



西北工业大学

NORTHWESTERN POLYTECHNICAL UNIVERSITY

C++程序设计

Programming in C++



1011018

主讲：李少楠，计算机学院

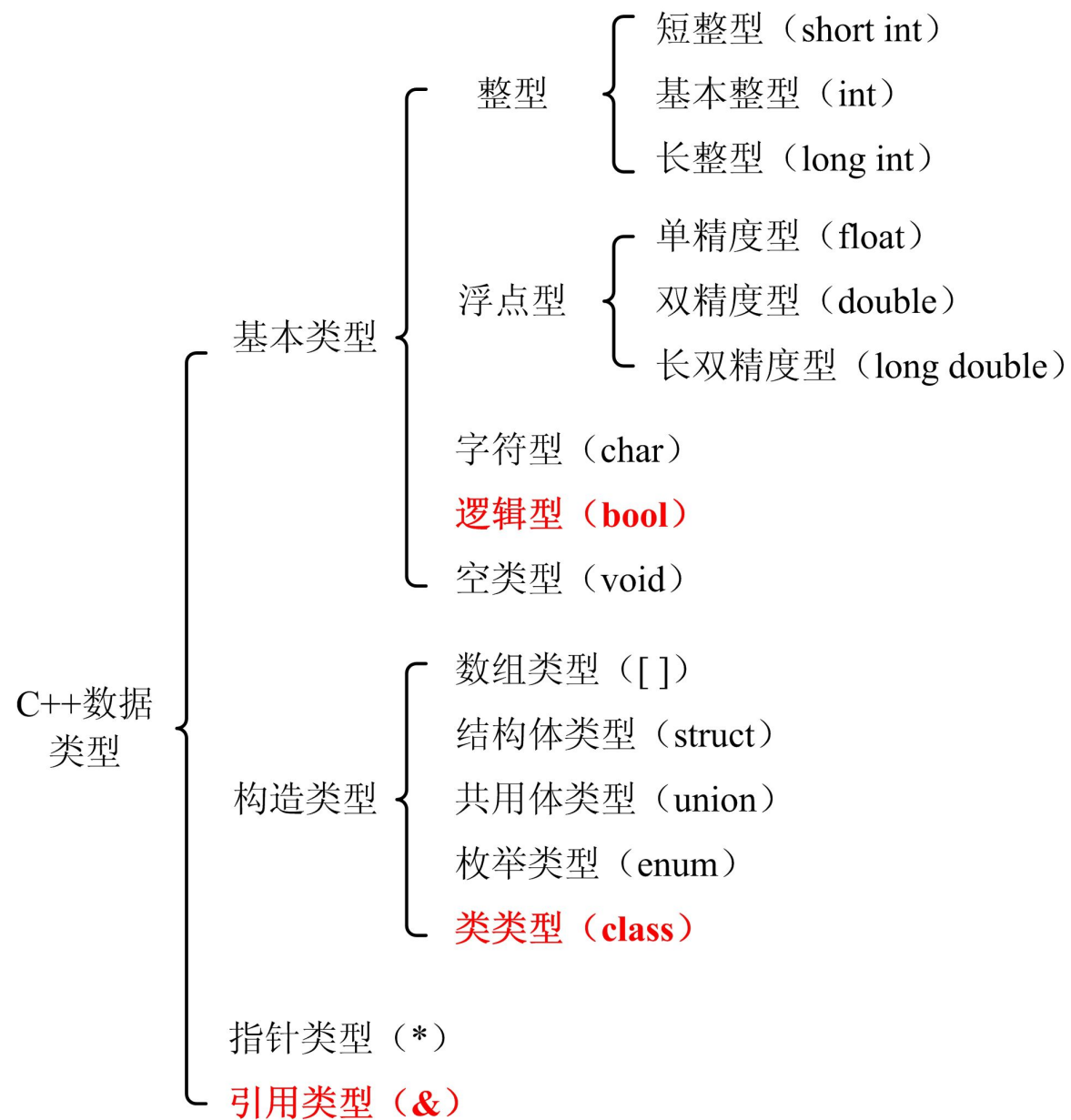
程序中数据的表示

1、数据类型



- ▶ 利用计算机求解问题，首先要做的就是将数据引入到计算机中。
- ▶ 在程序语言中是通过数据类型来描述不同的数据形式的，数据类型不同，求解问题的方法也会不同。

