



规格严格 功夫到家



第2章 C数据类型



哈尔滨工业大学

计算机科学与技术学院

苏小红

sxh@hit.edu.cn

本章学习内容

- ✍ 常量和变量，整型和实型
- ✍ 标识符命名，变量的定义和赋值
- ✍ 计算数据类型占用内存空间的大小

2.1 常量与变量

■ C语言程序处理的数据形式

- 常量 (constant)

- 在程序中不能改变其值的量

- 变量 (variable)

- 其值在程序执行过程中是可以改变的

一个简单的C程序例子

```
#include <stdio.h>
```

```
/*函数功能：计算两个整数相加之和  
入口参数：整型数据a和b  
返回值：    整型数a和b之和  
*/
```

```
int Add(int a, int b)  
{  
    return (a + b);  
}
```

```
/*主函数*/
```

```
main()  
{  
    int x, y, sum = 0;  
  
    printf("Input two integers:");  
    scanf("%d%d", &x, &y);  
    sum = Add(x, y);  
    printf("sum = %d\n", sum);  
}
```

编译预处理命
令

程序注释

并列的两个函数
其中主函数是
程序的入口

```
/*输入两个整型数x和y*/  
/*调用函数Add计算x和y相加之和*/  
/*输出x和y相加之和*/
```

C程序常见符号分类

■ 关键字 (Keyword)

- 又称保留字 (C Reserved Word)
 - A word that has special meaning in C

■ 标识符 (Identifier)

- 系统预定义标识符 (C Standard Identifier)
 - A word having special meaning but may be redefined (but is not recommended!!)
- 用户自定义标识符
 - 变量, 函数名,

```
#include <stdio.h>

int Add(int a, int b)
{
    return (a + b);
}

main()
{
    int x, y, sum = 0;

    printf("Input two integers:");
    scanf("%d%d", &x, &y);
    sum = Add(x, y);
    printf("sum = %d\n", sum);
}
```

C程序常见符号分类

■ 运算符 (Operator)

- 详见附录C

■ 分隔符 (Separator)

- 空格、回车/换行、逗号等

■ 其他符号

- {和}标识函数体或语句块
- /*和*/是程序注释的定界符

■ 常量 (Constant)

```
#include <stdio.h>

int Add(int a, int b)
{
    return (a + b);
}

main()
{
    int x, y, sum = 0;

    printf("Input two integers:");
    scanf("%d%d", &x, &y);
    sum = Add(x, y);
    printf("sum = %d\n", sum);
}
```

2.1.1 常量 (Constant)

■ 在程序中不能改变

十进制

长整型

无符号

十六进制

■ 包括：

十

指数形式

单

长双精度实型

— 整型(如 0, 67,

■ 默认为int

— 实型(如 2.3, 1.2e-5, 2.73F, 2.73L)

■ 默认为double

— 字符型(如 'z', '3', '\$', '\n')

■ 用\开头的字符为转义字符, 代表1个字符

— 字符串(如 "UKM", "1", "5a")

— 枚举型

2.1.2 变量 (Variable)

- 变量的值在程序执行过程中是可以改变的
- 变量的属性

- Name 变量名
- Type 变量类型
- Value 变量的值
- Address 变量的存储单元——地址

变量声明(*Variable Declaration*)

■ 变量的声明

类型关键字 变量名;

■ 使用变量的基本原则

- 变量必须先声明，后使用
- 所有变量必须在第一条可执行语句前声明

■ 声明的顺序无关紧要

■ 一条声明语句可声明若干个同类 型的变量

```
1 main()
2 {
3     int a;
4     float b;
5     char c;
6     a = 1;
7     b = 2.5;
8     c = 'A';
9 }
```

```
1 main()
2 {
3     int a;
4     a = 1;
5     float b;
6     b = 2.5;
7     char c;
8     c = 'A';
9 }
```

变量声明(*Variable Declaration*)

- 声明变量是初始化变量的最好时机
 - 不被初始化的变量，其值为随机数

```
1 main()
2 {
3     int a = 1;
4     float b = 2.5;
5     char c = 'A';
6 }
```

```
#include <stdio.h>

main()
{
    int x, y, sum;
    sum = x + y;
    printf("sum=%d\n", sum);
}
```



结果会是什么？

变量赋值(Variable Assignment)

运算规则：

变量 \leftarrow 表达式

语法：

变量 = 表达式；

规则： 左值和右值类型一致

Valid Example:

```
int x;  
x = 12; ✓
```

Invalid Example:

