1-2“Hello World”——程序的基本构成

C/C++程序基本框架

头文件、命名空间、主函数

在第一节课中我们已经编写了Hello World程序：

认识框架结构

```
#include<bits/stdc++.h>
using namespace std;
int main(){
  cout<<"Hello World!";
  return 0;
}
```

C/C++程序基本框架中各句代码的功能可以用**后厨做饭来比喻**，包括厨师、材料以及做菜的步骤：

1. **#include<bits/stdc++.h>**【引入头文件/外部库——做菜用的食材。include包含，bits（二进制）位，std标准standard的缩写，h头head的缩写】
   a. #include表示引入后面<>里面的内容，被称为头文件/库，里有我们编程要用的资源**（做菜用的食材）**。
   b. bits/stdc++.h被称为“万能头文件”，stdc++.h是C++标准库，包含了我们编程所需要的绝大多数资源。bits是存放C/C++标准库的子目录。

2. **using namespace std;**【使用标准命名空间——**做菜的帮手团队**。using使用use的ing形式，name名字，space空间，std标准standard的缩写】
   a. using namespace表示使用某个命名空间，命名空间里面有帮助我们处理资源的帮手
   b. std是标准standard的缩写，是命名空间的名字。表示我们使用的是标准命名空间（**标准帮手团队,主厨掌勺，小工削土豆...**），如cout就是其中一个帮手，用于在黑框框上进行输出。
3. **int main(){.....}**【主函数——**做菜的步骤**。int整数integer的缩写，main主要的，return返回】
   a. 主函数中存放程序的主体内容，代表了这个程序要做些什么（如cout<<"Hello Word!";）。
   b. 每一句用**分号结尾**的话就是一个操作，称为“一句C/C++代码”，这些代码用花括号括起来，{}中的代码要统一缩进一个Tab。
   c. return 0 表示程序给计算机一个0，代表着程序顺利完成，当不返回0（如返回1）时，表示程序没有顺利执行。
   d. cout是输出（c+out），<<后面是要输出的内容。

C/C++程序注释

注释是在编程代码中添加的文本，程序会忽略掉注释的内容。注释主要有下面两个功能：

1. 解释或说明代码的目的、功能或特定部分，帮助自己记录思路
2. 将一部分程序给“屏蔽”而不执行

C/C++程序注释分为以下两种：

1. 单行注释 // 斜杠后面的一行都会被忽略
2. 多行注释 /* 从这里开始的很多行都是注释，
   内容会被忽略直到结尾 */

下面是带单行注释的Hello Word程序：

演示单行注释

```
#include<bits/stdc++.h>//导入万能头文件
using namespace std;//使用标准命名空间
int main(){//主函数
    cout<<"Hello World!";//输出Hello Word!
    return 0;//返回0，表示程序结束
}
```

C/C++程序设计的基本原则1、2

Hello Word程序程序反映了C/C++程序设计的两个基本原则：

1. 逐行执行思想

C/C+程序是逐行执行下去的，例如Hello Word程序，其执行就是按照“导入万能头文件->使用标准命名空间->执行主函数（输出Hello Word->返回0）”这样执行下去的。随着后面代码的增加，这条原则也依然适用（后面要学习的分支和循环只不过是对这条原则的补充）。因此在我们阅读代码时，也应该逐行阅读。

2. 基本基操思想

每一句C++代码，就是一个基本操作，可以分为“操作内容”和“操作对象”。如cout<<"Hello Word!";中操作内容是cout，操作对象是"Hello Word!"。return 0中操作内容是return，操作对象是0。

C/C+基本操作对象——常量与变量

（一）常量

常量是不变的量，指程序中使用的一些具体的数、字符等。常见的常量主要包括：

1. 数字常量：
   a. 【整数常量】1,2,3,4,5,6……
   b. 【小数常量】3.14, 5112.6534，5.0……（注意5和5.0是不同的，**5是整数常量，5.0是小数常量**）
2. 字符常量：'a’,'A’,'+’ ……【使用单引号引起】
3. 字符串常量："Hello Word!"【使用双引号引起】

