据时，我们应该选择用实型来存储，如我们每个人的身高、企业的营业额等数据时。

字符：

类型名：char 功能：说明了一个字符数据集合 表示方法：字符类型中的一个元素表示就是用单 引号括起来的一个且仅一个字符。 ‘1’ 、 ‘A’ 、 ‘b’ 字符类型占内存一个字节。

布尔类型：只有两个值true和false 。 例如，5>3 表达式的值为真（true），2<1 表达式的值为假（false）。

**变量**

变量提供一个具名的、可供程序操作的存储空间，C++中的每个变量都有其数据类型，数据类型决定着变量所占的内存空间大小、存储值的范围以及变量能参与的运算。

**变量定义的基本形式：类型说明符 变量名列表;**

举例：int sum; float price,cost,pay; int a=5,b=10,c=1; //定义变量的同时赋值

#include<iostream>

using namespace std;

int main(){

int a,b,c; //定义3个整型变量a、b、c。

cin>>a>>b;

c=a+b;

cout<<c;

}

**变量命名规则：**

1.以大写字母、小写字母或下划线(\_)开始。

2.可以由以大写字母、小写字母、下划线(\_)或数字0~9组成。

3.大写字母和小写字母代表不同的标识符。

**下列用户自定义的标识符中哪些是非法的？为什么？**

Xyz Name1 4ab \_\_abc123 abc R

AB\_S b(3) m+n \_123 \_a one 333

**赋值**

在C++语言中，“=”赋值运算符，而不表示“等于”。赋值语句是由赋值表达式再加上分号构成的表达式语句。赋值语句的一般形式： **变量=表达式;**

注意，赋值运算符的左侧运算对象必须是一个变量。以下赋值语句是非法的。1024=k;//1024是常亮 i+j=k;//等号左边不能是表达式

#include<iostream>

using namespace std;

int main(){

**int** a,b,c;

cin>>a>>b;

c=a+b; //赋值，将a和b 的值赋值给c。

cout<<c;

}