

# Ch.1 C/C++的特性

—

2017 / 9

# 如何学习编程

- 从一门流行的语言开始
- 学会使用编辑工具和IDE
- 学会调试的方法
- 到应用环境中去使用
- 适应不同开发环境
- 做应用项目



# 开始学一门编程语言

- “Hello World”
- 数据类型
- 基础逻辑（条件、分支、循环）
- 特性和思想（比如OOP）
- 不断增加和改进的特性
- 在项目中使用

# Now, C/C++

- Hello world

# Hello world大有文章

```
1  #include<iostream>
2
3  int main(int argc, char * argv[])
4  {
5      printf("Hello world\n");
6
7      std::cout << "OK, C ++ " << std::endl;
8      using namespace std;
9      cout << "OK, using namespace" << endl;
10
11     cout << argc << endl;
12
13     for (int i = 0; i < argc; i++)
14         cout << argv[i] << endl;
15
16     return 0;
17 }
```



# 命令行程序

在命令行环境下运行的程序

- `main` 程序从这里运行
- `argc` 传入参数的个数
- `argv` 以字符串方式传入的“参数”，参数之间以空格或是Tab分隔
- Enter执行

# namespace

- `std::cout`
- `using namespace std;`
- 自定义namespace及使用

# namespace

```
4  void show()
5  {
6      printf("global show\n");
7  }
8
9  namespace circle
10 {
11     void show()
12     {
13         printf("A Circle\n");
14     }
15 }
16
17 namespace heart
18 {
19     void show()
20     {
21         float y, x, a;
22         for (y = 1.5f; y > -1.5f; y -= 0.1f)
23         {
24             for (x = -1.5f; x < 1.5f; x += 0.05f)
25             {
26                 a = x*x + y*y - 1;
27                 putchar(a*a*a - x*x*y*y*y <= 0.0f ? '*' : ' ');
28             }
29             putchar('\n');
30         }
31     }
32 }
```

```
33 int main()
34 {
35     show();
36     ::show();
37     circle::show();
38     heart::show();
39
40     system("pause");
41     return 0;
42 }
43
44
```



# 多文件项目

- 文件头声明，头文件包含
- 预编译处理
- 宏替换
- 宏函数

# 函数调用

- 函数重载 `overload`
- 函数重写 `override`
- 递归，要有终结条件



# 重载函数

```
int max(int x, int y)
{    return (x > y ? x : y);}
float max( float a, float b )
{    return (a > b ? a : b);}
double max(double m, double n)
{    return (m > n ? m : n);}
```

```
cout<<max( 'a' , 'b' )<<endl;   正确
cout<<max( 'a' , 99)<<endl;      正确
cout<<max(100, 'a' )<<endl;      正确
cout<<max( 'a' , 12.3)<<endl;    错误
cout<<max(12, 12.3)<<endl;       错误
cout<<max(3.4f, 12.3)<<endl;     错误
```

