**using声明**

#include <stdio.h>

#include <iostream>

using std::cin; // 可以直接使用cin，无需携程std::cin

using namespace std; // 使用std命名空间，可以省略std

int main()

{

cout << "Enter Continue" << endl;

cin.get();

return 0;

}

**标准库类型string**

#include <string> // 引入string标准库

string s1;

string s1;

string s2 = string("a"); // ???，为什么没有new，c++不用new新建对象，string("a")为构造函数

string s3 = "a"; // ???，为什么这样子可以，可能是string重载的赋值运算符

**for循环string**

string s = "abc";

for(auto &c : s){

c = toupper(c); // 这里可以直接操作是因为 auto &c，使用的是引用类型

cout << c << endl;

}

cout << s << endl; // 输出大写 ABC

**字串查找**

// find 查找子串，返回子串第一次出现的位置

if(str1.find(str2) != str1.npos){

    // str1 不包含 str2

}

**标准库vector**

vector<string> list = {"ABC", "DEF"};

list.push\_back("GH"); // 添加

int size = list.size();

list[0] = "AB";

**迭代器**

vector<string> list = {"ABC", "DEF"};

// 之所以能这样使用是因为vector实现了迭代器，如下这两段代码是一样的

for(auto item : list)

{

cout << item << endl;

}

// 迭代器有begin()和end()

// begin()返回集合开始指针

// end()返回集合结束指针的后一个指针，其指向的内容也就是空的

for(auto p = list.begin(); p != list.end(); p++)

{

auto &item = \*p;

cout << item << endl;

}

**数组**

// 数组的大小在初始化时确定

string l[2] = {"ABC", "DEF"};

string list[] = {"ABC", "DEF"};

for(auto item : list)

{

cout << item << endl;

}

**指针于数组**

string list[] = {"ABC", "DEF"};

string \*startP = list; // list即 list[0]的地址

string \*endP = &list[2];

for(auto p = startP; p != endP; p++){

cout << \*p << endl;

}

cout << endP - startP << endl; // 输出2，数组大小

**多维数组**

string lists[][3] = {

{"ABC", "DEF"},

{"A", "B"},

{"A", "C"}

};

cout << lists[2][1] << endl; // 输出c

**->运输符**

p->Felid1 等价于 (\*p).Felid1

**标准库函数begin(), end()**

int list[] = {1, 2};

int \*startList = begin(list); // 获取数组的开始位置

int \*endList = end(list); // 获取数组的结束位置