《计算概论A》课程 程序设计部分

C++语言基本成分 —— 数据成分

李　戈

北京大学 信息科学技术学院 软件研究所

lige@sei.pku.edu.cn

北京大学

程序设计语言的构成

- 语言的种类千差万别，但是，一般说来，基本成分不外四种：
  - 数据成分，用以描述程序中所涉及的数据；
  - 运算成分，用以描述程序中所包含的运算；
  - 控制成分，用以表达程序中的控制构造；
  - 传输成分，用以表达程序中数据的传输；

——计算机科学技术百科全书

简单程序的组成

```
//**********************
//*   example.cpp    *
//**********************
#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; //函数结束返回
}
```

注释

预编译：文件包含命令

声明名字空间

函数名

变量定义

数据成分

控制语句

控制成分

运算表达式

赋值语句

运算成分

函数体

输入输出成分

C++程序设计的基本构成

—— 数据成分（基本数据成分）

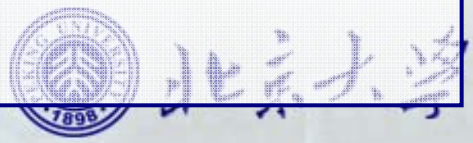
问　题

- 尝试回答一下：
  - 什么是内存？内存里有什么？
  - 内存的作用是什么？
  - 说说以下几个概念之间的关系：

内存、存储单元、字节

再想一下这个程序的执行过程

```
#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;
}
```



变量的定义

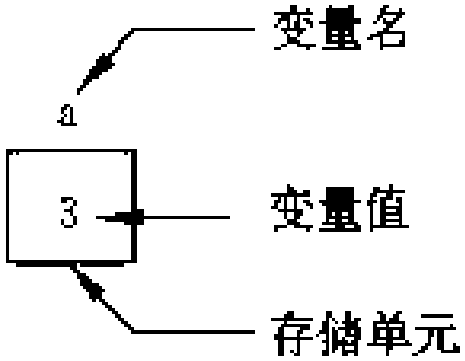
- 变量必须先定义，再使用
- 变量的定义格式

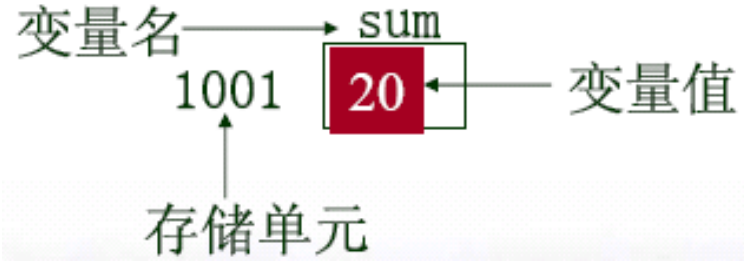
（变量类型）（变量标识符）；

```
int       Max;
int       Max = 0;
char      character;
char      character = 'A' ;
double    Result  = 12.345 ;
```