3.1 数据类型



3.1 数据类型

表3-1 基本类型数据的内存长度和数值范围

类型	类型标识符	内存长度	数值范围	精度
整型	[signed] int	4	-2147483648~+2147483647	
无符号整型	unsigned [int]	4	0~4294967295	
短整型	[signed] short [int]	2	-32768~+32767	
无符号短整型	unsigned short [int]	2	0~65535	
长整型	[signed] long [int]	4	-2147483648~+2147483647	
无符号长整型	unsigned long [int]	4	0~4294967295	
字符型	[signed] char	1	-128~+127	
单精度型	float	4	$3.4\times10^{-38}\sim3.4\times10^{38}$	7
双精度型	double	8	$1.7\times10^{-308}\sim1.7\times10^{308}$	16
长双精度型	long double	同上	同上	19
逻辑型	bool	1	0或1	

▶ 1. 整形

- ▶ C++语言整型分为长整型（long int）、基本整型（int）和短整型（short int）
- ▶ long int可以简写为long、short int可以简写为short。
- ▶ $\text{short} \leq \text{int} \leq \text{long}$

3.1 数据类型

- ▶ 不同的数据类型规定了不同的机器数长度，决定了对应数据的数值范围，当一个整数超出此范围时计算机将其转换为在数值范围内所允许的一个数，称为**整型数据的溢出处理**。
- ▶ 一般地，超过最大值的有符号整型数值会向上溢出变成负数，超过最小值的数据会向下溢出变成正数。

图3.1 short型的溢出

0111111111111111		32767		1000000000000000		- 32768
+ 0000000000000001		+ 1		+ 1111111111111111		- 1
1000000000000000		- 32768 (补码)		1 0111111111111111		32767
向上溢出				向下溢出		

▶ 2. 浮点型

- ▶ C++语言浮点型又称实型，分为单精度（float）、双精度（double）和长双精度（long double）3种。
- ▶ 在C++中规定float型在内存中占用4个字节，提供7位有效数字；
- ▶ double型和long double型在内存中占用8个字节，提供16位有效数字。

- ▶ 因为浮点型数据长度是有限的，所以浮点数存在计算误差。
- ▶ 虽然浮点数精度越高计算结果越精确，但其处理时间也长。
- ▶ 一个较大的浮点数与一个很小的浮点数做加法时，由于精度限制使得很小的浮点数被忽略了，从而使得这样的加法无意义。

3.1 数据类型

【例3.1】

```
1 #include <iostream>
2 #include <iomanip>
3 using namespace std;
4 int main()
5 {
6     float a=0.00000678f;
7     double b=0.00000678;
8     a = a + 111111.111f;
9     b = b + 111111.111;
10    cout<<setiosflags(ios::fixed)<<setprecision(16);
11    cout<<"a="<<a<<" ,b="<<b<<endl;
12    return 0;
13 }
```

运行结果：

```
a=111111.1093750000000000,b=111111.1110067800036632
```

▶ 2. 字符型

- ▶ C++语言字符型分为有符号（signed char）和无符号（unsigned char）两种，其中signed书写时可以省略。
- ▶ 字符型数据在内存中占用1个字节，采用二进制形式存储。

3.1 数据类型

- ▶ 字符型数据可以存储单字节字符，如ASCII码，此时在内存中的数据是字符的ASCII码值。例如字符'A'在内存中的存储形式为

0	1	0	0	0	0	0	1
---	---	---	---	---	---	---	---

'A'的ASCII码值

- ▶ 在C++语言中**字符型数据和整型数据之间可以通用**。一个字符数据可以赋给整型变量，一个整型数据也可以赋给字符型变量，还可以对字符型数据进行算术运算。

3.1 数据类型

【例3.2】

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, j;
6     char c1, c2;
7     c1='a'; //字符数据赋值给字符型
8     c2=98; //整数数据赋值给字符型
9     i='A'; //字符数据赋值给整型
10    j=66; //整数数据赋值给整型
11    cout<<"i="<<i<<" , j="<<j<<" , c1="<<c1<<" , c2="<<c2<<endl;
12    cout<<"c1-32="<<c1-32<<endl; //字符型可以进行减法运算
13    return 0;
14 }
```

运行结果：

```
i=65 , j=66 , c1=a , c2=b
c1-32=33
```

CP 程序设计