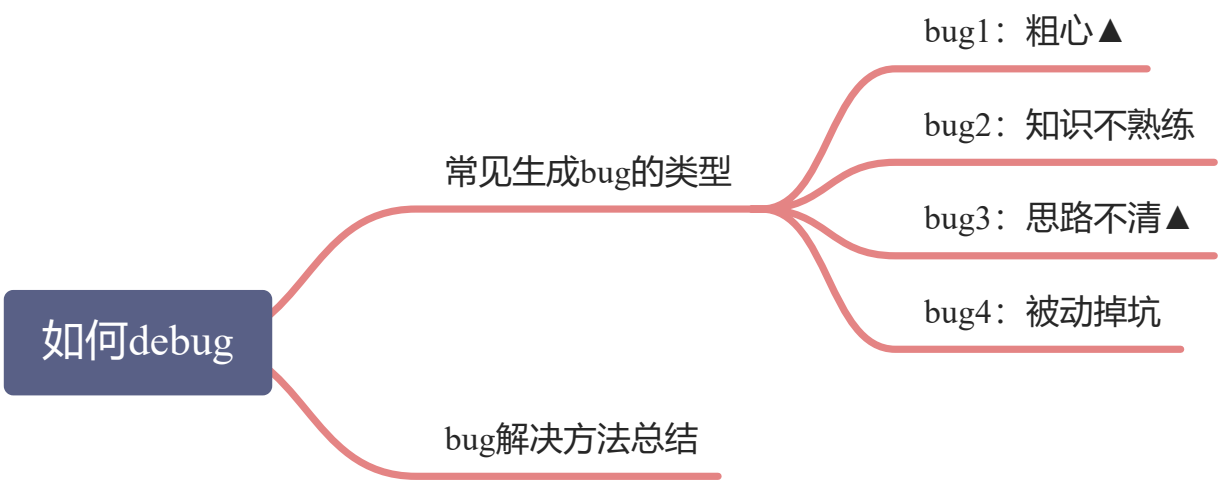
第11课 如何debug

一、课程结构导图



注：▲为重点知识点。

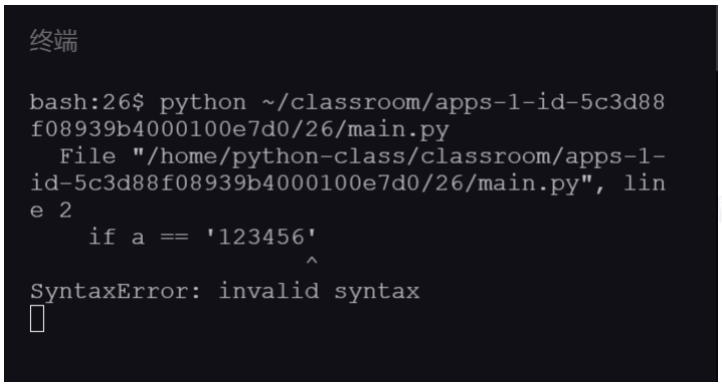
二、常见生成bug的类型

2.1 粗心

错误示例：

```
a = input('请输入密码: ')
if a == '123456'
    print('通过')
```

报错显示：



报错分析：

1. 报错显示为语法错误，当我们遇到终端报错时，可以按照以下三个步骤去排查：
   a. 报错的代码行是第几行
   b. 指向的报错位置在哪里
   c. 报错类型是什么
2. 根据终端代码的报错显示，报错行是在第2行（line 2），报错的位置是if语句的末尾（^），报错类型是语法错误（SytnaxEroor）。
3. 仔细观察一下，可以发现，if语句末尾少了冒号，导致终端报错语法错误的显示。

正确示例：

```
a = input('请输入密码: ')
if a == '123456':
    print('通过')
```

自检清单：

自检清单
1、漏了末尾的冒号，如if语句，while语句、定义函数
2、缩进错误，该缩进的时候没有缩进
3、把英文字符写成中文字符，如''，{}
4、字符串拼接错误，如字符串与数字拼接在一起

5、没有定义变量就使用
6、==（等于号）与=（赋值号）混用

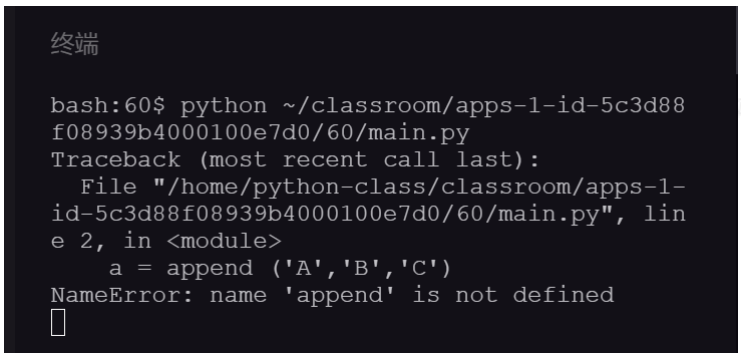
注：因为粗心导致的bug可能所占的比重最大，这里提供一份自检清单。

2.2 知识不够熟练

错误示例：

```
1 a = []
2 a = append ('A','B','C')
3 print(a)
```

报错显示：



报错分析：

1. 终端报错显示是第2行代码报错，没有指出报错位置，报错类型是append未声明。
2. 我们通过思考，append是一个函数，而append()函数的用法是list.append(元素)，通过知识点使用的回顾，能明显的知道append()函数的用法使用错误了。
3. 当你发现知识点记不清或者不能确定的时候，就要及时复习或者上网搜索。不要强行写出自己不敢确定的代码，这种情况往往容易出错。