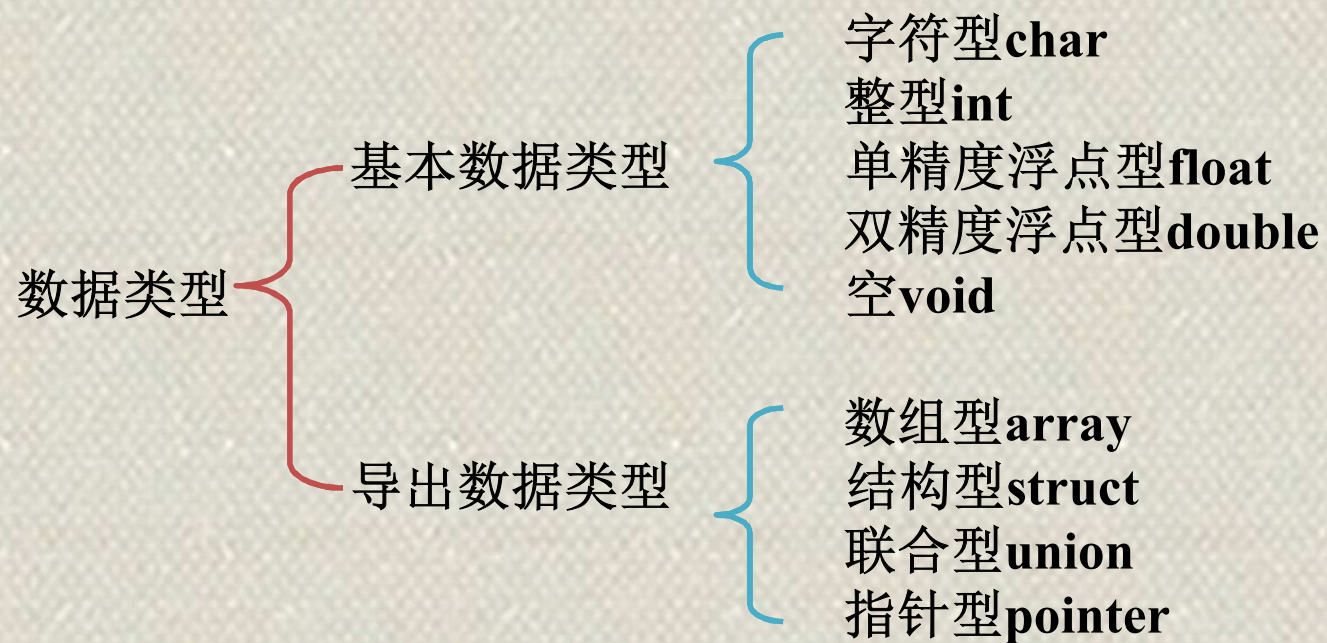


# 递归

```
short fac(short n)
{
    if(n == 0)
        return 1;
    return n * fac(n-1);
}
```

# • 数据类型

类型标识符	类 型 名	字 节 数	数值范围
bool	布尔型	1	true 或 false
char	字符型	1	-128 ~ 127
[ signed ] char	有符号字符型	1	-128 ~ 127
unsigned char	无符号字符型	1	0~255
short [ int ]	短整型	2	-32 768 ~ 32 767
[ signed ] short [ int ]	有符号短整型	2	-32 768 ~ 32 767
unsigned short [ int ]	无符号短整型	2	0 ~ 65 535
int	整型	4	-2 147 483 648 ~ 2 147 483 647
[ signed ] int	有符号整型	4	-2 147 483 648 ~ 2 147 483 647
unsigned int	无符号整型	4	0 ~ 4 294 967 294
long [ int ]	长整型	4	-2 147 483 648 ~ 2 147 483 647
[ signed ] long [ int ]	有符号长整型	4	-2 147 483 648 ~ 2 147 483 647
unsigned long [ int ]	无符号长整型	4	0 ~ 4 294 967 294
float	单精度型	4	$-3.4 \times 10^{38} \sim 3.4 \times 10^{38}$
double	双精度型	8	$-1.7 \times 10^{308} \sim 1.7 \times 10^{308}$
long double	长双精度型	8	$-1.7 \times 10^{308} \sim 1.7 \times 10^{308}$



**C/C++保证: long的精度不低于int , int的精度不低于short;  
double的精度不低于float**



# 运算符优先级

优先级	运算符	功能及说明	结合性	目数
1	( )	改变运算优先级	左向右	双目
	::	作用域解析、全局		
	[ ]	数组下标		
	. ->	访问成员，由指针访问成员		
2	++ --	后缀自增、自减运算	右向左	单目
	new delete	动态分配或释放内存		
	++ --	前缀自增、自减运算符		
	*	间接访问 dereference		
	&	取地址 address-of		
	+ -	正、负号		
	!	逻辑非		
	~	按位取反		
	sizeof	取类型或表达式的字节长度		
	typeid	取表达式的类型信息		
	(type)expr	强制类型转换		
2*	.* ->*	由成员指针访问成员	左向右	双目

优先级	运算符	功能及说明	结合性	目数
3	* / %	乘、除、取余	左向右	双目
4	+ -	加、减	左向右	双目
5	<< >>	左移位、右移位	左向右	双目
6	< > <= >=	小于、大于、小于等于、大于等于	左向右	双目
7	== !=	等于、不等于	左向右	双目
8	&	按位与	左向右	双目
9	^	按位异或	左向右	双目
10		按位或	左向右	双目
11	&&	逻辑与	左向右	双目
12		逻辑或	左向右	双目
13	? :	条件运算	右向左	三目
14	= *= /= %= += -= <<= >>= &=  = ^=	赋值运算	右向左	双目
15	,	逗号运算	左向右	双目

# 位运算与乘除法

- $a \gg 1 \quad a / 2$
- $A \ll 2 \quad a * 4$



# 指针和引用

- 到底什么是引用？

# 传值与传引用

- 传值：传递的是值，对象的值
- 传引用：是传递对象本身
- 张三，32岁，值：“张三”，“32”；引用：张三（“张三”，“32”）
- 把张三的数据扔进垃圾桶
- 把张三扔进垃圾桶



Q & A