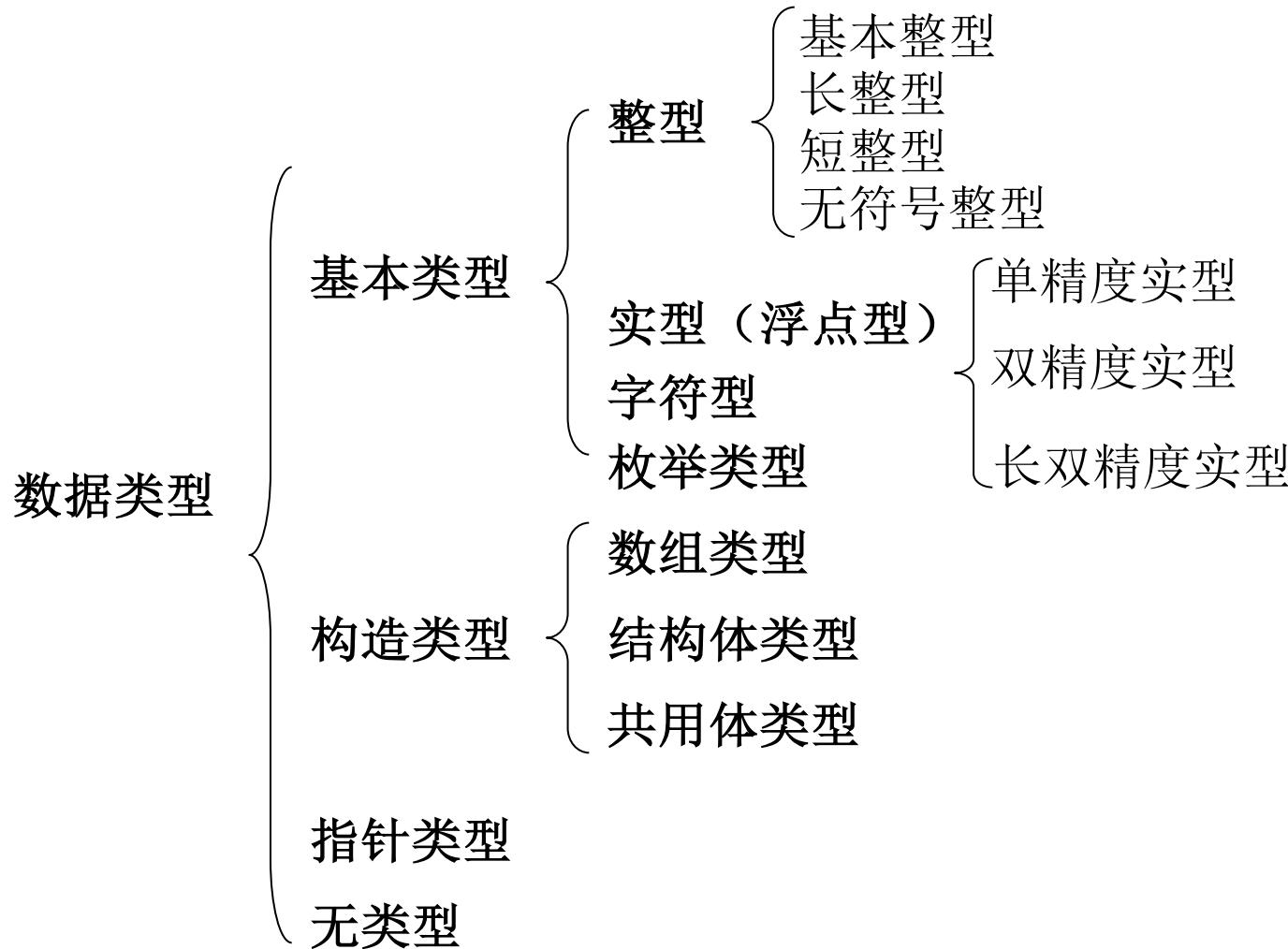
```
int y;  
y = 5.75;
```

2.3 数据类型 (*Data Type*)

- 为什么要区分类型?
- 变量的类型决定了
 - 数据的存储形式
 - 合法的取值范围
 - 占用内存空间的大小
 - 可参与的运算种类



数据类型 (*Data Type*)



基本数据类型

■ **int**

- 整数，在目前绝大多数机器上占4个字节
- TC 2.0，2个字节

■ **float**

- 单精度浮点数，4个字节

■ **double**

- 双精度浮点数，8个字节

■ **char**

- 字符，1个字节
- 表示256个ASCII字符，或0~255的整数

数据类型修饰符

short

- short int, 简写为short, 短整数, 2个字节

long

- long int, 简写为long, 长整数, 4个字节
- long double, 长双精度(高精度)浮点数, 10个字节

unsigned

- 用来修饰char、int、short和long
- 无符号整数(正整数和0)

不同类型取值范围不同

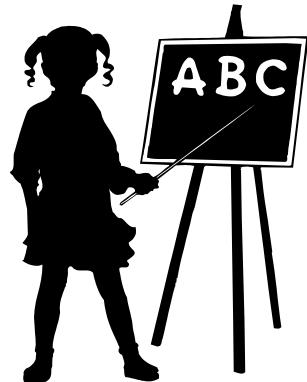
■ C语言直接提供的任何类型都有取值范围

数据类型..	所占字节数 (bytes) ..	取值范围..
char.. signed char..	1..	-128~127..
unsigned char..	1..	0~255..
short int.. signed short int..	2..	-32768~32767..
unsigned short int..	2..	0~65535..
unsigned int..	4..	0~4294967295..
int.. signed int..	4..	-2147483648~2147483647..
unsigned long int..	4..	0~4294967295..
long int .. signed long int ..	4..	-2147483648~2147483647..
float ..	4..	$3.4 \times 10^{-38} \sim 3.4 \times 10^{38}$..
double..	8..	$-1.7 \times 10^{-308} \sim 1.7 \times 10^{308}$..
long double..	10..	$-1.2 \times 10^{-4932} \sim 1.2 \times 10^{4932}$..

