

# 调试概要

18级信息安全 李昊翔

yoj常见问题和相应对策：

Compile Error （观察代码错误）

Memory Limit Exceeded （观察代码错误）

Runtime Error （调试）

Time Limit Exceeded （观察代码错误或调试）

Wrong Answer （调试）

# Compile Error (编译错误)

- 是否没引入头文件
  - 是否写错了C语言关键词
  - 是否使用了未定义的变量
  - .....
- 这一类问题要去看vscode控制台一栏，“问题”和“终端”中的输出信息



# Memory Limit Exceeded (超出内存限制)

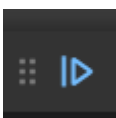
- 声明了过长的数组、字符串等
- 分配了过多的栈、堆空间
- 这一类问题要仔细检查代码，减少程序内存分配

# Runtime Error (运行时错误)

- 有可能是出现了野指针等问题
  - 这时可以一步步检查每条语句
- 方法一：在每行代码均打上断点：

```
test.c > main()
1  #include <stdio.h>
2  #include <math.h>
3  int main() {
4      int a,b,c;
5      double q,s;
6      scanf("%d%d%d",&a,&b,&c);
7      q=(a+b+c)/(double)2;
8      s = sqrt(q*(q-a)*(q-b)*(q-c));
9      printf("%f",s);
10     return 0;
11 }
```

程序停下来后按  或  即可运行到下一条代码。




含义是让程序继续运行至下一断点



含义是让程序运行至下一条语句

方法二：在程序开头打下断点

再按下  一步步运行程序。

```
C test.c > main()
1  #include <stdio.h>
2  #include <math.h>
3  int main() {
4      int a,b,c;
5      double q,s;
6      scanf("%d%d%d",&a,&b,&c);
7      q=(a+b+c)/(double)2;
8      s = sqrt(q*(q-a)*(q-b)*(q-c));
9      printf("%f",s);
10     return 0;
11 }
```

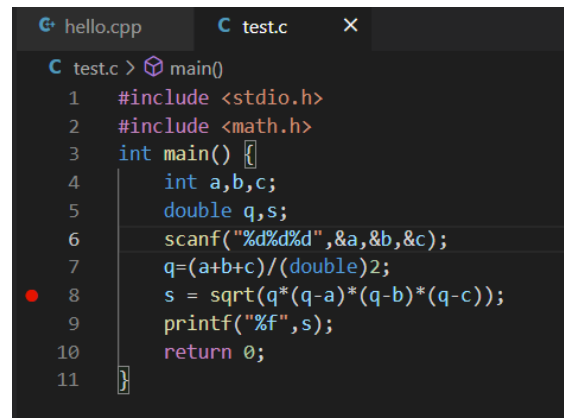
这样，程序会在某一条语句处崩溃，就知道哪里出现了错误。

# Time Limit Exceeded (超出时间限制)

- 是否有死循环
  - 是否陷入了无穷递归
  - 是否代码算法效率不够高
- 
- 前两种情况可以将断点打在循环或递归函数后，运行程序查看是否会停在断点处。
  - 后一种情况可以改进程序算法。

