

6 异常

6.1 非本地控制转移

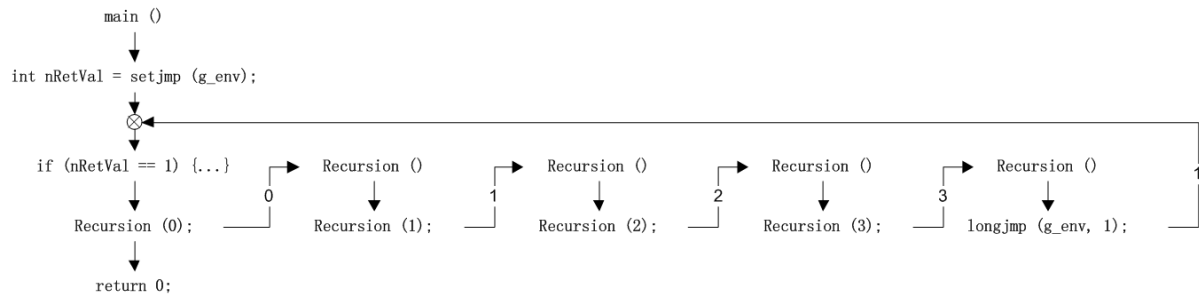
6.1.1 setjmp-longjmp机制

```
1 // longjmp.cpp
2
3 // setjmp-longjmp机制
4
5 #include <setjmp.h>
6
7 #include <iostream>
8
9 using namespace std;
10
11 jmp_buf jb;
12
13 class A {
14 public:
15     A(int n) : n(new int(n)) {
16         cout << "A::A(" << *this->n << ") invoked" << endl;
17     }
18
19     ~A(void) {
20         cout << "A::~A(" << *n << ") invoked" << endl;
21         delete n;
22     }
23
24 private:
25     int* n;
26 };
27
28 void foo(void) {
29     A a(3);
30     longjmp(jb, 404);
31 }
32
33 void bar(void) {
34     A a(2);
35     foo();
36 }
37
38 void hum(void) {
39     A a(1);
40     bar();
41 }
42
43 int main(void) {
44     int errcode = 0;
45
46     if (errcode = setjmp(jb)) {
47         cout << "Some error(" << errcode << ") happened!" << endl;
```

```

48         return -1;
49     }
50
51     hum();
52
53     return 0;
54 }

```



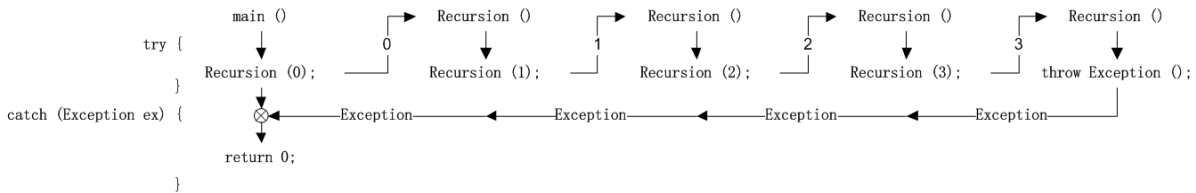
6.2.2 try-catch机制

```

1  // exception.cpp
2
3  // try-catch机制
4
5  #include <iostream>
6
7  using namespace std;
8
9  class A {
10 public:
11     A(int n) : n(new int(n)) {
12         cout << "A::A(" << *this->n << ") invoked" << endl;
13     }
14
15     ~A(void) {
16         cout << "A::~A(" << *n << ") invoked" << endl;
17         delete n;
18     }
19
20 private:
21     int* n;
22 };
23
24 void foo(void) {
25     A a(3);
26     throw 404;
27 }
28
29 void bar(void) {
30     A a(2);
31     foo();
32 }
33
34 void hum(void) {
35     A a(1);
36     bar();

```

```
37 }
38
39 int main(void) {
40     try {
41         hum();
42     }
43     catch (int errcode) {
44         cout << "Some error(" << errcode << ") happened!" << endl;
45         return -1;
46     }
47
48     return 0;
49 }
```



