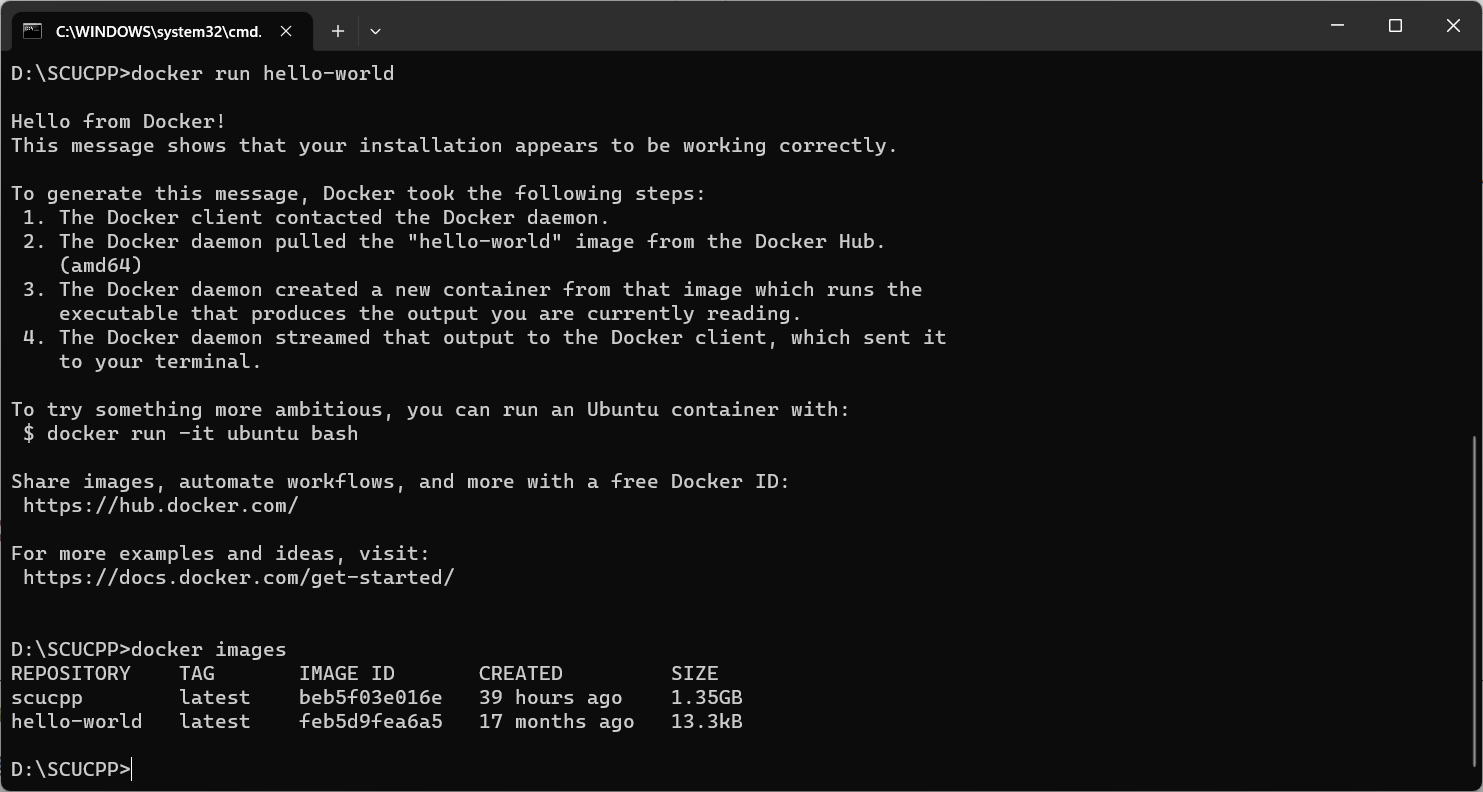
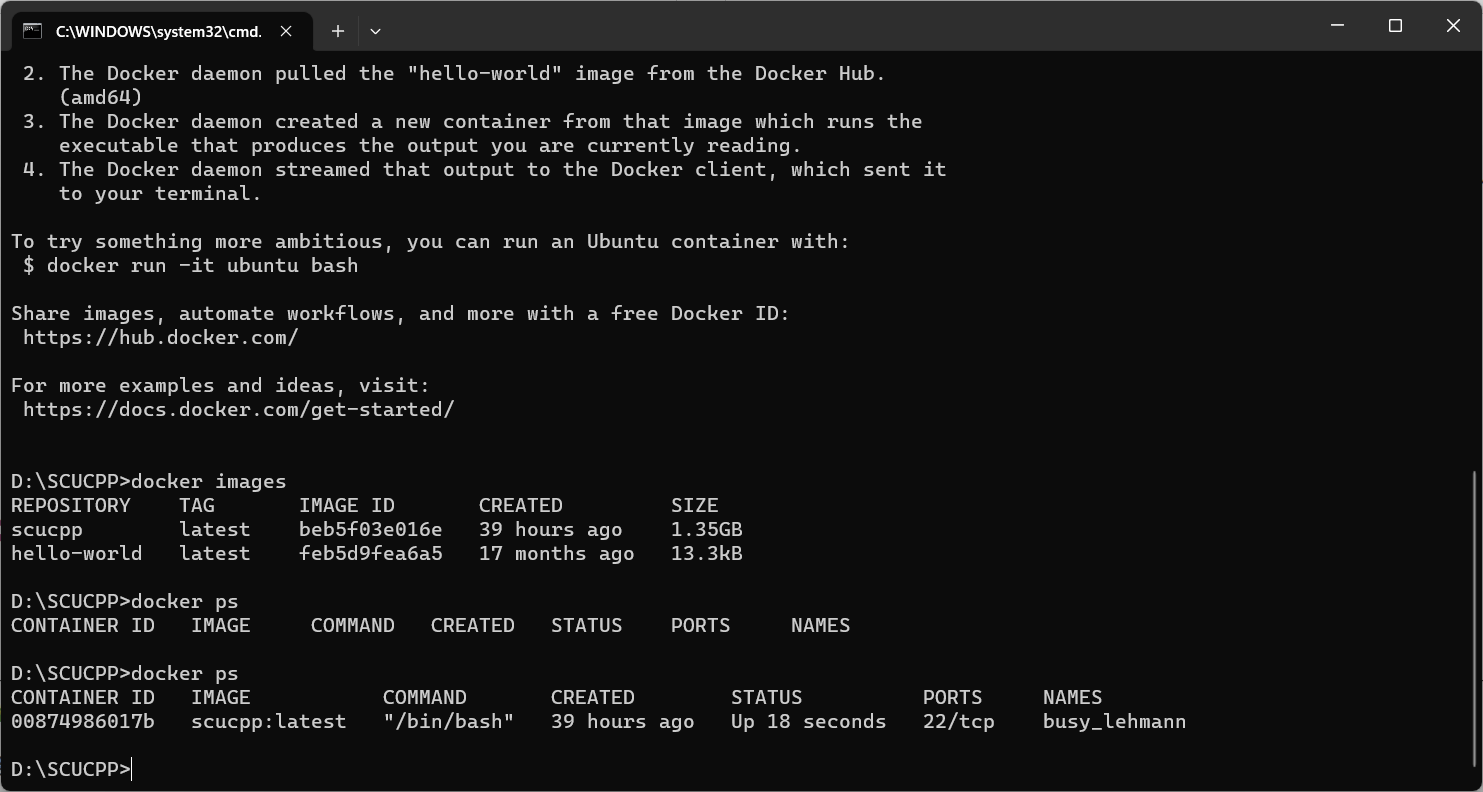
