面向对象程序设计
420420

标准库

- 1、C++标准库 STL

- C++标准定义了庞大且功能丰富的标准库，其内容分为10类，包括：

- ①C1语言支持　②C2输入/输出　③C3诊断功能
- ④C4通用工具　⑤C5字符串　⑥C6容器
- ⑦C7迭代器　⑧C8算法　⑨C9数值操作
- ⑩C10本地化

- C++标准库所有的头文件都没有扩展名（.h），内容总共在51个标准头文件中定义。其中18个<cname>形式的头文件（<complex>除外）内容与标准C语言的name.h头文件相同，但包含了C++扩展的功能。

- 在<cname>形式的头文件中，与宏相关的名称在全局作用域中定义，其他名称在std命名空间中声明。

- 另外，在C++中还可以使用name.h形式的C语言头文件，但不建议这样用。

42.1 C++标准库

表42-1 C++标准库索引

分类	功能	头文件	功能	头文件
C1	C标准定义	<cstddef>	C++数值类型特性	<limits>
	整型大小	<climits>	浮点型特性	<cfloat>
	C标准实用工具	<cstdlib>	动态内存管理	<new>
	运行时类型信息	<typeinfo>	异常处理	<exception>
	可变参数	<cstdarg>	非局部跳转	<csetjmp>
	C中断处理	<csignal>		
C2	标准输入输出	<iostream>	输入输出操纵器	<iomanip>
	iostream基类	<ios>	输入输出前向声明	<iosfwd>
	输入流类	<istream>	输出流类	<ostream>
	字符串流类	<sstream>	文件流类	<fstream>
	流缓存类	<streambuf>	C标准输入输出	<cstdio>

续表42-1 C++标准库索引

分类	功能	头文件	功能	头文件
C3	异常类	<stdexcept>	C断言验证	<cassert>
	C出错码	<cerrno>		
C4	实用元件	<utility>	函数对象	<functional>
	内存管理器	<memory>	C时间日期	<ctime>
C5	字符串类	<string>	C字符串	<cstring>
	单字节字符类型	<cctype>	多字节字符类型	<cwctype>
	扩展多字节宽字符	<cwchar>	C字符串流类	<strstream>
C6	向量	<vector>	列表	<list>
	双队列	<deque>	队列	<queue>
	栈	<stack>	映射	<map>
	集合	<set>	位集	<bitset>

续表42-1 C++标准库索引

分类	功能	头文件	功能	头文件
C7	迭代器	<iterator>		
C8	算法	<algorithm>	ISO646字符集替换	<ciso646>
C9	复数	<complex>	数值矢量	<valarray>
	数学运算	<numeric>	C数学库	<cmath>
C10	本地化	<locale>	C语言本地化	<clocale>

- C++标准库由三组库构成：
- （1）C库：由C标准库扩展而来，强调结构、函数和过程，不支持面向对象技术。
- （2）C++库：增加了面向对象的库，包含了具有关键功能的类。
- （3）标准模板库（STL）：高效的C++程序库。该库包含了诸多在计算机科学领域里所常用的基本数据结构和基本算法。

- 以上库文件都被定义在namespace std中。
- C++库可以不规定头文件的扩展名。

- 1. C标准库
- 1）<cmath>：提供给了22个不同的数学函数，如abs()、sqrt()、exp()等。
- 2）<cstdlib>：提供了常用的数学函数，3个特殊函数和2个常量。
  - exit()：退出程序，清除静态对象。
  - abort()：强制退出程序。
  - atexit()：将一个函数作为参数并在程序退出时调用。
  - EXIT_SUCCESS：程序已经正常完成。
  - EXIT_FAULURE：程序正常结束。

- 3）<cassert>：定义了一个函数assert，用来测试一个条件，并在条件为假时强制程序退出。通常用来判断内存分配是否成功。

- 4）<cctype>：提供了确定字符类型的功能。可以判断一个字符是否为数字字符、图形字符、小写、大写或空格，并可以将字符进行大小写转换。

- 5）<cerrno>：定义了一个变量，用它可以确定程序中发生的错误的类型。

- 2. C++类库

- 1）<string>：支持字符串处理的类库。

- 2）<complex>：提供了复数的表示和算法。
  - float_complex类
  - double_conmplex类
  - long_double_complex类

- 3）<ios>：定义了I/O流类的虚基类ios，通常不直接用它。
- 4）<iostream>：标准I/O流类库，提供了cin、cout等全局对象类来支持输入输出功能。
- 5）<istream>：输入流类库。
- 6）<ostream>：输出流类库。
- 7）<fstream>：读写文件的类。

- 3. 标准模板库STL

- 在标准库中，容器、迭代器、算法和数值操作合称为标准模板库。
- STL被组织为以下13个头文件：<algorithm>、<deque>、<functional>、<iterator>、<vector>、<list>、<map>、<memory>、<numeric>、<queue>、<set>、<stack>和<utility>。
- 几乎所有的标准模板库代码都采用了类模板和函数模板的形式，因此相比于传统的由函数和类组成的库来说STL提供了更好的代码重用。