6.2.3 几种错误处理机制对比

对比项	逐层检查返回值	setjmp-longjmp机制	try-catch机制
流程	复杂	简单	简单
栈对象	全析构	不析构	全析构

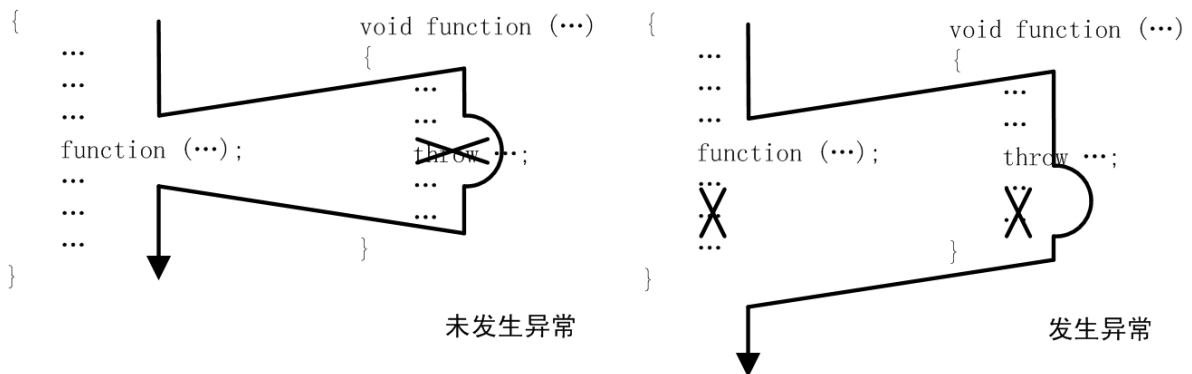
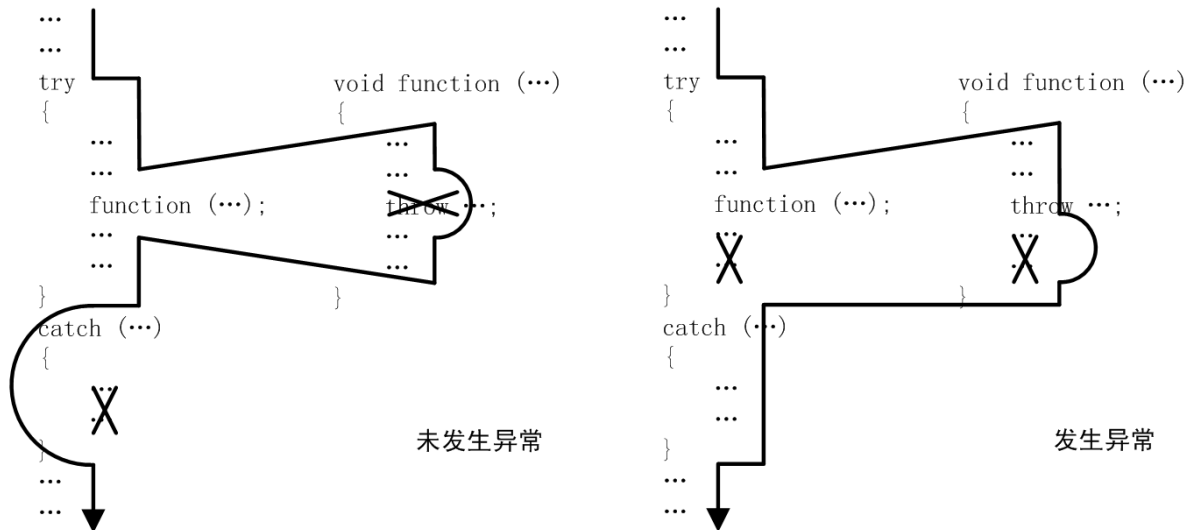
6.2 异常处理流程

```
1 // exflow.cpp
2
3 // 异常处理流程
4
5 #include <iostream>
6
7 using namespace std;
8
9 void foo(void) {
10     cout << "Before throw" << endl; // 执行
11
12     throw -1;
13
14     cout << "After throw" << endl; // 不执行
15 }
16
17 int main(void) {
18     try {
19         cout << "Before invoke" << endl; // 执行
20
21         foo();
22
23         cout << "After invoke" << endl; // 不执行
24     }
```

```

25     catch (int ex) {
26         cout << "Caught exception" << endl; // 执行
27     }
28
29     cout << "After try-catch" << endl; // 执行
30
31     return 0;
32 }