（二）变量

变量是变化的量，程序用来存储数据的地方，其中的数据是可以改变的。变量就像一个箱子，不同种类的东西需要不同大小的箱子（如小笼子装鸟，中笼子装兔子，大笼子装大象），不同种类的数据，需要不同的变量类型。变量有三要素：类型（什么样的笼子）、名字（笼子上的号牌）、存的值（笼子里的动物）。

1. 常见的变量类型
    a. 整数类型：int类型。存放一个整数。3,4,5
    b. 小数类型：float类型。存放一个小数/实数/浮点数（在编程中，这三种说法等价）。3.14，5.0
    c. 字符类型：char类型。存放一个字符。'a'，'b'，'c'
    d. 字符串类型：string类型。存放一个字符串。"hello"
1. 变量的简单使用：
    a. 变量的定义与初始化
    b. 变量赋值
    c. 变量输出

演示变量的简单使用

```
#include<bits/stdc++.h>//导入万能头文件
using namespace std;//使用标准命名空间
int main(){//主函数
      //（1）定义变量：把盒子变出来，<变量类型 变量名字1,变量名字2，变量名字3……;>
      int a;
      float f1,f2,f3;
      char c1,c2,c3;
      //（2）变量初始化：把盒子变出来的同时，放入东西。使用赋值运算符=
      int b=4,c=5;
      //（3）变量赋值：向已经变出来的盒子中放东西。使用赋值运算符=
      a=3;
      //（4）输出变量：把盒子里的东西（变量的值），展示在黑窗口上。使用cout<<
      cout<<a;
      cout<<b<<c;
      return 0;//返回0，表示程序结束
}
```

1. 变量的命名规则

命名规则	错误例子/不合适例子
只能由英文字母、数字和下划线 _ 组成，英文字母区分大小写	int ab^;
不能以数字开头	int 3a;
不能和其他“关键字”重复。——关键字（又称保留字，是C++有特殊用途的名字）有很多， 比如int、char等	int char;
1. 最好使用驼峰命名法，即首单词的首字母小写，后续不同单词间的首字母大写 2. 例如 dogName。或者使用下划线分割单词dog_name	int dogname

C/C+基本操作内容——算术运算、输入与输出、赋值

（一）算术运算

在编程中，运算通常分为**算术运算、关系运算、逻辑运算**。目前我们重点关注算术运算。不论哪种运算，研究对象都是由运算对象和运算符构成的运算式（操作内容和操作对象），这样的式子在C/C++被称为表达式，每个表达式的本质上都是一个常数。运算对象既可以是常量，也可以是变量。变量用名字表示，参与运算的是此时此刻存的值（常量）。

1. 常见的算数运算符

算数运算符	名称	语法格式	例子	对应的常数
+	加	运算对象1+运算对象2	3+1 a+3	4 变量a存的值+3（a=2，则对应5）
-	减	运算对象1-运算对象2	3-1 a-3	2 变量a存的值-3（a=2，则对应-1）
*	乘	运算对象1*运算对象2	3*1 a*3	3 变量a存的值×3（a=2，则对应6）
/	除	运算对象1/运算对象2	4/2 a/3	2 变量a存的值÷3（a=6，则对应2）
%	取余/取模	运算对象1%运算对象2	3%2 a%3	1 变量a存的值÷3得到的余数（a=7，则对应1）

2. **乘号在C/C++中使用*表示，且不能省略**
3. 除号使用过程中如果除不尽，则直接丢到小数部分（如整数间的运算int a=5/2; 则a=2）【原因后面会解释】
4. 表达式书写过程中，可以使用和数学中同样的括号()，都可以来规范表达式之间的运算顺序
5. 如何理解“表达式本质上是一个常数”
    a. 表达式本质上就是它的计算结果——一个常数
    b. 可以像操作普通常数一样操作一个表达式整体（赋值、输出……）。