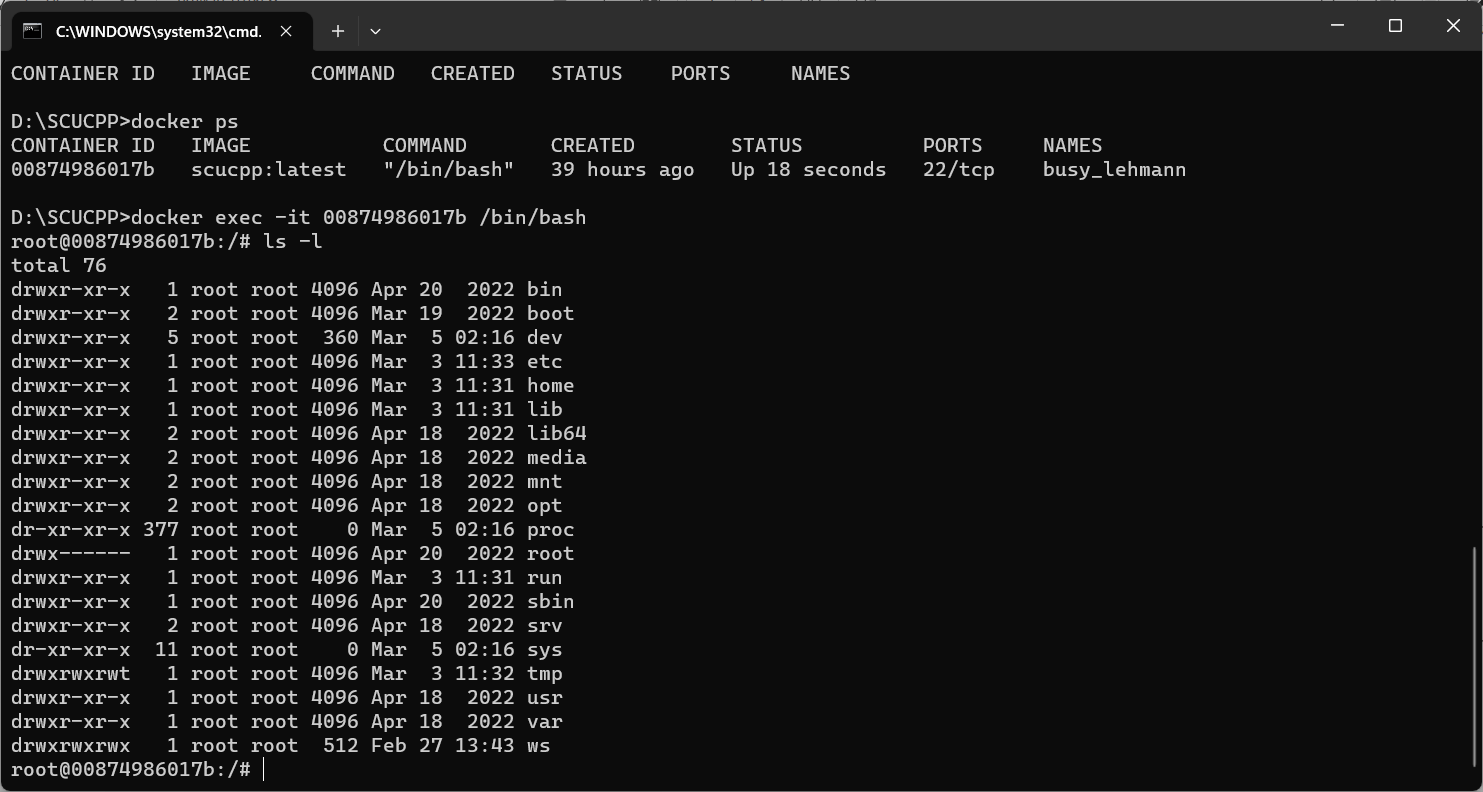
2022141450162 朱骥锋 assignment1