变量的存储

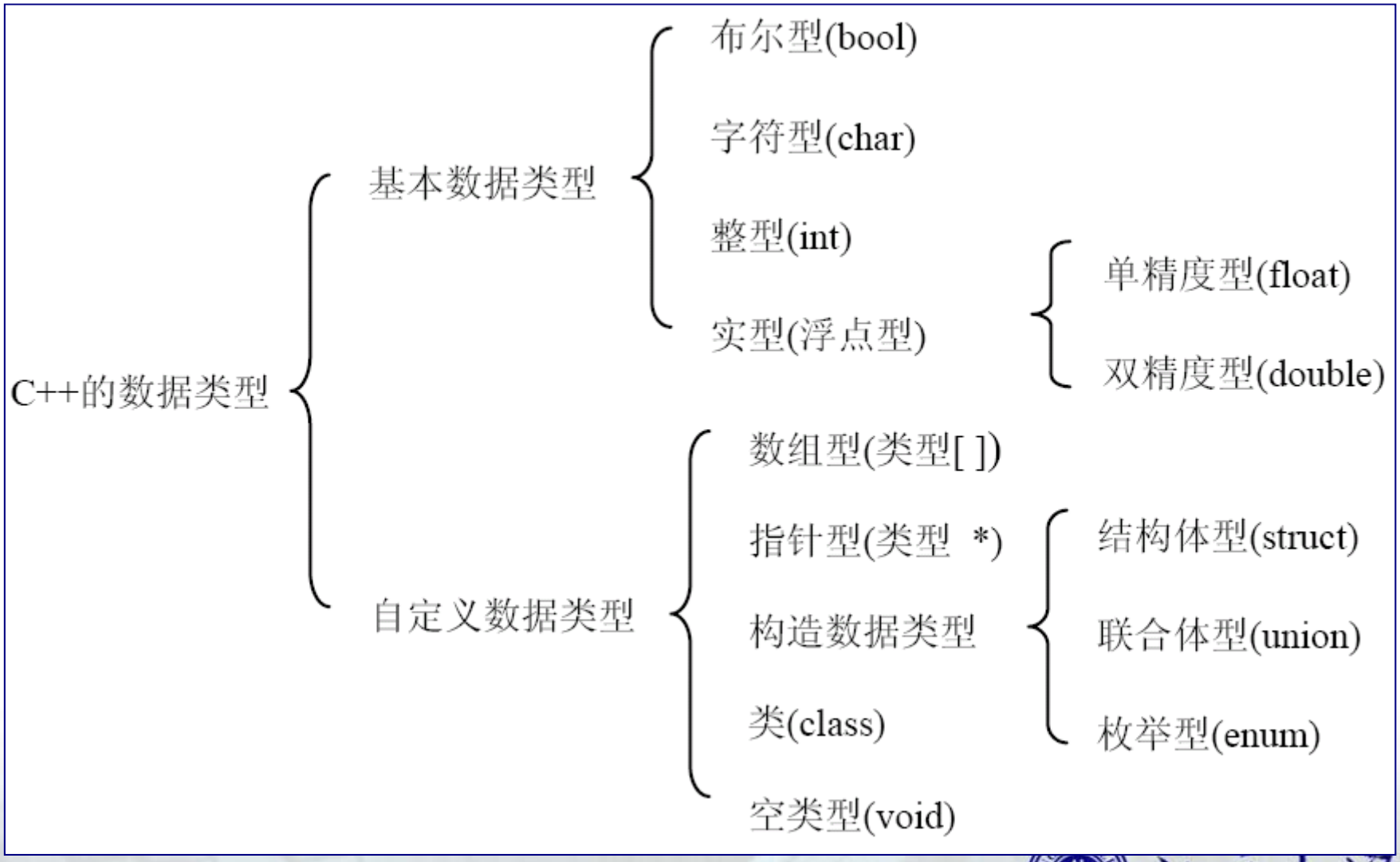
- 变量
  - 每个变量在内存中占用一定的存储单元
  - 变量的名字对应于存储单元的地址
  - 变量的类型对应于存储单元的大小





- 变量的定义
  - 类型说明符 变量名标识符，变量名标识符，...；

计算机能够哪种类型的数据?



整型数据的分类

- 按照数据的表示范围分类
  - 基本型：以int表示
  - 短整型：以short int或short表示
  - 长整型：以long int 或long 表示
- 按照数据有无符号分类
  - 有符号型：表示某个范围内的整数
    - [signed] int； [signed] short； [signed] long
  - 无符号型：表示某个范围内的正整数
    - unsigned int； unsigned short； unsigned long