理解表达式

```
#include<bits/stdc++.h> // 导入万能头文件
using namespace std; // 使用标准命名空间
int main() {
    // 变量的定义与初始化
    int a = 3, b = 4, c = 5;

    // 计算表达式
    int d = c * c - (a * a + b * b);

    // 两种等价的输出
    cout << d;
    cout << c * c - (a * a + b * b);
```

```
13
14        // 返回0，表示程序结束
15        return 0;
16    }
```

（二）赋值

1. 赋值的语法是 **数据类型 变量名=常量/变量;**
2. 赋值的本质就是"**搬运**（先算出右边的值，再复制到左边的过程）";

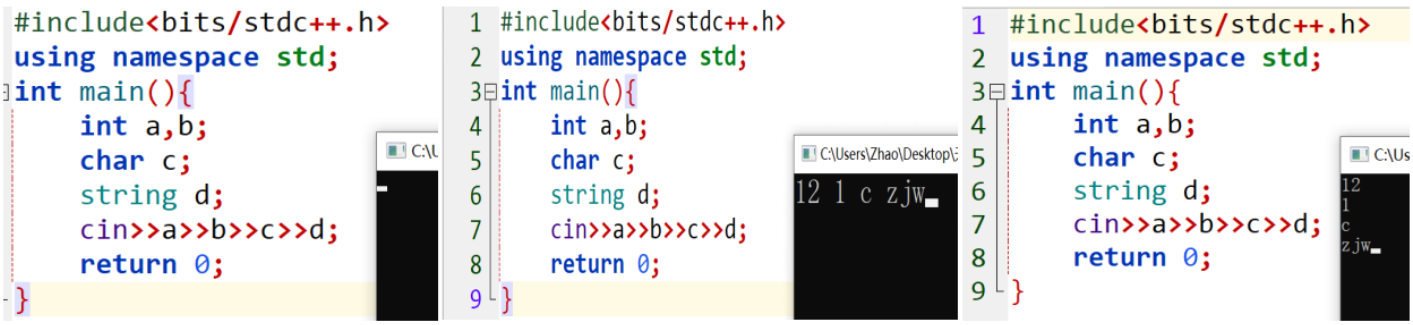
演示赋值过程

```
1   #include<bits/stdc++.h> // 导入万能头文件
2   using namespace std; // 使用标准命名空间
3
4   int main() {
5       // 变量的定义与初始化
6       int a, b = 4; // 变量 a 和 b 的定义，b 初始化为 4
7
8       // a 把 3 搬运到自己的手上
9       a = 3;
10
11      // b 把自己的东西给 a 搬，a 把 b 的东西内容搬到自己东西本上【b 的东西是不变的！】
12      a = b;
13
14      // 先算出 b * 3 + 5 的值，然后 a 把答案搬到自己东西本上
15      a = b * 3 + 5;
16
17      // 先算出 a 此时此刻存的值 + 1 的值，然后 a 把答案搬到自己东西本上
18      a = a + 1;
19
20      // 返回 0，表示程序结束
21      return 0;
22  }
```

（三）输入与输出

1. 程序的输入输出相当于是使用传送带（>>和<<）传送货物，不要记混了方向，**知其一反推其二。**
2. 输入是将**外界的东西**通过传送带放入箱子>>。
3. 输出是把**箱子里的东西拿出来**放到传送带上送走<<。

1. 输入
    a. 输入的语法是< cin>>变量名1>>变量名2>>变量名3……；>，实际上是用户通过键盘，告诉变量哪些内容要放进到自己的箱子上（赋值）。程序逐行执行到cin处，黑框框会光标闪烁，等待用户敲键盘
    b. 程序根据变量类型、输入顺序，逐个输入向让变量搬写的内容，中间空格/回车隔开，输入完成以回车结束
    c. 下面是先从输入中读入一个int，再读入一个int，再读入一个char，再读入一个string的例子：



2. 输出
    a. 输出的语法是cout<<内容1<<内容2<<内容3……；
    b. 是将计算机的一些内容，展现到黑框上。
    c. 这里的内容包括：变量、常量、endl【换行符】、表达式等（本质上都是常量/值）。
    d. 注意C/C++的输出不会自动添加什么（如空格、换行符），遵循“所写即所得”。
    e. 多个cout语句本身的换行不会导致输出结果的换行。在编程题中，只有输出和测试点完全一致，才能算作通过（**不检查每行后额外的空格**）。

（四）基本操作内容的练习

1. 完成题库练习：...