1. **配置对应的dockers环境**

****

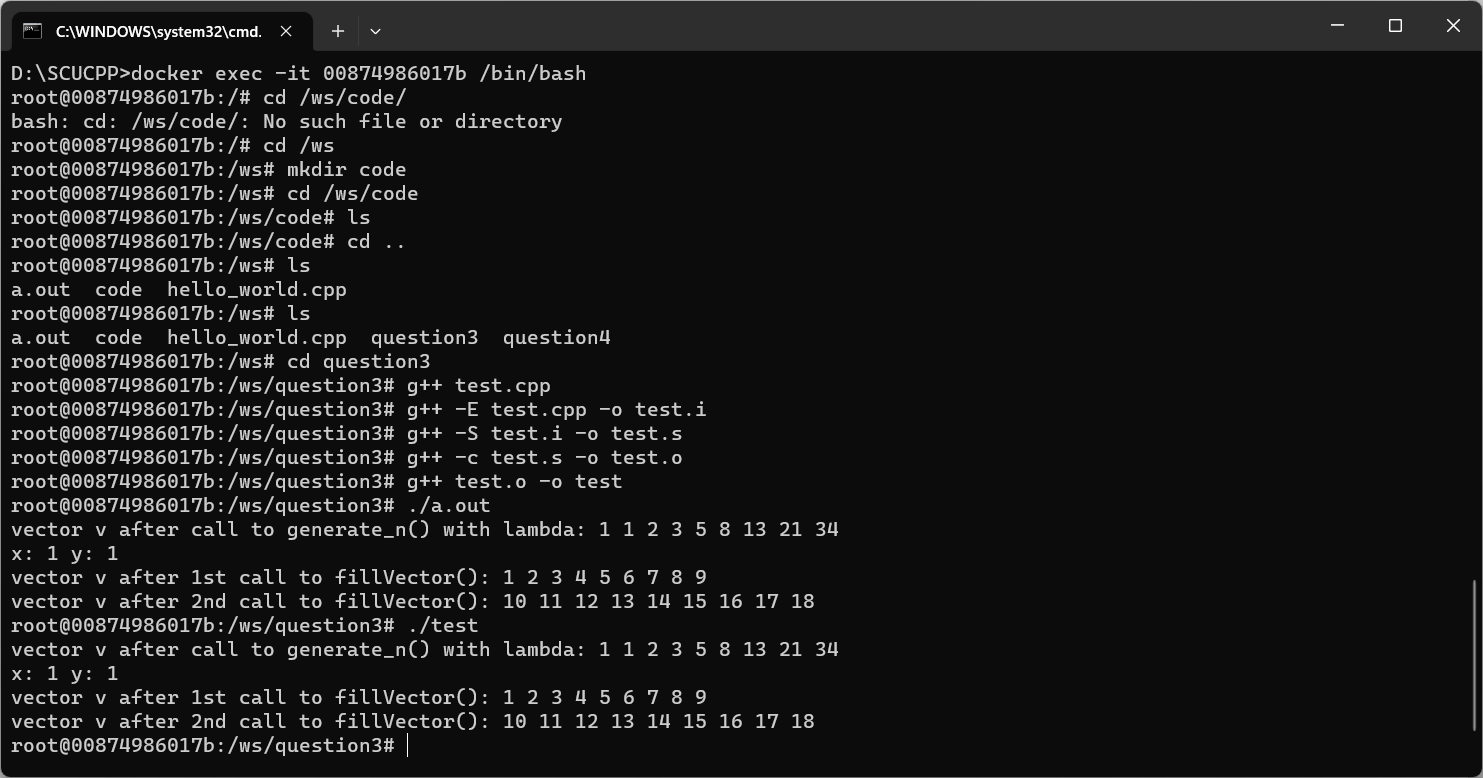


执行 docker run hello-world,docker images,docker ps 的运行截图



使用 docker exec -it /bin/bash 进入 docker 后 cd /ws/code/的截图