整型数据的范围

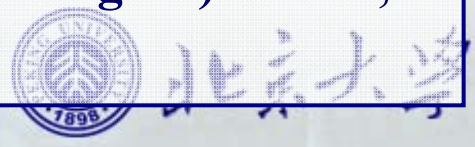
- C/C++标准没有具体规定以上各类数据所占内存字节数，只要求long型数据长度不短于int型，short型不长于int型。
- VC中每种类型所占内存空间和表示的范围：

short [int]	2	−32 768～32 767
signed short [int]	2	−32 768～32 767
unsigned short [int]	2	0～65 535
int	4	−2 147 483 648～2 147 483 647
signed int	4	−2 147 483 648～2 147 483 647
unsigned int	4	0～4 294 967 295
long [int]	4	−2 147 483 648～2 147 483 647
signed long [int]	4	−2 147 483 648～2 147 483 647
unsigned long [int]	4	0～4 294 967 295

如何知道某种类型的数占多少字节?

- sizeof 运算符
  - 用于计算某种类型的对象在内存中所占的字节数。

```
#include <iostream>
using namespace std;
void main()
{
    cout<<"sizeof(short int)="<<sizeof(short int)<<endl;
    cout<<"sizeof(signed short int)="<<sizeof(signed short int)<<endl;
    cout<<"sizeof(unsigned short int)="<<sizeof(unsigned short int)<<endl;
    cout<<"sizeof(int)="<<sizeof(int)<<endl;
    cout<<"sizeof(signed int)="<<sizeof(signed int)<<endl;
    cout<<"sizeof(unsigned int)="<<sizeof(unsigned int)<<endl;
    cout<<"sizeof(long int)="<<sizeof(long int)<<endl;
    cout<<"sizeof(signed long int)="<<sizeof(signed long int)<<endl;
    cout<<"sizeof(unsigned long int)="<<sizeof(unsigned long int)<<endl;
}
```



以不同的方式输出整数

```
#include <iostream>
using namespace std;
int main()
{
    int x = 010,  y = 10,  z = 0x10;
    cout<<"以十进制形式显示：";
    cout<<"x="<<x<<", y="<<y<<", z="<<z<<endl;
    cout<<"以八进制形式显示：";
    cout<<oct<<"x="<<x<<",y="<<y<<",z="<<z<<endl;
    cout<<"以十六进制形式显示：";
    cout<<hex<<"x="<<x<<",y="<<y<<",z="<<z<<endl;
    return 0;
}
```

整型数据的初始值

- 在VC中，定义一个int类型的变量，而不给它赋值，会出现什么情况?
- 例：

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

当输出是个很大的负数时，检查一下是否忘记赋值。

整型数据的溢出

- 在VC中一个int型变量的最大允许值为2147483647，如果再加1，会出现什么情况?
- 例：

```
int main()
{
   int a，b;
   a=2147483647;
   b=a+1;
   cout<<a<<“ ”<<b;
   return 0;
}
```

a：

0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

b：

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

整型数据的存储

- 存储空间的分配

  有符号整数变量：

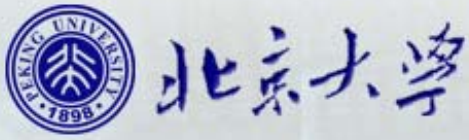
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

  无符号整数变量：

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 整数补码的定义

$$[X]_{补} = \begin{cases} X & \text{if} \quad 0 \le X < 2^n \\ 2^{n+1} + X & \text{if} \quad -2^n \le X < 0 \end{cases}$$



```
#include <iostream>
using namespace std;
int main()
{
    int x = 0x7fffffff;
    cout<<"以十六进制形式显示：";
    cout<<hex<<"x="<<x<<endl;
    cout<<"以十进制形式显示：";
    cout<<dec<<"x="<<x<<endl;
    int y = 0xffffffff;
    cout<<"以十六进制形式显示：";
    cout<<hex<<"y="<<y<<endl;
    cout<<"以十进制形式显示：";
    cout<<dec<<"y="<<y<<endl;
    int z = 0x80000000;
    cout<<"以十六进制形式显示：";
    cout<<hex<<"z="<<z<<endl;
    cout<<"以十进制形式显示：";
    cout<<dec<<"z="<<z<<endl;
    return 0;
}
```

