【原理基础】有 return 的情况下 try catch finally 的执行顺序

作者：moonce

原文链接：https://ld246.com/article/1539616331806

来源网站：链滴

概念

任何执行try 或者catch中的return语句之前，都会先执行finally语句，如果finally存在的话。如果finaly中有return语句，那么程序就return了，所以finally中的return是一定会被return的，编译器把finall中的return实现为一个warning。

举例

1.按顺序执行

```
try{
  //...
} catch(){
  //...
}finally{
  //...
}
return;
```

2.程序执行try块中return之前（包括return语句中的表达式运算）代码,再执行finally块，最后执行try return;finally块之后的语句return，因为程序在try中已经return所以不再执行。

```
try{
  //...
  return;
}catch(){
  //...
} finally{
  //...
}
return;
```

3.程序先执行try，如果遇到异常执行catch块，有异常：则执行catch中return之前（包括return语句的表达式运算）代码，再执行finally语句中全部代码，最后执行catch块中return. finally之后也就是4的代码不再执行。无异常：执行完try再finally再return.

```
try{
  //...
} catch(){
  //...
  return;
} finally{
  //...
}
return;
```

4.程序执行try块中return之前（包括return语句中的表达式运算）代码,再执行finally块，因为finally中有return所以提前退出。

```
try{
  //...
  return;
}catch(){
```

```
  //...
} finally{
  //...
  return;
}
```

5.程序执行catch块中return之前（包括return语句中的表达式运算）代码,再执行finally块，因为finall块中有return所以提前退出。

```
try{
  //...
}catch(){
  //...
  return;
} finally{
  //...
  return;
}
```

6.程序执行try块中return之前（包括return语句中的表达式运算）代码；有异常：执行catch块中retun之前（包括return语句中的表达式运算）代码,则再执行finally块，因为finally块中有return所以提退出。无异常：则再执行finally块，因为finally块中有return所以提前退出。

```
try{
  //...
  return;
}catch(){
  //...
  return;
} finally{
  //...
  return;
}
```

7.在try语句中，在执行return语句时，要返回的结果已经准备好了，就在此时，程序转到finally执行。在转去之前，try中先把要返回的结果存放到不同于x的局部变量中去，执行完finally之后，在从中出返回结果，因此，即使finally中对变量x进行了改变，但是不会影响返回结果。它应该使用栈保存返值。

```
public class FinallyTest {
  public static void main(String[] args) { //返回结果为2
  }
  static int test(){
    int x = 1;
    try{
      x++;
      return x;
    }finally{
      ++x;
    }
  }
}
```

总结

- 不管有木有出现异常，finally块中代码都会执行;
- 当try和catch中有return时，finally仍然会执行;
- finally是在return后面的表达式运算后执行的（此时并没有返回运算后的值，而是先把要返回的值存起来，管finally中的代码怎么样，返回的值都不会改变，任然是之前保存的值），所以函数返回值在finally执行前确定的;
- finally中最好不要包含return，否则程序会提前退出，返回值不是try或catch中保存的返回值。