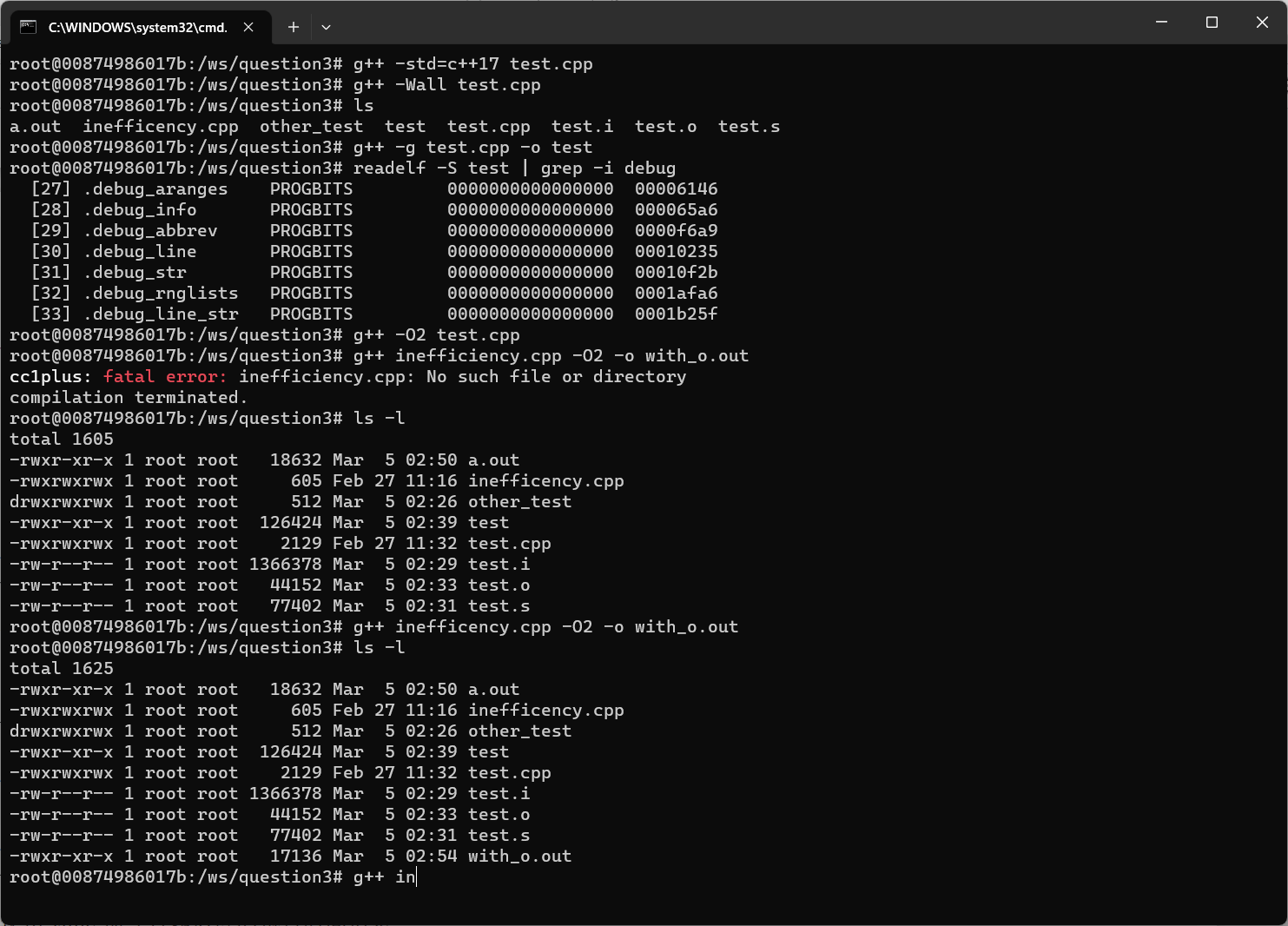
1. **利用g++编译单文件**

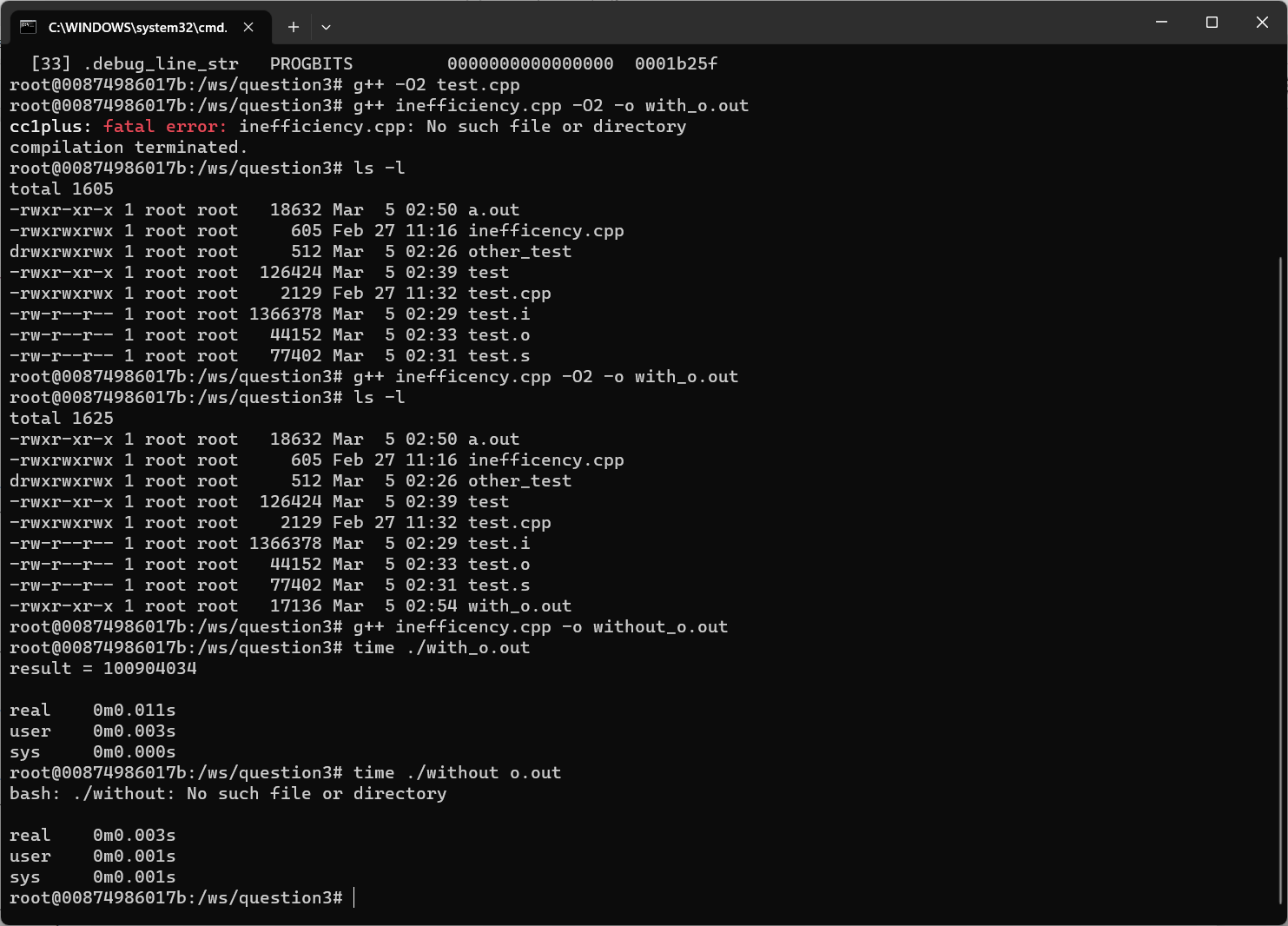
****

对test.cpp文件直接编译得到可执行的test文件与分步编译得到可执行的a.out文件