实型数据的分类及范围

- 实型数据
  - 单精度型（float型）
  - 双精度型（double型）
  - 长双精度型（long double型）
- 实型数据的范围（VC中）
  - **float:** 浮点型，占4个字节
    - 范围：$-3.4*10^{38}$～$3.4*10^{38}$，有效为7位
  - **double:**双精度，占8个字节
    - 范围：$-1.7*10^{308}$～$1.7*10^{308}$，有效为15位
  - **long double:**长双精度，占8个字节
    - 范围：$-1.7*10^{308}$～$1.7*10^{308}$，有效为15位

实型数据的舍入误差

- 由于实型变量是由有限的存储单元组成的，因此能提供的有效数字总是有限的，在有效位以外的数字将被舍去。
- 例：实型数据的舍入误差

```
#include <iostream>
using namespace std;
int main()
{
    float a, b;
    a = 123456.789e5;
    b = a + 20 ;
    cout<<a<<" "<<b<<endl;
    cin.get();
    return 0;
}
```

- a+20的理论值应是12345678920，而一个实型变量只能保证的有效数字是7位有效数字；
- 运行程序得到的a和b的值是

  1.23457e+010

- 应当避免将一个很大的数和一个很小的数直接相加或相减，否则就会“丢失”小的数。

布尔型

- 布尔类型(bool)是C++新增的类型，只能取true 和false 两个值，分别对应整数的1 和0。

```
void main()
{
    bool b1=true,b2=false;
    cout<<"b1=true 时，b1="<<b1<<endl;
    cout<<"b2=false 时，b2="<<b2<<endl;
    int i;
    cout<<"请输入一个正整数i：";
    cin>>i;
    b1=i>3;
    cout<<"b1=i>3 时，b1="<<b1<<endl;
    b2=(bool)-100;
    cout<<"b2=-100 时，b2="<<b2<<endl;
}
```

字符型数据

- 字符型常量
  - 用单引号括起来的单个字符。如 ‘a’, ‘x’, ‘D’, ‘#’
  - 一些特殊的字符常量，以 \ 开头的字符序列，有特殊的含义, 故叫转义符，如：\n表示换行

```
#include <iostream>
using namespace std;
int main( )
{
   char a = 'a';
   cout<<a<<endl;
   cout<<"This is the first line !  \n";
   cout<<‘a’<<‘\n’;
   return 0;
}
```

C++中的转义字符

转义字符	描　　述
\a	响铃(audible bell)
\b	退格(backspace)
\f	换页(formfeed)
\n	换行(newline)
\r	回车(carriage return)
\t	水平制表(horizontal tab)
\v	垂直制表(vertical tab)
\\	反斜线(backslash)
\'	单引号(single quote)
\"	双引号(double quote)
\DDD	八进制数 DDD 对应的字符(octal number)
\xHH	十六进制数 HH 对应的字符(hexadecimal number)

字符型数据

- 一个字符占一个字节
  - 字符要转化成数字存储；
  - 一个字符变量一般存放一个字符；
  - 字符与字符串是不同的数据类型；
    - 不能用字符变量来存放字符串；
- 字符数据的存储形式
  - 在内存中字符变量的存储形式是ASCII码

ASCII码

- ASCII码(America Standard Code for Information Interchange，美国信息交换标准码)
  - 美国国家标准学会（American National Standards Institute － ANSI） 制定
  - 早期使用7 位表示，只规定了128个最常用的英文字符；
  - 现在则使用8个位，共可表示256个不同的文字与符号；
  - 在此基础上发展了16bits的Unicode

字符型数据

L \ H	0000	0001	0010	0011	0100	0101	0110	0111
0000	NUL	DLE	SP	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	)	8	H	X	h	x
1001	HT	EM	(	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