正确示例：

```
a = []
a.append('A')
a.append('B')
a.append('C')
print(a)
```

2.3 思路不清

错误示例：

```
import random
guess = ''
while guess not in ['正面','反面']:
    print('------猜硬币游戏------')
    print('猜一猜硬币是正面还是反面？')
    guess = input('请输入“正面”或“反面”：')

toss = random.randint(0,1)

if toss == guess:
    print('猜对了！你真棒')
else:
    print('没猜对，你还有一次机会。')
    guess = input('再输一次“正面”或“反面”：')
    if toss == guess:
        print('你终于猜对了！')
    else:
        print('大失败！')
```

思路分析：

1. 该代码并没有出现报错，但运行代码后，永远的结果都是'大失败！'。
2. 代码中语法使用都是正确的，语法的格式也没有出现纰漏，那么很大可能是因为代码逻辑发生了错误，也就是我们经常会遇到的情况，即代码没有报错，但运行结果却没有达到预期结果。
3. 当我们遇到代码逻辑不清的情况时，我们可以通过用''#''或三引号暂时注释掉部分代码，使用print()函数打印重要的变量结果进行观察（如下代码）。

```
import random
guess = ''
while guess not in ['正面','反面']:
    print('------猜硬币游戏------')
    print('猜一猜硬币是正面还是反面？')
    guess = input('请输入“正面”或“反面”：')
    print(guess)  # 打印guess变量

toss = random.randint(0,1)
print(toss) # 打印toss变量

# 注释代码
'''
if toss == guess:
    print('猜对了！你真棒')
else:
    print('没猜对，你还有一次机会。')
    guess = input('再输一次“正面”或“反面”：')
    if toss == guess:
        print('你终于猜对了！')
    else:
        print('大失败！')
'''
```

4. 通过打印两个变量（guess和toss）的结果，我们可以观察到，guess变量的值永远是正面或者反面，而toss变量的值永远是0或1，guess是一个字符串，而toss是一个整数，两者不可能相等，所以代码中的if语句是不可能成立的。
5. 发现了以上的逻辑错误后，我们可以通过修改guess的输入提示，去达到我们预期的目标。

正确示例：

```
import random
guess = ''

while guess not in [0,1]:
    print('------猜硬币游戏------')
    print('猜一猜硬币是正面还是反面？')
    guess = int(input('“正面”请输入0,“反面”请输入1：'))
    # 提示用户正面输入0，反面输入1，并强制转换为整数型。
```

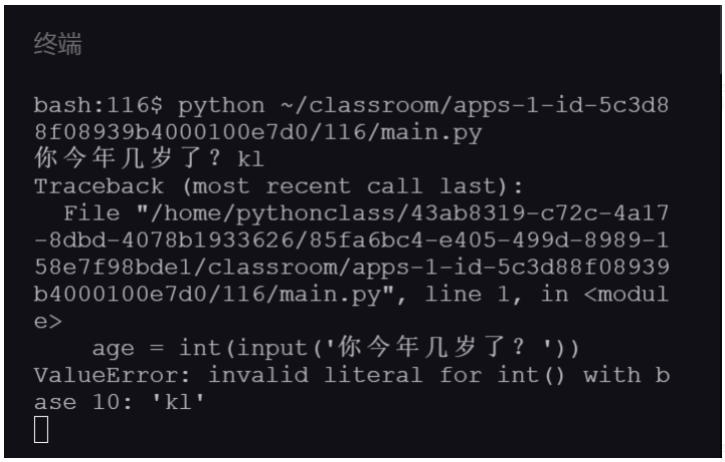
```
9
10 toss = random.randint(0,1)
11
12 if toss == guess:
13     print('猜对了！你真棒')
14 else:
15     print('没猜对，再给你一次机会。')
16     guess = int(input('再输一次（“正面”请输入0,“反面”请输入1）：'))
17     if toss == guess:
18         print('你终于猜对了！')
19     else:
20         print('大失败！')
```


2.4 被动掉坑

错误示例：

```
1 age = int(input('你今年几岁了？'))
2 if age < 18:
3     print('不可以喝酒噢')
```

报错显示：



报错分析：

1. 根据终端代码的显示，报错行是在第1行，没有指出报错位置，报错类型是传入无效参数（ValueError）。
2. 检查代码语法，格式和逻辑，都没有发现错误，在这种情况下，我们进入了被动掉坑，被动掉坑是指有时候代码逻辑上并没有错，但可能因为用户的错误操作或者是一些“例外情况”而导致程序崩溃。
3. 我们可以通过try....except语句捕获异常。当出现异常报错时（用户执行错误的操作），执行except语句。

正确示例：

```
while True:
    try:
        age = int(input('你今年几岁了?'))
        break
    except ValueError:
        print('要输入整数哟~')
if age < 18:
        print('不可以喝酒噢')
```

代码解析：try...expect语句是一个整体（可类比为if....else）。try语句为捕获的语句，也就是第3行代码，当用户输入的是整数时，会执行break语句，跳出while循环，执行if语句进行判断。当用户输入的是非整数时，会执行except语句，提示用户需要输入的数值是整数，返回执行第三行代码，直到用户输入的值是整数，才会停止循环。

三、bug解决方法总结

方法总结：

1. 终端出现报错时，要先从终端中查找出报错的代码行是第几行，指向的报错位置在哪里，报错类型是什么。
2. 针对粗心造成的bug，可通过报错指向的报错位置，仔细的观察。
3. 针对知识点不熟造成的bug，要记得多复习，查阅笔记，针对性地做练习掌握用法。
4. 针对思维不清的bug，要多用print()函数和#注释一步步地排查错误。
5. 针对容易被忽略的例外情况从而被动掉坑的bug，可以用try...except语句让程序顺利运行。

关于所有报错类型查询网站：https://www.runoob.com/python/python-exceptions.html <https://www.runoob.com/python/python-exceptions.html> 。