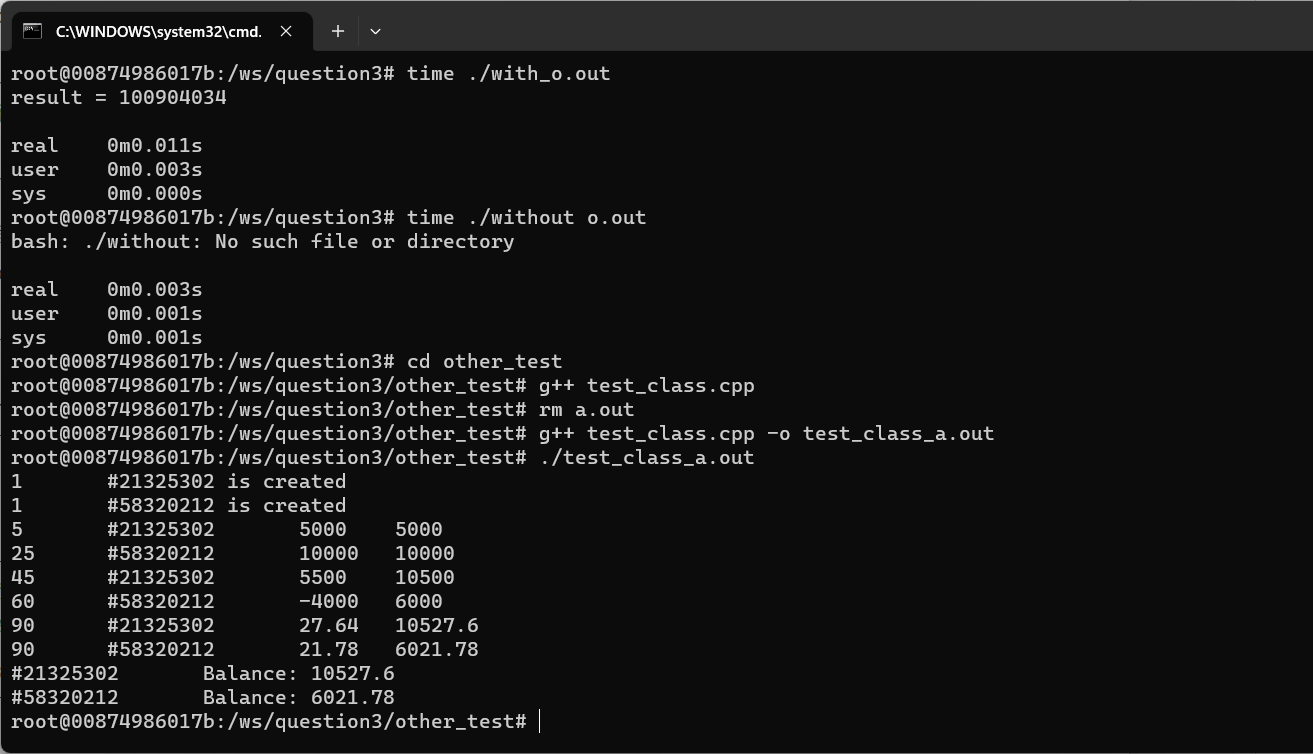
用ls命令列出/ws目录中所有文件

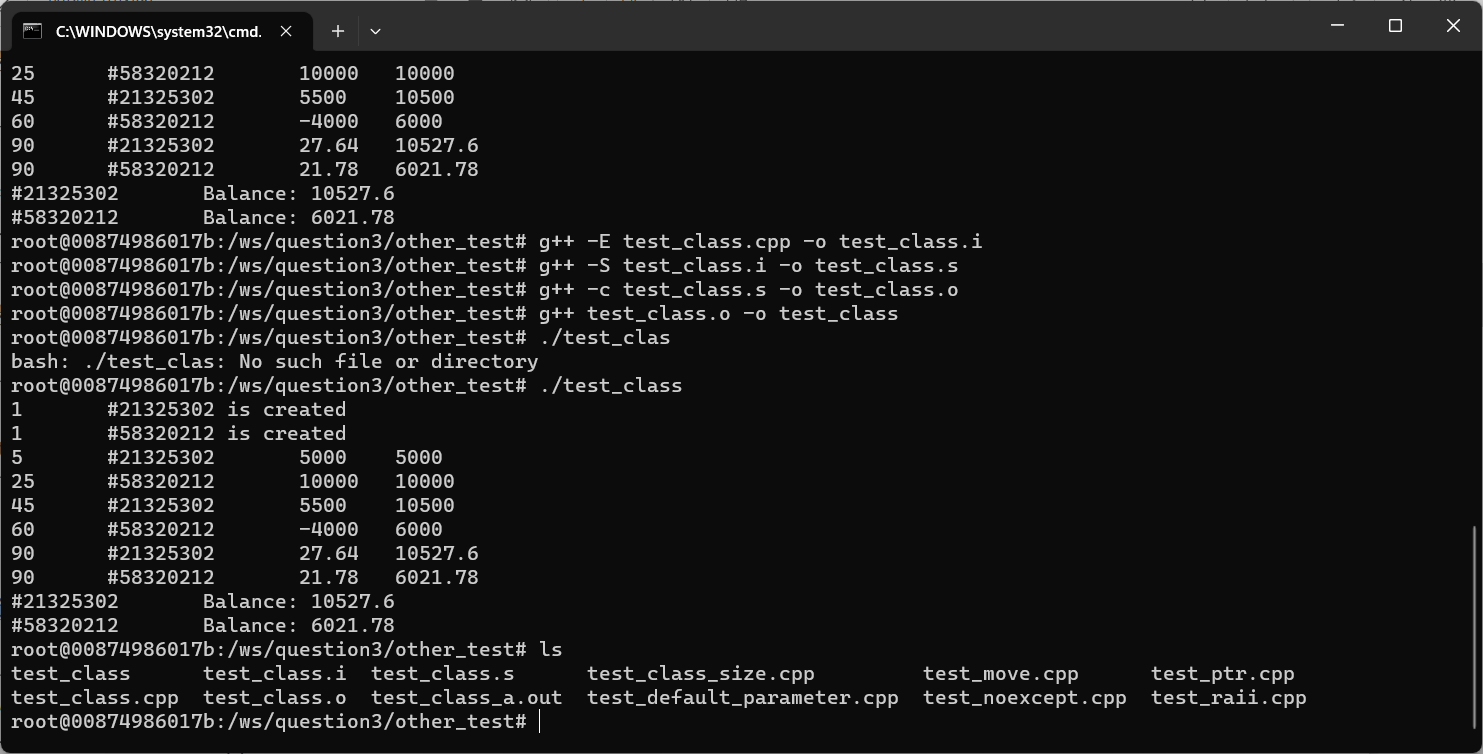
并获得相应运行截图

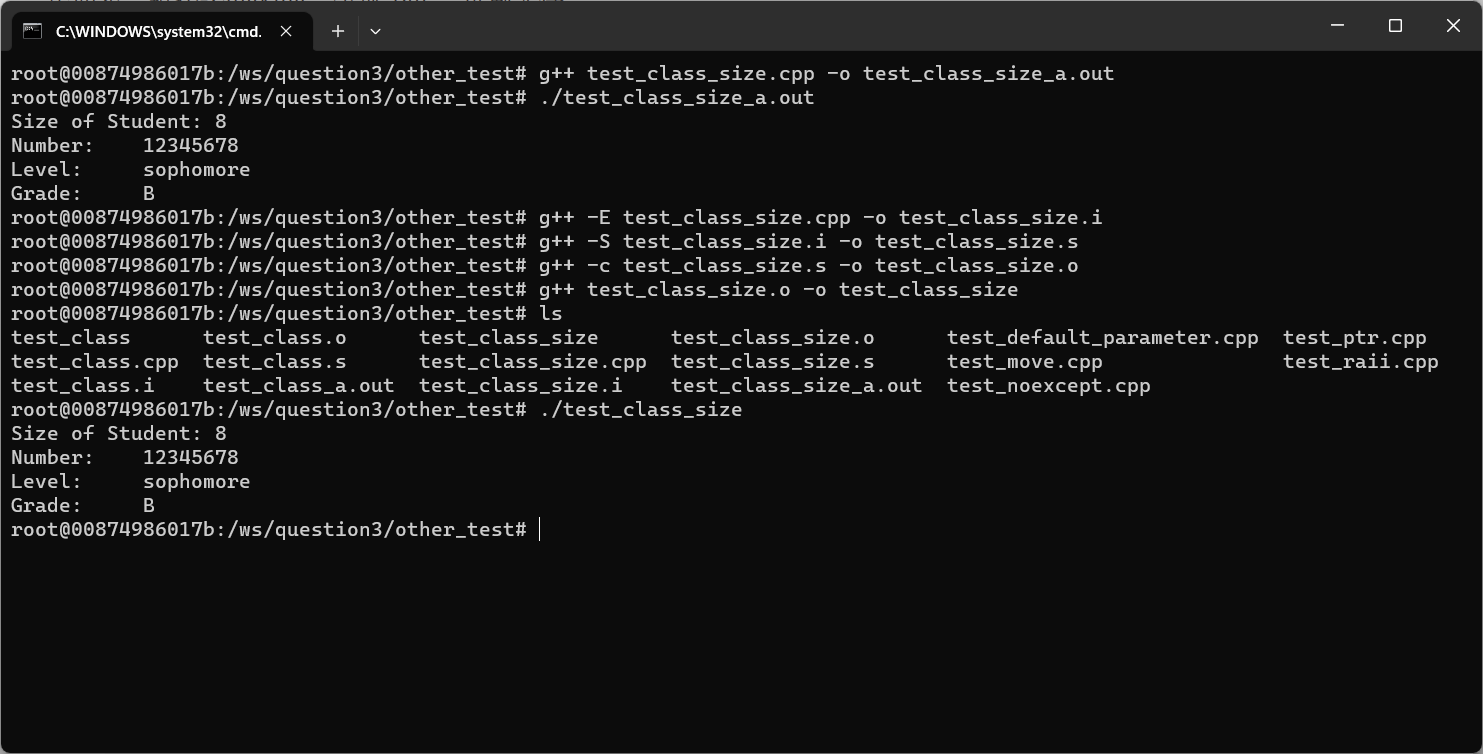