```



6.3 异常处理模式

- throw语句抛出基本类型对象，catch子句捕获基本类型异常

```

1 // usage1.cpp
2
3 // throw语句抛出基本类型对象，catch子句捕获基本类型异常
4
5 #include <iostream>
6
7 using namespace std;
8
9 void login(const string& username, const string& password) {
10     if (username != "tarena")
11         throw 1;
12
13     if (password != "123456")
14         throw 2;

```

```

15 }
16
17 int main(int argc, char* argv[]) {
18     if (argc < 3) {
19         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
20         return -1;
21     }
22
23     try {
24         login(argv[1], argv[2]);
25     }
26     catch (int ex) {
27         if (ex == 1)
28             cout << "Username error" << endl;
29         else
30             if (ex == 2)
31                 cout << "Password error" << endl;
32
33         cout << "Login fail" << endl;
34
35         return -1;
36     }
37
38     cout << "Login success" << endl;
39
40     return 0;
41 }

```

- throw语句抛出类类型对象，catch子句捕获类类型异常

```

1 // usage2.cpp
2
3 // throw语句抛出类类型对象，catch子句捕获类类型异常
4
5 #include <iostream>
6
7 using namespace std;
8
9 class UsernameException {};
10 class PasswordException {};
11
12 void login(const string& username, const string& password) {
13     if (username != "tarena")
14         throw UsernameException();
15
16     if (password != "123456")
17         throw PasswordException();
18 }
19
20 int main(int argc, char* argv[]) {
21     if (argc < 3) {
22         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
23         return -1;
24     }

```

```

25
26     try {
27         login(argv[1], argv[2]);
28     }
29     catch (UsernameException) {
30         cout << "Username error" << endl;
31         cout << "Login fail" << endl;
32         return -1;
33     }
34     catch (PasswordException) {
35         cout << "Password error" << endl;
36         cout << "Login fail" << endl;
37         return -1;
38     }
39
40     cout << "Login success" << endl;
41
42     return 0;
43 }

```

- 对于存在继承关系的异常，对子类异常的捕获一定要先于对基类异常的捕获

```

1  // usage3.cpp
2
3  // 对于存在继承关系的异常，对子类异常的捕获一定要先于对基类异常的捕获
4
5  #include <iostream>
6
7  using namespace std;
8
9  class UsernameException {};
10 class PasswordException : public UsernameException {};
11
12 void login(const string& username, const string& password) {
13     if (username != "tarena")
14         throw UsernameException();
15
16     if (password != "123456")
17         throw PasswordException();
18 }
19
20 int main(int argc, char* argv[]) {
21     if (argc < 3) {
22         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
23         return -1;
24     }
25
26     try {
27         login(argv[1], argv[2]);
28     }
29     catch (PasswordException) { // 子类
30         cout << "Password error" << endl;
31         cout << "Login fail" << endl;
32         return -1;
33     }

```

```

34     catch (UsernameException) { // 基类
35         cout << "Username error" << endl;
36         cout << "Login fail" << endl;
37         return -1;
38     }
39
40     cout << "Login success" << endl;
41
42     return 0;
43 }

```

- 在catch块中重新抛出异常

```

1  // usage4.cpp
2
3  // 在catch块中重新抛出异常
4
5  #include <iostream>
6
7  using namespace std;
8
9  class UsernameException {};
10 class PasswordException {};
11 class DownloadException {};
12
13 void login(const string& username, const string& password) {
14     if (username != "tarena")
15         throw UsernameException();
16
17     if (password != "123456")
18         throw PasswordException();
19 }
20
21 void download(const string& username, const string& password) {
22     try {
23         login(username, password);
24     }
25     catch (UsernameException) {
26         cout << "Username error" << endl;
27         cout << "Login fail" << endl;
28         throw DownloadException();
29     }
30     catch (PasswordException) {
31         cout << "Password error" << endl;
32         cout << "Login fail" << endl;
33         throw DownloadException();
34     }
35
36     cout << "Login success" << endl;
37
38     cout << "Downloading..." << endl;
39 }
40
41 int main(int argc, char* argv[]) {
42     if (argc < 3) {

```

```

43     cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
44     return -1;
45 }
46
47 try {
48     download(argv[1], argv[2]);
49 }
50 catch (DownloadException) {
51     cout << "Download fail" << endl;
52     return -1;
53 }
54
55 cout << "Download success" << endl;
56
57 return 0;
58 }