四、C/C++中的非数值数据

在计算机中，一切最终都会变为数（0和1），int、float都是数，变成0和1很简单（称为进制转换），其他的数据类型同样如此。

（一）字符类型char与ASCII码

在计算机中，每个字符都有一个对应的整数（int），我们称为字符的ASCII码（American Standard Code for Information Interchange，美国信息交换标准码）。**当我们给计算机一个字符时，它实际上看到的是一个int，然后再把它转化成0和1组成的序列。**下，感兴趣的同学自行查阅完整ASCII码表，表示了128个（0-127）字符。

1. 目前我们只需关注的核心部分内容

48	0	64	@	80	P	96	`	112	p
49	1	65	A	81	Q	97	a	113	q
50	2	66	B	82	R	98	b	114	r
51	3	67	C	83	S	99	c	115	s
52	4	68	D	84	T	100	d	116	t
53	5	69	E	85	U	101	e	117	u
54	6	70	F	86	V	102	f	118	v
55	7	71	G	87	W	103	g	119	w
56	8	72	H	88	X	104	h	120	x
57	9	73	I	89	Y	105	i	121	y
58	:	74	J	90	Z	106	j	122	z
59	;	75	K	91	[	107	k	123	{
60	<	76	L	92	\	108	l	124	|
61	=	77	M	93	]	109	m	125	}
62	>	78	N	94	^	110	n	126	~
63	?	79	O	95	_	111	o	127	⌂

1. 这里的0-9是字符'0'-'9'，和数字0-9不同
    a. 在黑框框输出的地方没区别，但存储存在差异
    b. '0'-'9'存入**char**，实际上是存储的是48-57。
    c. 0-9存入**int**，实际上就是存数字0-9
2. 连续的数字、大小写字母，ASCII码也是连续的
    a. 数字字符**'0'-'9'是48-57**，大写字母**'A'-'Z'是65-90**，小写字母**'a'-'z'是97-122**
    b. 大小写字母之间相差的是32，不是26（因为中间还有一段符号）
3. ASCII码是美国人设计用来表示英文的，不涉及中文所以char中不能存中文
4. 字符串string可以保存任意内容，可以存储中文

（二）布尔类型bool与逻辑真值

在C/C++中，还有一种特殊的数据类型，被称为布尔类型，用于存储逻辑真值。

1. 逻辑真值
    a. 当我们说一句判断句时，这句话总是要么是对的要么是错的。
    b. 如："今天是周一"、"3是小于4的"、"5是大于7小于10的"。这种判断句的对错在计算机中同样是一个需要处理的信息，也应该可以被表示为0和1。
    c. 通常用1表示“对的/成立的/真的”，写作true或者T；用0表示“错的/不成立的/假的”，写作false或者F
    d. **C++中逻辑真值是布尔类型的（bool）**，通常用于后续的关系与逻辑运算、程序结构判断。

2. 布尔类型

   a. 布尔类型的常量：true、false【和字符类似，true、false本质上也是数字（输出时也用数字来表示），true是1、false是0】

   b. **布尔类型的变量：bool flag1=true，bool flag2=false；**

（三）C/C++的类型转换

1. 当去外国旅游时，我们得把人民币兑换成外币，这样才能在外国消费。在C/C++中， 不同类型的数据可以进行混合运算、赋值。但类似兑换外币，我们需要统一数据类型，才可以进行统一处理。
2. C/C++的类型转换规则：需要注意的是，其中整数转成小数后，.0000000输出一般不显示，但是是存在的。