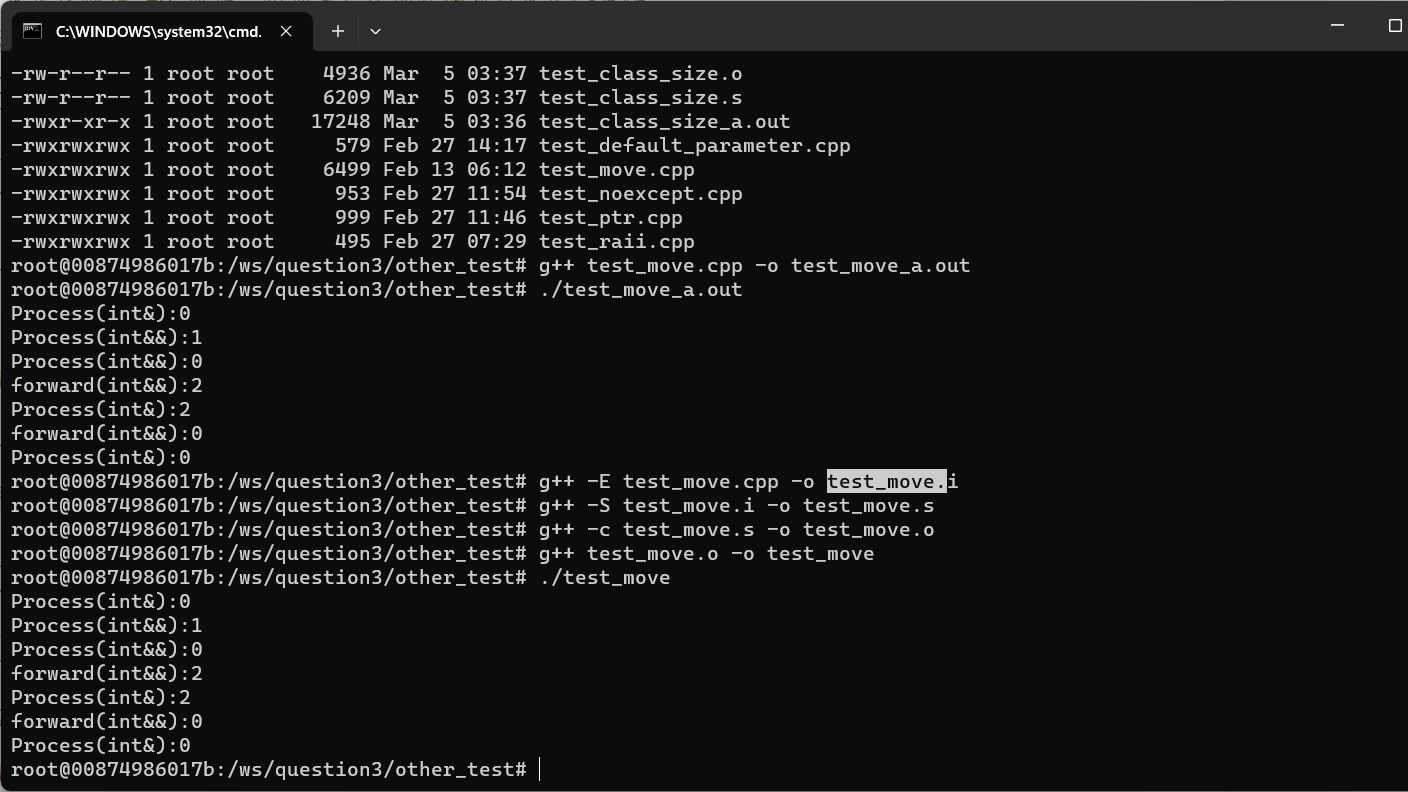


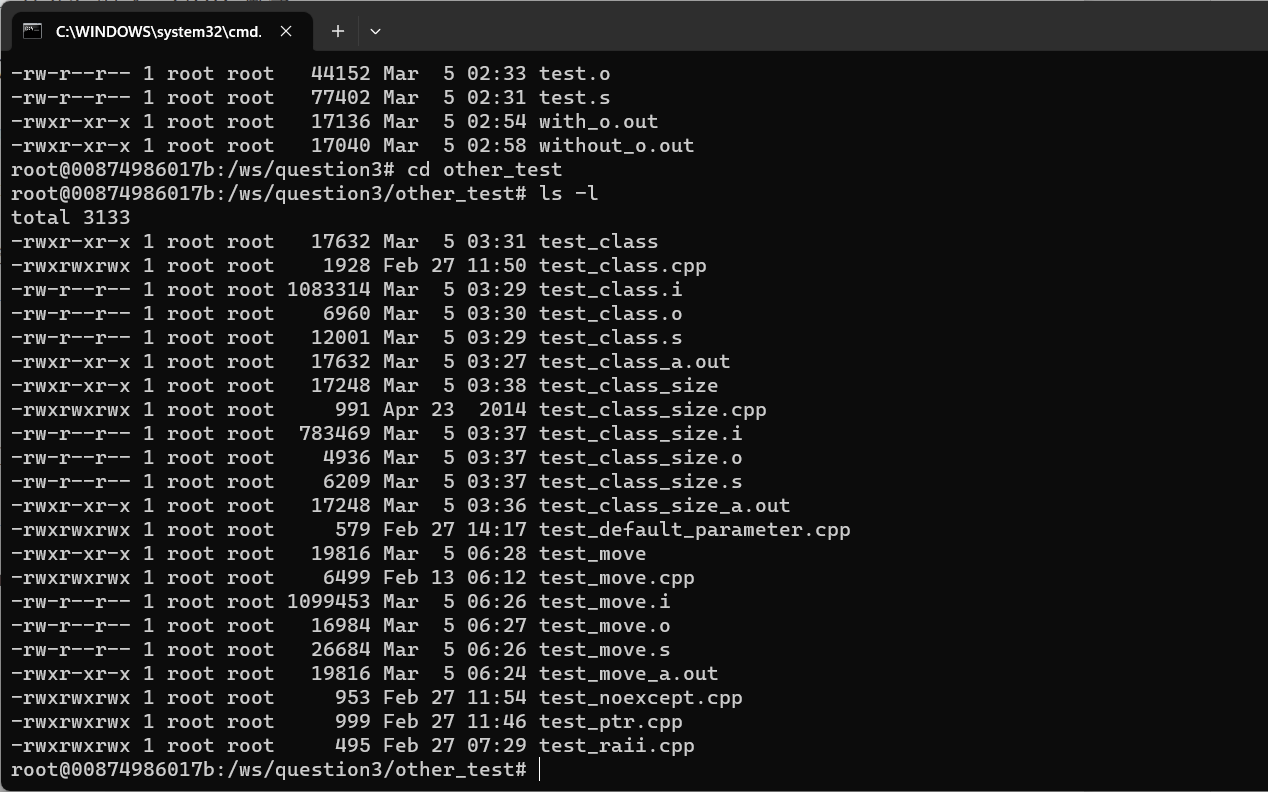
inefficency.cpp 优化编译与非优化编译的执行信息





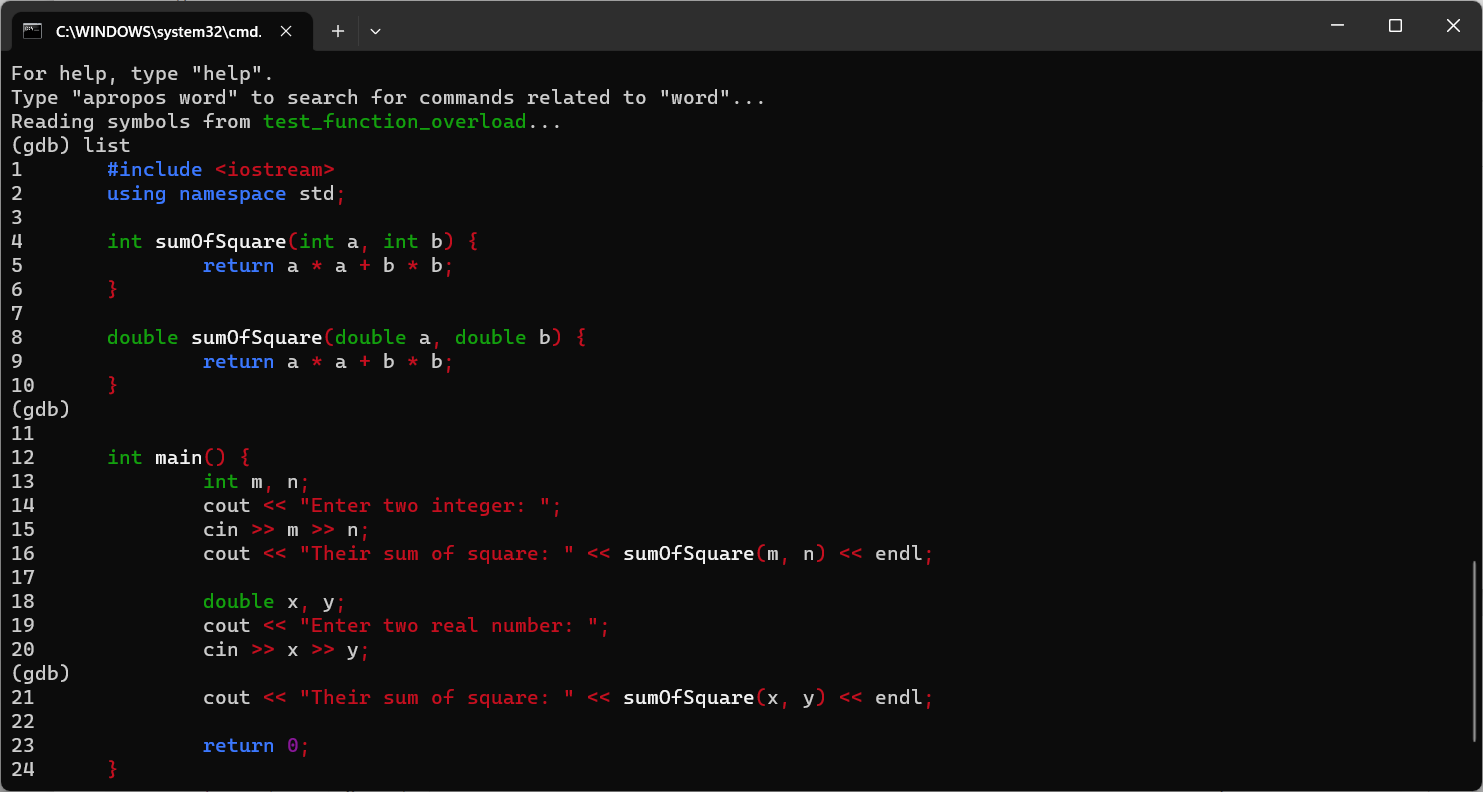