```

- 一个函数可以收到被其调用的函数所抛出的异常，但不一定非要捕获并处理该异常

```

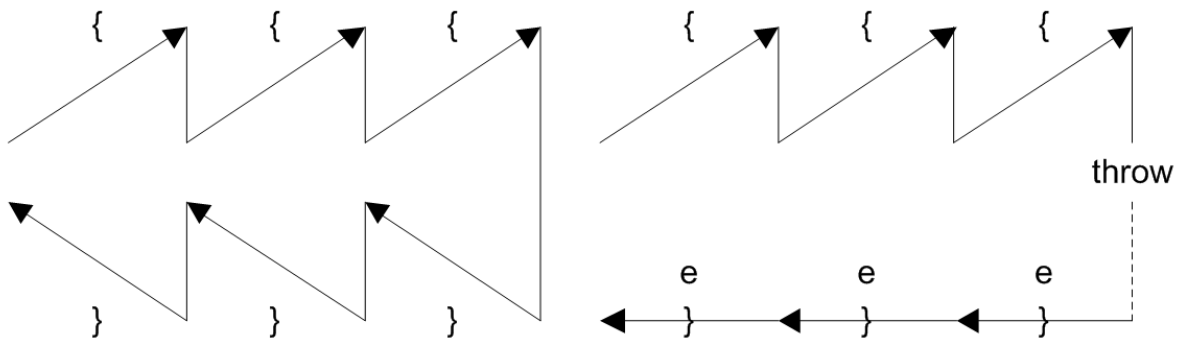
1  // usage5.cpp
2
3  // 一个函数可以收到被其调用的函数所抛出的异常，但不一定
4  // 非要捕获并处理该异常。该异常将从此函数中继续向上抛出
5
6  #include <iostream>
7
8  using namespace std;
9
10 class UsernameException {};
11 class PasswordException {};
12
13 void login(const string& username, const string& password) {
14     if (username != "tarena")
15         throw UsernameException();
16
17     if (password != "123456")
18         throw PasswordException();
19 }
20
21 void download(const string& username, const string& password) {
22     login(username, password);
23
24     cout << "Downloading..." << endl;
25 }
26
27 int main(int argc, char* argv[]) {
28     if (argc < 3) {
29         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
30         return -1;
31     }
32
33     try {
34         download(argv[1], argv[2]);
35     }

```

```

36     catch (UsernameException) {
37         cout << "Username error" << endl;
38         cout << "Download fail" << endl;
39         return -1;
40     }
41     catch (PasswordException) {
42         cout << "Password error" << endl;
43         cout << "Download fail" << endl;
44         return -1;
45     }
46
47     cout << "Download success" << endl;
48
49     return 0;
50 }

```



未发生异常

发生异常

- 包含更多有用信息的异常

```

1  // usage6.cpp
2
3  // 包含更多有用信息的异常
4
5  #include <iostream>
6
7  using namespace std;
8
9  class Exception {
10 public:
11     Exception(const char* file, int line, const char* func, const char*
desc)
12         : file(file), line(line), func(func), desc(desc) {}
13
14 private:
15     string file;
16     int line;
17     string func;
18     string desc;
19
20     friend ostream& operator<<(ostream& os, const Exception& ex) {
21         return os << ex.file << ':' << ex.line << '(' << ex.func << "):
" << ex.desc;
22     }
23 };

```

```

24
25 class UsernameException : public Exception {
26 public:
27     UsernameException(const char* file, int line, const char* func)
28         : Exception(file, line, func, "Username error") {}
29 };
30
31 class PasswordException : public Exception {
32 public:
33     PasswordException(const char* file, int line, const char* func)
34         : Exception(file, line, func, "Password error") {}
35 };
36
37 void login(const string& username, const string& password) {
38     if (username != "tarena")
39         throw UsernameException(__FILE__, __LINE__, __func__);
40
41     if (password != "123456")
42         throw PasswordException(__FILE__, __LINE__, __func__);
43 }
44
45 int main(int argc, char* argv[]) {
46     if (argc < 3) {
47         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
48         return -1;
49     }
50
51     try {
52         login(argv[1], argv[2]);
53     }
54     catch (const UsernameException& ex) {
55         cout << ex << endl;
56         cout << "Login fail" << endl;
57         return -1;
58     }
59     catch (const PasswordException& ex) {
60         cout << ex << endl;
61         cout << "Login fail" << endl;
62         return -1;
63     }
64
65     cout << "Login success" << endl;
66
67     return 0;
68 }