不同类型取值范围不同

- Most significant bit (MSB) is sign(最高位为符号位)

Type	Min value	Max value
short int	$-32768 (-2^{15})$	$32767 (2^{15} - 1)$
unsigned short int	0	$65535 (2^{16} - 1)$
unsigned char	0	255



2.4 如何计算变量或数据类型所占内存空间的大小

英文称谓	中文称谓	换算方法
bit(b)	位 (比特)	
Byte(B)	字节	$1 \text{ B} == 8 \text{ b}$
Kilobyte(KB)	K	$1 \text{ KB} == 1,024 \text{ B}$
Megabyte(MB)	兆	$1 \text{ MB} == 1,024 \text{ KB}$
Gigabyte(GB)	G	$1 \text{ GB} == 1,024 \text{ MB}$
Terabyte(TB)	T	$1 \text{ TB} == 1,024 \text{ GB}$

1 TB = 1024 GB

1 GB = 1024 MB

1 MB = 1024 KB

1 KB = 1024 B

1 B = 8 b



2.4 如何计算变量或数据类型所占内存空间的大小

- 一个位有多大?
 - 只能是0或者1，二进制

- 一个字节有多大?
 - 可以表示0~255之间的整数



不同类型占用的内存字节数不同

- 同种类型在不同的平台其占字节数不尽相同
 - 如int在16位、32位和64位系统上分别占2、4和8个字节
- 不要对变量所占的内存空间字节数想当然
 - 用**sizeof**获得变量或者数据类型的长度
- 现象与危害
 - 在平台间移植时会出现问题，导致数据丢失或者溢出



Type	Usual size	Supercomputer
int	32 bits	64 bits
short int	16 bits	32 bits
long int	32 bits	128 bits
float	32 bits	64 bits
double	64 bits	128 bits
char	8 bits	

sizeof到底是什么？

- C语言的关键字，并非函数
 - 计算类型占用的字节数
- 两种语法形式

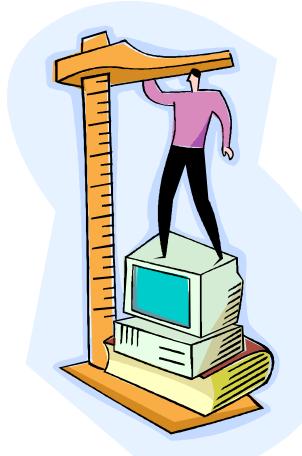
sizeof(类型)

- 结果为类型占用的字节数

sizeof(表达式)

- 结果为表达式值所属类型占用的字节数

一般都使用**sizeof(变量名)**



现场演示例2.2 在TC和VC、CB下的运行结果

```
#include <stdio.h>

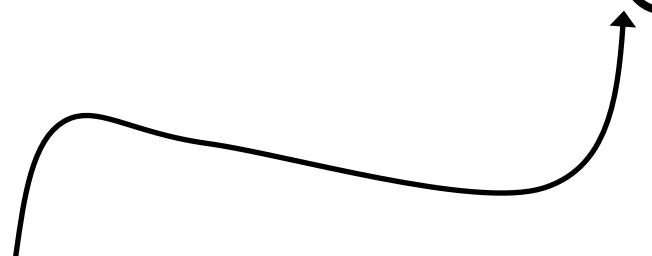
main()
{
    printf("Data type           Number of bytes\n");
    printf("----- -----\n");
    printf("char                 %d\n", sizeof(char));
    printf("int                  %d\n", sizeof(int));
    printf("short int            %d\n", sizeof(short));
    printf("long int             %d\n", sizeof(long));
    printf("float                %d\n", sizeof(float));
    printf("double               %d\n", sizeof(double));
}
```

2.5 变量的赋值和赋值运算符

简单赋值 (Simple Assignment)

语法：

变量 = 表达式 ;



Don't forget the semicolon !!

Every assignment expression has a value

多重赋值 *Multiple Assignment*

语法:

变量1 = 变量2 = 表达式 ;

变量1 = (变量2 = 表达式) ;

Don't forget the semicolon !!

右结合: 从右向左赋值

多重赋值

Multiple Assignment

Example:

```
→int a, b;  
→float x, y;  
.  
.  
.  
→a = b = 0;  
→x = y = 100.0;
```

a	0
b	0
x	100.0
y	100.0



■ Questions and answers