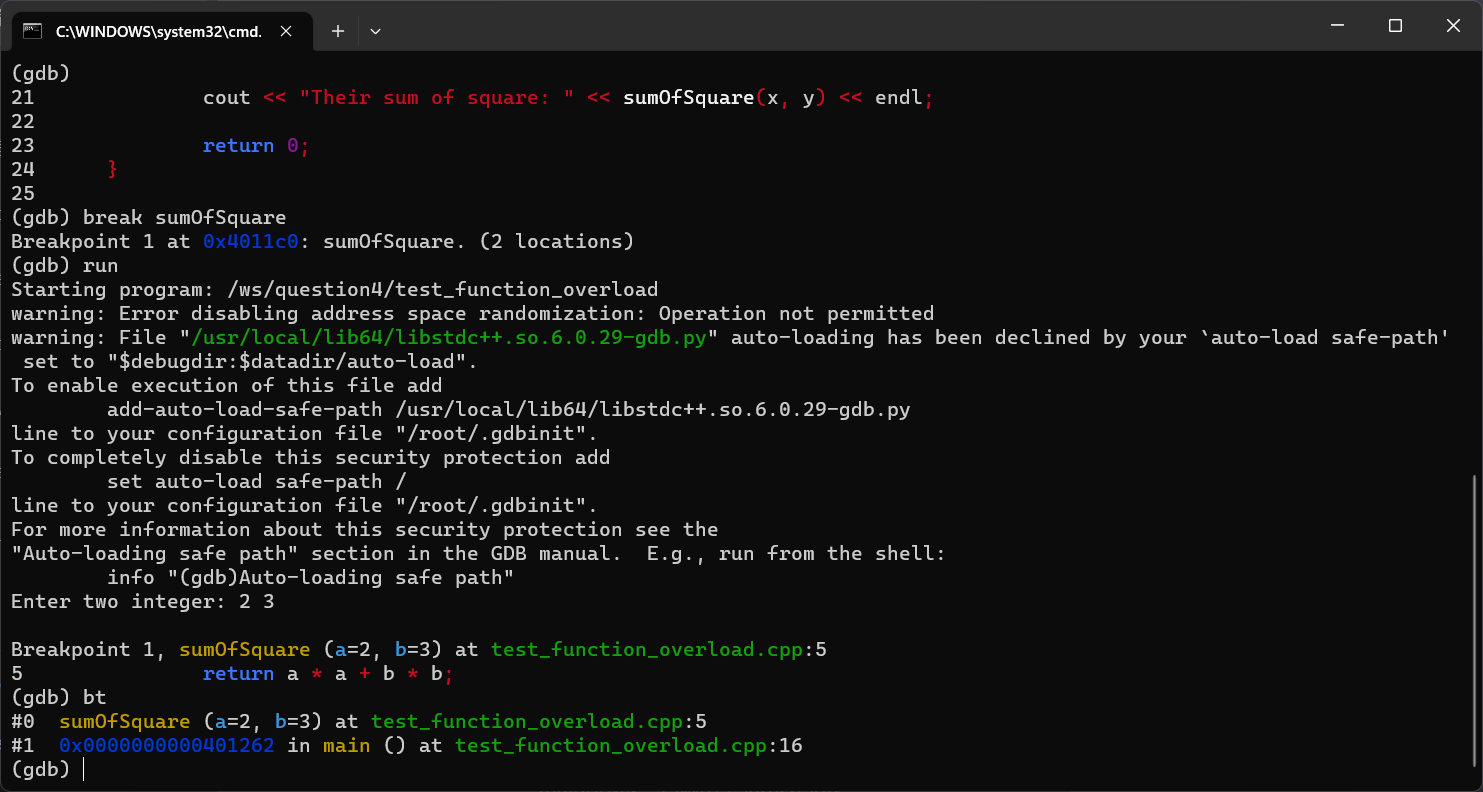


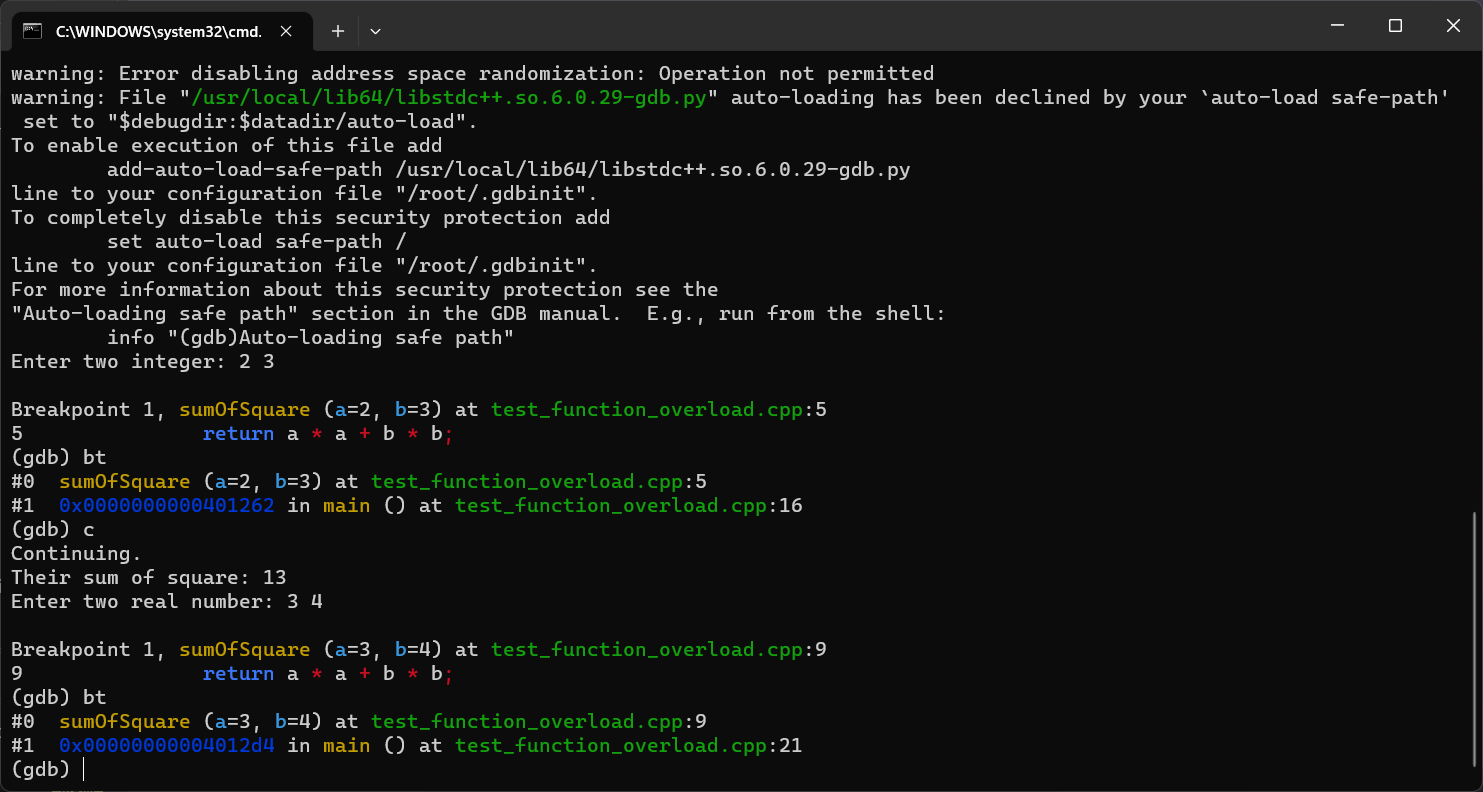


对test\_class.cpp，test\_class\_size.cpp，test\_move.cpp文件利用不同的 g++ 参数进行编 译对应生成文件与最终的执行截图