```

- 异常对象最好以引用而非对象的形式捕获，避免因对象复制增加额外的开销，或引发新的异常
- 通过修改异常对象的状态，跟踪异常处理过程

```

1 // usage7.cpp
2
3 // 通过修改异常对象的状态，跟踪异常处理的过程
4

```

```

5  #include <iostream>
6
7  using namespace std;
8
9  void foo(void) {
10     throw string(__func__);
11 }
12
13 void bar(void) {
14     try {
15         foo();
16     }
17     catch (string& ex) {
18         throw (ex += "->") += __func__;
19     }
20 }
21
22 void hum(void) {
23     try {
24         bar();
25     }
26     catch (string& ex) {
27         throw (ex += "->") += __func__;
28     }
29 }
30
31 int main(void) {
32     try {
33         hum();
34     }
35     catch (string& ex) {
36         cout << ((ex += "->") += __func__) << endl;
37         return -1;
38     }
39
40     return 0;
41 }

```

- 捕获其它异常

```

1  // usage8.cpp
2
3  // 捕获其它异常
4
5  #include <iostream>
6
7  using namespace std;
8
9  class UsernameException {};
10 class PasswordException {};
11 class HostnameException {};
12
13 void login(const string& username, const string& password) {
14     if (username != "tarena")
15         throw UsernameException();
16 }

```

```

17     if (password != "123456")
18         throw PasswordException();
19 }
20
21 void download(const string& username, const string& password, const
string& hostname) {
22     login(username, password);
23
24     if (hostname != "www.tedu.cn")
25         throw HostnameException();
26
27     cout << "Downloading..." << endl;
28 }
29
30 int main(int argc, char* argv[]) {
31     if (argc < 4) {
32         cout << "Usage: " << argv[0] << " <username> <password>
<hostname>" << endl;
33         return -1;
34     }
35
36     try {
37         download(argv[1], argv[2], argv[3]);
38     }
39     catch (UsernameException) {
40         cout << "Username error" << endl;
41         cout << "Download fail" << endl;
42         return -1;
43     }
44     catch (PasswordException) {
45         cout << "Password error" << endl;
46         cout << "Download fail" << endl;
47         return -1;
48     }
49     catch (...) {
50         cout << "Other error" << endl;
51         cout << "Download fail" << endl;
52         return -1;
53     }
54
55     cout << "Download success" << endl;
56
57     return 0;
58 }

```

6.4 构造函数中的异常

- 构造函数中出现无法处理的问题时，可以抛出异常以告知对象的创建者
- 构造函数抛出异常，引发不完整构造，析构函数不会被调用，已经分配的资源需要在构造函数中释放，或使用智能指针

```

1 // consex.cpp
2
3 // 构造函数抛出异常

```

```
4
5 #include <memory>
6 #include <iostream>
7
8 using namespace std;
9
10 class A {
11 public:
12     A(void) {
13         cout << "A::A() invoked" << endl;
14     }
15
16     ~A(void) {
17         cout << "A::~A() invoked" << endl;
18     }
19 };
20
21 class B {
22 public:
23     B(void) {
24         cout << "B::B() invoked" << endl;
25     }
26
27     ~B(void) {
28         cout << "B::~B() invoked" << endl;
29     }
30 };
31
32 class C {
33 public:
34     C(void) {
35         cout << "C::C() invoked" << endl;
36     }
37
38     ~C(void) {
39         cout << "C::~C() invoked" << endl;
40     }
41 };
42
43 class D {
44 public:
45     D(void) {
46         cout << "D::D() invoked" << endl;
47     }
48
49     ~D(void) {
50         cout << "D::~D() invoked" << endl;
51     }
52 };
53
54 class E : public D {
55 public:
56     E(void) : b(new B), a(new A) {
57         cout << "E::E() invoked" << endl;
58
59         delete b; // 注意!
```

```

60         throw 1;
61     }
62
63     ~E(void) {
64         cout << "E::~~E() invoked" << endl;
65
66         delete b;
67     }
68
69 private:
70     C c;
71     B* b;
72     auto_ptr<A> a;
73 };
74
75 int main(void) {
76     try {
77         E e;
78     }
79     catch (int ex) {
80         cout << "Caught an exception: " << ex << endl;
81         return -1;
82     }
83
84     return 0;
85 }