# Wrong Answer (结果错误)

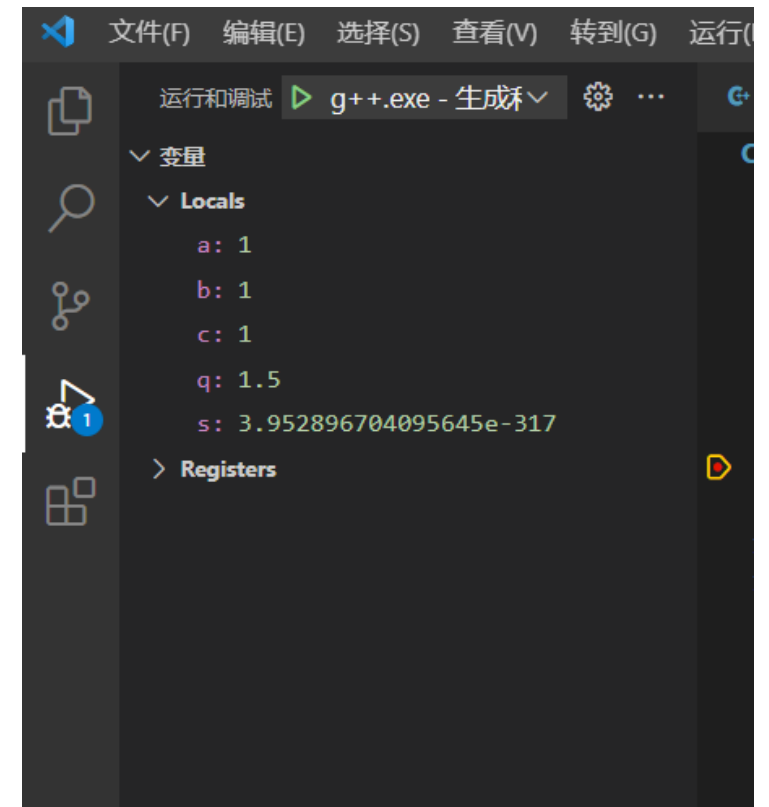
一般是代码实现有问题，可以在关键变量处打下断点，查看关键变量中的值是否符合我们的预期。比如三角形求面积，可以在给面积赋值处打下断点：



```
hello.cpp  test.c  x
C test.c > main()
1  #include <stdio.h>
2  #include <math.h>
3  int main() {
4      int a,b,c;
5      double q,s;
6      scanf("%d%d%d",&a,&b,&c);
7      q=(a+b+c)/(double)2;
8      s = sqrt(q*(q-a)*(q-b)*(q-c));
9      printf("%f",s);
10     return 0;
11 }
```

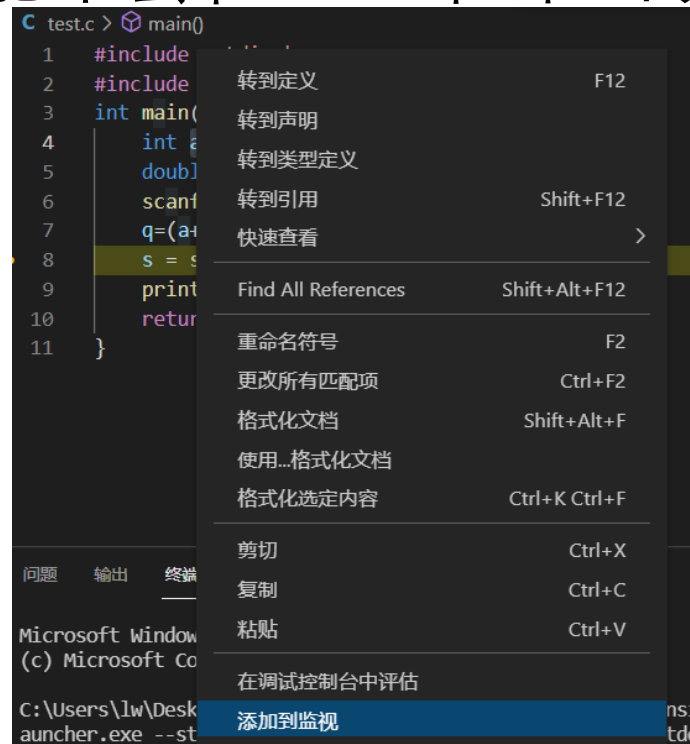
程序运行到此后查看左边调试栏中变量的值：

通过这些值，我们可以准确定位到错误地点。





- 有时候变量可能不会在Local栏中出现，我们可以选中变量后，点击添加监视：



- 就可以在变量下的监视栏中看到我们想看到的变量了：