北京大学

字符型数据

- 由于存储类型和整型相同
  - 字符数据和整型数据可以相互赋值
  - ASCII码可以和整数一样进行运算
  - 字符和整型均可以以两种形式输出

```
int a=32;
char b='a';
 int c = b-a;    //结果为 65
char a=6+256;
                  //溢出！因为a只能占用8位，最大256
                  //输出：♠ 与 a = 6; 相同
```

C++基本数据类型

数据类型标识符	字节数	数值范围
bool	1	false, true
char	1	−128～127
signed char	1	−128～127
unsigned char	1	0～255
short [int]	2	−32 768～32 767
signed short [int]	2	−32 768～32 767
unsigned short [int]	2	0～65 535
int	4	−2 147 483 648～2 147 483 647
signed int	4	−2 147 483 648～2 147 483 647
unsigned int	4	0～4 294 967 295
long [int]	4	−2 147 483 648～2 147 483 647
signed long [int]	4	−2 147 483 648～2 147 483 647
unsigned long [int]	4	0～4 294 967 295
float	4	3.4e−38～3.4e38
double	8	1.7e−308～1.7e308
long double	8	1.7e−308～1.7e308

关于变量

C++ 程序的 标识符

- 什么是标识符
  - 用来标识符号常量名、变量名、函数名、数组名、类型名、文件名的有效字符序列称为标识符(identifier)。
  - C++语言规定标识符只能由字母、数字和下划线三种字符组成，且第一个字符必须为字母或下划线，且不可与保留字（关键字）相同。
    - 合法的标识符：
      sum，average， -total， class， day， month，
      student-name，tan，lotus-1-2-3，basic， li-ling
    - 不合法的标识符：
      M.D.John，￥123，＃33，3D64，a>b

C++ 语言的 保留字

- 在程序里具有语言预先定义好的特殊意义，因此不能用于其他目的，不能作为普通的名字使用。
  - C++保留字：

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

变量的命名

- 匈牙利命名法
  - 由Microsoft的著名开发人员、Excel的主要设计者查尔斯·西蒙尼在他的博士论文中提出来的，由于西蒙尼的国籍是匈牙利，所以这种命名法叫匈牙利命名法。
- C标识符的命名规则：
  1. 标识符的名字以一个或者多个小写字母开头，用这些字母来指定数据类型。
  2. 在标识符内，前缀以后就是一个或者多个第一个字母大写的单词，这些单词清楚地指出了源代码内那个对象的用途。
- 如：chGrade; nLength; bEnable; strStudentName

由相同类型的变量可以组成

——数组

北京大学

普通数组

- 定义其他类型的数组并附初始值
  - **int a[26];** //定义数组必须指明数组大小
  - **float a[26];**
  - **bool a[26];**
  - **char a[26] = {‘a’, ‘b’, ‘c’, …, ‘z’};**
- 数组的使用
  - 要使用第n个数：**a[n-1];** **a[n-1] = 1+2;**
  - 注意：对数组**a[n]**，只能使用**a[0] – a[n-1];**
    a[n]不能用；

从 字符数组 到 字符串

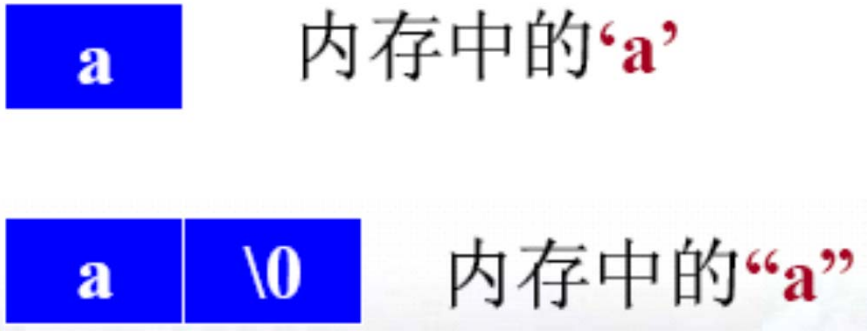
- 字符数组
  - **char c[5]={‘C’, ‘H’, ‘I’, ‘N’, ‘A’};**
- 有一种字符数组
  - **char c[6]={‘C’, ‘H’, ‘I’, ‘N’, ‘A’, ‘\0’};**