3. 什么时候会发生类型转换？

   a. 隐式的/自动的：在赋值、算数计算的过程中自动发生

      i. 赋值：赋值符号右边，变为赋值符号左边的类型，再赋值

   ——int a=3.14；　　float a=3；

4. 算术运算：在运算顺序不变的前提下，自左向右扫描各个小项，小项运算结果的类型是运算对象中“更厉害/精度更高”的变量类型

   ——bool<char<int<float

5. 显式的/强制的：通常在输出中进行，用(目标类型)（待转表达式）的语法进行

   ——cout<<(int)3.14<<” ”<<(float)(5/2)<<” ”<<(int)(1.0*5/2);

下面是类型转换的实例：

```cpp
#include<bits/stdc++.h> // 导入万能头文件
using namespace std; // 使用标准命名空间

int main() {
    // 变量的定义与初始化
    int a = 3.14; // a 是 3，发生隐式类型转换
    float f1 = 3; // f1 是 3.0000000，发生隐式类型转换，但输出时不显示 .0000000
    // 计算并赋值
    int b = 3.14 * a + 2; // a 是整数，3.14 是小数，3.14 * a 是 9.42 是小数，a 发生隐式类型转换
                          // 9.42 是小数，2 是整数，9.42 + 2 是小数 11.42，2 发生隐式类型转换
                          // b 是整数，11.42 是小数，赋值后 b 是 11，11.42 发生隐式类型转换
    cout << (int)3.14 << endl; // 强制类型转换
    cout << (int)(3.14 * a + 2) << endl; // 强制类型转换，需要转换的为表达式
```

```
    // 【如何让 a/2 正确变成小数】
    // 【错误方法】
    cout << (float)(a / 2) << endl; // a 是整数，2 是整数，a / 2 是 1 整数，1.5 发生隐式类型转换
                                    // 1 强制类型转换 float 为 1.0000000，但输出时不显示 .0000000
    float f2 = a / 2;
    cout << f2 << endl; // a 是整数，2 是整数，a / 2 是 1 整数，1.5 发生隐式类型转换
                        // 1 是整数，f2 是小数，赋值后 f2 是 1.0000000，但输出时不显示 .0000000
    // 【正确方法】
    cout << 1.0 * a / 2 << endl; // a 是整数，1.0 是小数，1.0 * a 是 3.0 是小数，a 发生隐式类型转换
                                 // 3.0 是小数，2 是整数，3.0 / 2 是 1.5 是小数，2 发生隐式类型转换
    float f3 = 1.0 * a / 2;
    cout << f3 << endl; // a 是整数，1.0 是小数，1.0 * a 是 3.0 是小数，a 发生隐式类型转换
                        // 3.0 是小数，2 是整数，3.0 / 2 是 1.5 是小数，2 发生隐式类型转换
                        // 1.5 是小数，f3 是小数，f3 是 1.5，无类型转换
    // 【如何获取数学除法中的商和余数】
    int n1 = 3, n2 = 2;
    int n3 = n1 / n2; // 获取商
    int n4 = n1 % n2; // 获取余数

    return 0; // 返回 0，表示程序结束
}
```

（四）C/C++程序设计的基本原则3、4

1. 数值本质思想：C/C++程序设计中，一切的一切都应该被当做数值来看，例如：
   a. 变量参与计算时，实际上参与计算的是其存储的常量
   b. 对一个表达式进行计算时（如赋值和输出），表达式整体视为一个常量（计算结果）
   c. 后续字符类型参与计算时，本质上是其ASCII码值（一个整数常量）参与计算
2. 类型统一思想：运算过程中，必须关注参与每一小步的运算对象的类型是否正确。最常见的就是整数除以整数导致小数丢失，需要通过类型转换/乘1*0保证小数部分不被丢失。

（五）类型转换的练习

1. 完成题库练习...