- 1）<algorithm>：提供给了通用的算数算法和STL的一般算法。
  - sort：以升序重新排列范围内的元素。
  - swap：交换存储在两个对象中的值。
  - max_element：指出序列中最大的元素。
  - find：对范围内的元素进行查找 。
  - copy：复制序列。
  - replace：将范围内的所有等于 old_value 的元素都用 new_value 替代。

- 2）<numeric>：提供给了4类对序列进行数字处理的算法。
  - accumulate()：累加
  - product()：内乘
  - Partial_sum()：部分和
  - adjacent_difference()：邻接差值

标准库

- 2、字符串流

- 字符串流是以内存中的string对象或字符数组（C风格字符串）为输入输出的对象，将数据输出到内存中的字符串存储区域，或者从字符串存储区域读入数据。

- 字符串流的作用是利用输入输出操作方式将各种类型的数据转换成字符序列，或者相反。由于计算机物理的或逻辑的设备大多数能处理的数据是字符序列（或字节序列），因此字符串流就是程序数据与设备进行数据交换重要的桥梁。

- 字符串流类有（istringstream、ostringstream、stringstream）和（istrstream、ostrstream、strstream）两种。前一种以C++ string串对象作为字符串，定义在头文件<sstream>中；后一种以字符数组作为字符串（即C风格字符串），定义在头文件<strstream>中。根据C++标准的定义，strstream是过时的，未来版本的C++标准可能不再支持strstream，因此推荐使用前一种。

- 使用字符串流类的方法是建立字符串流对象，通过构造函数或成员函数str()与某个字符串关联，此后对字符串流对象按输入输出形式进行操作。字符串流类的构造函数和常用成员函数原型如下：

```
istringstream(openmode which=ios_base::in); //构造字符串输入流对象
//构造字符串输入流对象，并初始化为str的内容
istringstream(const string& str,openmode which=ios_base::in);
ostringstream(openmode which=ios_base::out); //构造字符串输出流对象
//构造字符串输出流对象，并初始化为str的内容
ostringstream(const string& str,openmode which=ios_base::out);
//构造字符串输入输出流对象
stringstream(openmode which=ios_base::out|ios_base::in);
//构造字符串输入输出流对象，并初始化为str的内容
stringstream(const string & str,openmode which=ios_base::out|ios_base::in);
string str(); //返回当前字符串流缓冲区关联的字符串对象的副本
void str(const string & s); //复制字符串s内容到字符串流缓冲区关联的string对象中
```

- （1）ostringstream 对象用来进行格式化的输出，可以将各种类型转换为string类型，只支持 << 操作符，如：

```
ostringstream  oss;
oss<<1.234;     //double型数据1.234转化为字符串1.234
oss<<" ";       //字符串常量转化为字符串
oss<<1234;      //int型转化为字符串
oss<<endl;
cout<<oss.str();          //输出“1.234  1234”
oss.str("");              //每次使用前清空
oss<<3.14159;             //double型转化为字符串
Str2=oss.str();
cout<<Str2<<endl;     //显示器上输出“3.14159”
```

- （2）istringstream 对象用于把一个已定义字符串中的以空格隔开的内容提取出来，只支持 >> 操作符，如：

```
int  a;                  //整型的变量a
string  str1;        //string类型的对象str1
string  input="abc  123  def  456  ghi  789";    //string类型的对象input初始化
istringstream  iss(input);  //通过构造函数对iss进行赋值
while(iss>>str1>>a)    cout<<str1<<"  "<<a<<endl;
```

- 运行结果：

```
abc  123
def  456
ghi  789
```

- （3）stringstream类是istringstream和ostringstream类的综合，支持<<和>>操作符。

```
int a; string str1;
string input="abc 123 def 456 ghi 789";
stringstream ss;
ss<<input;                    //将input输出到ss里
while(ss>>str1>>a)    cout<<str1<<" "<<a<<endl;
```

- 运行结果：

```
abc 123
def 456
ghi 789
```

【例42.1】输入一个加减运算的表达式，输出其值。

```
1  #include  <iostream>
2  #include  <sstream>  //使用字符串流
3  using  namespace  std;  //字符串流定义在std命名空间
4  int  main()
5  {  //字符串流示例
6      string  s1,  s2;          //定义2个string对象
7      ostringstream  oss;  //字符串输出流oss
8      istringstream  iss;     //字符串输入流iss
9      char  c1='+',  c2;
10     double  val,  sum=0.0;
11     cin>>s2;  //从键盘上输入字符串，比如”1000+200-30+4+0.4”
12     iss.str(s2);  //复制s2内容到字符串输入流
```

```
13      while (c1 != ' ') {
14          iss>>val>>c2;  //从iss中提取一个数字和字符
15          if (c1=='+') sum=sum+val;  //由值前面的运算符决定运算
16          else if (c1=='-') sum=sum-val;
17          c1=c2, c2=' ';  //保存当前运算符下次用
18      }
19      oss<<sum; s1 = oss.str();//运算结果输出到字符串流中,s1是其副本
20      cout<<s1<<endl;  //输出s1到显示器
21      return 0;
22  }
```