  等价于
  - **char c[6]=“CHINA”;**
- 也就是说：
  - **“CHINA” 等价于 {‘C’, ‘H’, ‘I’, ‘N’, ‘A’, ‘\0’}**

北京大学

关于字符串

- 单引号与双引号
  - ‘a’是字符常量，“a”是字符串常量，二者不同



- ‘A’有意义，“A”有意义，‘ABC’无意义！
- C语言中，没有“字符串”类型；

看看这段程序

```
#include <iostream>
using namespace std;
int main( )
{
    int int_char_a = 'ab';
    int int_char_b = 'abcde';
    cout<<dec<<"int_char_a = "<<int_char_a<<endl;
    cout<<hex<<"int_char_a = "<<int_char_a<<endl;
    cout<<dec<<"int_char_b = "<<int_char_b<<endl;
    cout<<hex<<"int_char_b = "<<int_char_b<<endl;
    char char_c = "ab";
    return 0;
}
```

问题

- 可不可以把多个变量“组合”在一起？

```
{
        int     id;
        char    name[20];
        char    sex;
        int      age;
        float   score;
        char    addr[30];
}
```

由不同类型的变量可以组成

——结构体

北京大学

结构体

- 声明一个名为“学生”的结构体

```
struct student \\结构体的名字为“student”;
 {
        int     id;          \\声明学号为int型;
        char    name[20];  \\声明姓名为字符数组;
        char    sex;         \\声明性别为字符型;
        int      age;        \\声明年龄为整型;
        float   score;     \\声明成绩为实型;
        char    addr[30];  \\声明地址为字符数组
 };          \\注意大括号后的“;”
```

定义结构体类型的变量

- 定义结构体变量

  struct student　　student1, student2;

  （结构体类型名）（结构体变量名）；

  - 对比：

    int a;(struct student 相当于 int)

    float a; (struct student 相当于 float)

结构体变量的引用

- 引用结构体变量中成员的方式为

  结构体变量名.成员名

  - 如：**student1.id = 10010;**

    student1.birthday.month = 10;

- 不能将一个结构体变量作为一个整体进行输入和输出
  - 不正确的引用：**cout<<student1;　　cin>>student1;**
- 只能对结构体变量中的各个成员分别进行输入和输出
  - 正确的引用：**cin>>student1.id;　cout<<student1.id;**

结构体示例

```
int main( )
{
        int day = 7;
        struct date Birthday;
        Birthday.day = 25;
        Birthday.month = 12;
        Birthday.year = 0;
        cout<<day<<endl;
        cout<<Birthday.day<<endl;
        return 0;
}
```

```
struct  date
{
   int    month;
   int    day;
   int    year;
};
```

所有“数”都是有类型的

——常量也一样！

常 量

- 常量
  - 在程序运行过程中，其值保持不变的量
- 字面常量
  - －1，0，123，4.6, －1.23；
- 符号常量
  - 用一个标识符代表一个常量的，称为符号常量

```
#include <iostream>
int main( )
{
        const float PI=3.14159f;
        float  r, area;
        cin>>r;
        area = r * r * PI;
        cout<<“area = ”<<area;
        return 0;
}
```

北京大学

常量有类型吗?

- 整型常量的后缀
  - n = 10000L; //长整型常量
  - m = -0x88abL; //长整型十六进制常量
  - k = 10000U; //无符号整型常量
  - i = 07777LU; //无符号长整型八进制常量
- 浮点型常量的后缀
  - x = 3.1415F //单精度浮点型常量
  - y = 3.1415L //长双精度浮点型常量
- 说明:
  - 浮点型常量默认为double 型;
  - U, L, F均可以小写;

好好想想，有没有问题?

谢　谢　！

北京大学