```

6.5 析构函数中的异常

- 不要主动抛出
- 尽量内部消化

```

1  // desex.cpp
2
3  // 析构函数不要抛出异常
4
5  #include <iostream>
6
7  using namespace std;
8
9  class A {
10 public:
11     ~A(void) {
12         cout << "A::~~A() invoked" << endl;
13
14         throw 1;
15     }
16
17     void foo(void) {
18         cout << "A::foo() invoked" << endl;
19
20         throw 2;
21     }
22 };

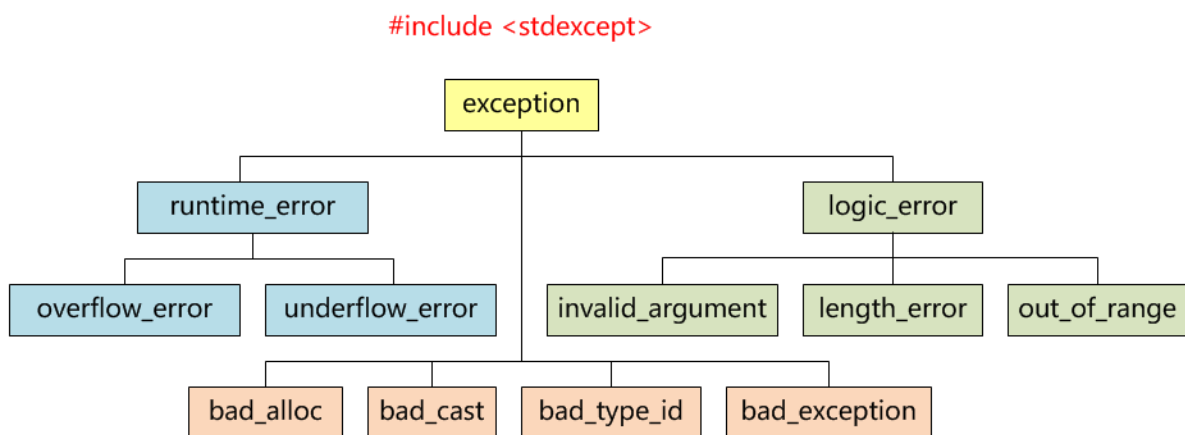
```

```

23
24 int main(void) {
25     try {
26         A a;
27         a.foo();
28     }
29     catch (int ex) {
30         cout << "Caught an exception: " << ex << endl;
31         return -1;
32     }
33
34     return 0;
35 }

```

6.6 标准库异常



- 从标准库异常中派生自己的异常，并覆盖exception异常基类中的虚函数what，返回有关异常的描述字符串，同时以多态方式，将自己的异常和标准库中的各种异常一并捕获

```

1 // usage9.cpp
2
3 // 从标准库异常中派生自己的异常，并覆盖exception异常基
4 // 类中的虚函数what，返回有关异常的描述字符串，同时以
5 // 多态方式，将自己的异常和标准库中的各种异常一并捕获
6
7 #include <iostream>
8
9 using namespace std;
10
11 class UsernameException : public exception {
12 public:
13     const char* what(void) const noexcept {
14         return "Username error";
15     }
16 };
17
18 class PasswordException : public exception {
19 public:
20     const char* what(void) const noexcept {
21         return "Password error";
22     }
23 };

```

```
24
25 void login(const string& username, const string& password) {
26     if (username != "tarena")
27         throw UsernameException();
28
29     if (password != "123456")
30         throw PasswordException();
31 }
32
33 int main(int argc, char* argv[]) {
34     if (argc < 3) {
35         cout << "Usage: " << argv[0] << " <username> <password>" <<
endl;
36         return -1;
37     }
38
39     try {
40         login(argv[1], argv[2]);
41     }
42     catch (const exception& ex) {
43         cout << ex.what() << endl;
44         cout << "Login fail" << endl;
45         return -1;
46     }
47
48     cout << "Login success" << endl;
49
50     return 0;
51 }
```