(1)#include <iostream>

using namespace std;

struct STRING

{

char \*str;//字符指针指向字符串

int ct;//字符串的长度

};

//定义函数 set show

void set(STRING &str,const char \*pstr)//添加函数参数

{

int i=0;

for(;pstr[i];i++)

str.ct=i;

str.str=new char[i];

//按要求3 补充代码

for(i=0;pstr[i];i++)

str.str[i]=pstr[i];

}

void show(STRING str,int n=0) //补充参数和执行代码

{

int i;

if(n==0)

printf("%s\n",str.str);

else

for(i=0;i<n;i++)

{

printf("%c",str.str[i]);

}

printf("\n");

}

void show(const char str[],int n=0) //补充参数和执行代码

{

int i;

if(n==0)

printf("%s\n",str);

else

for(i=0;i<n;i++)

printf("%c",str[i]);

printf("\n");

}

int main()

{

STRING str;

const char str\_1[]="hello world";

set(str,str\_1);//注意(1)第一个参数是引用

//(2)分配空间给STRING中str（3）把str\_1中的数据复制到str中

//（4）计算ct=字符串大小

show(str);//输出str字符串

show(str,3);//输出str前三个字符

show(str\_1);//输出str\_1

show(str\_1,3);//输出str\_1前三个字符

show("finished");

return 0;

}

(2)throw抛出的数据由程序设计者自定，可以是任何类型的数据，return只能返回一个值。

throw可以把安全的跨一个或多个函数，使得异常引发和处理机制分离更高效，return只会终止函数的执行并返回函数的值，不存在异常处理。

(3)#include <iostream>

using namespace std;

struct A{};

A a;

double Div(int x,int y)

{

if(y==0)

throw a;

return x/y;

}

int main()

{

int x,y;

TRY:

cin>>x>>y;

try

{

cout<<x<<"/"<<y<<"="<<Div(x,y)<<endl;

}

catch(A)

{

cout<<"错误：除数为0！"<<endl;

goto TRY;

}

return 0;

}

1. Terminate called after throwing an instance of ‘std::\_\_cxxll::basic\_string<char,std::char\_traits<char,std::allocator<char>>’

直接调用terminate程序终止。