1. **利用gdb调试文件**

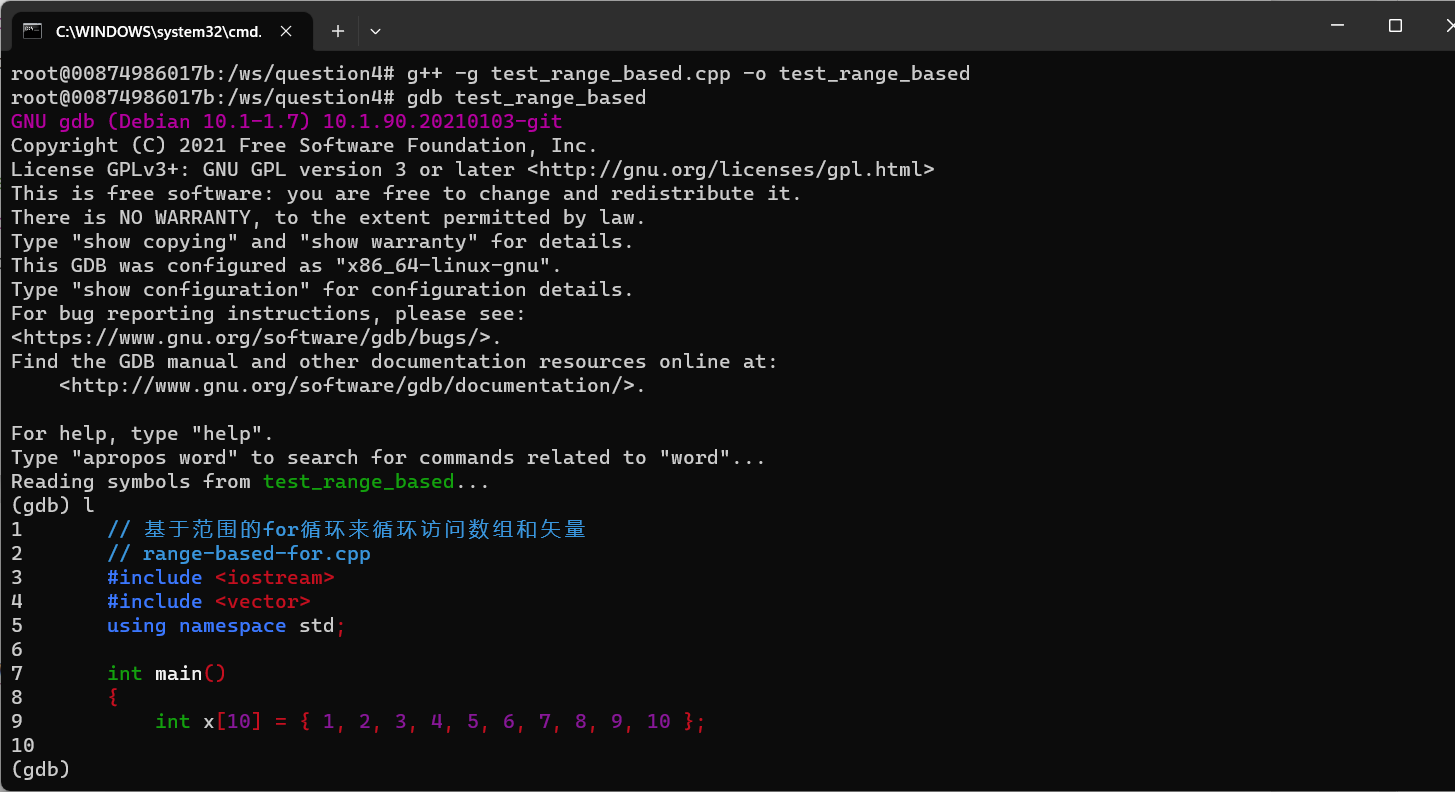
****

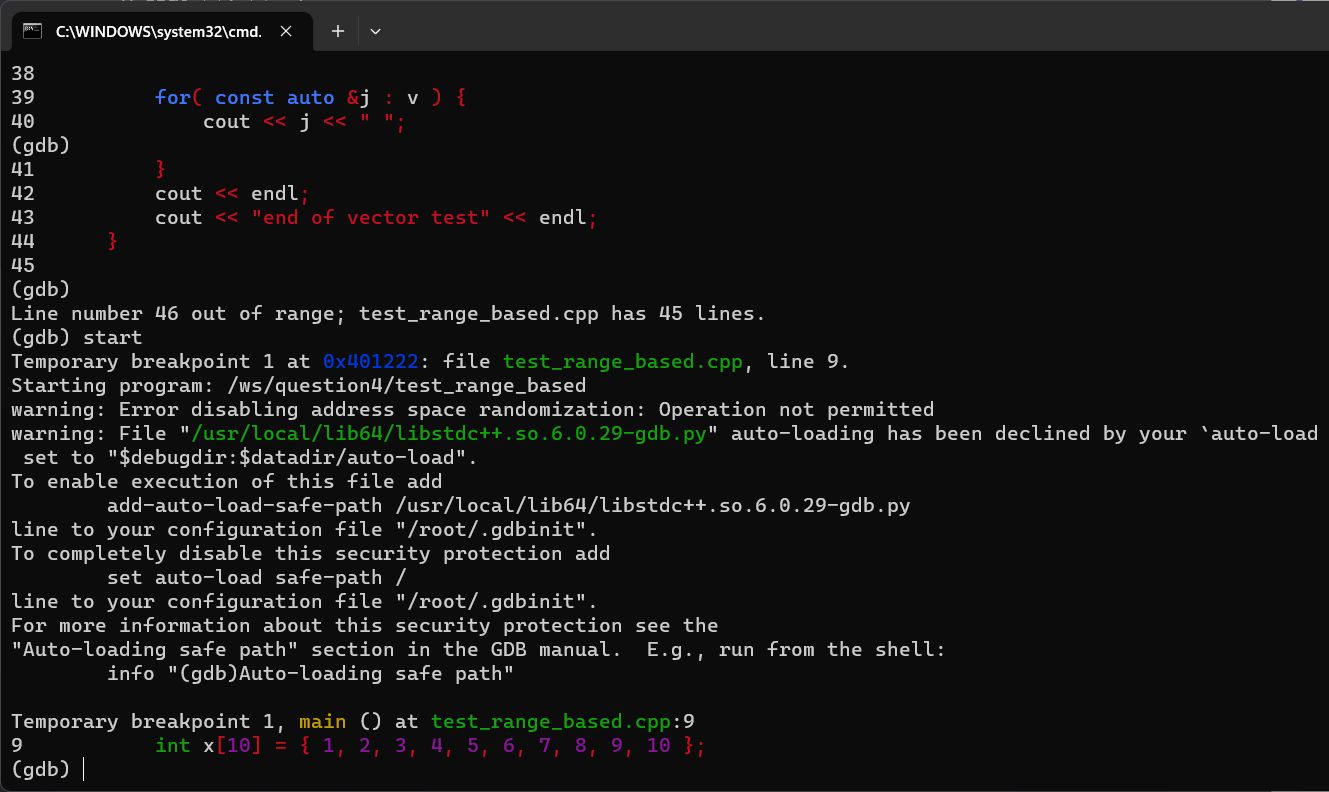
****

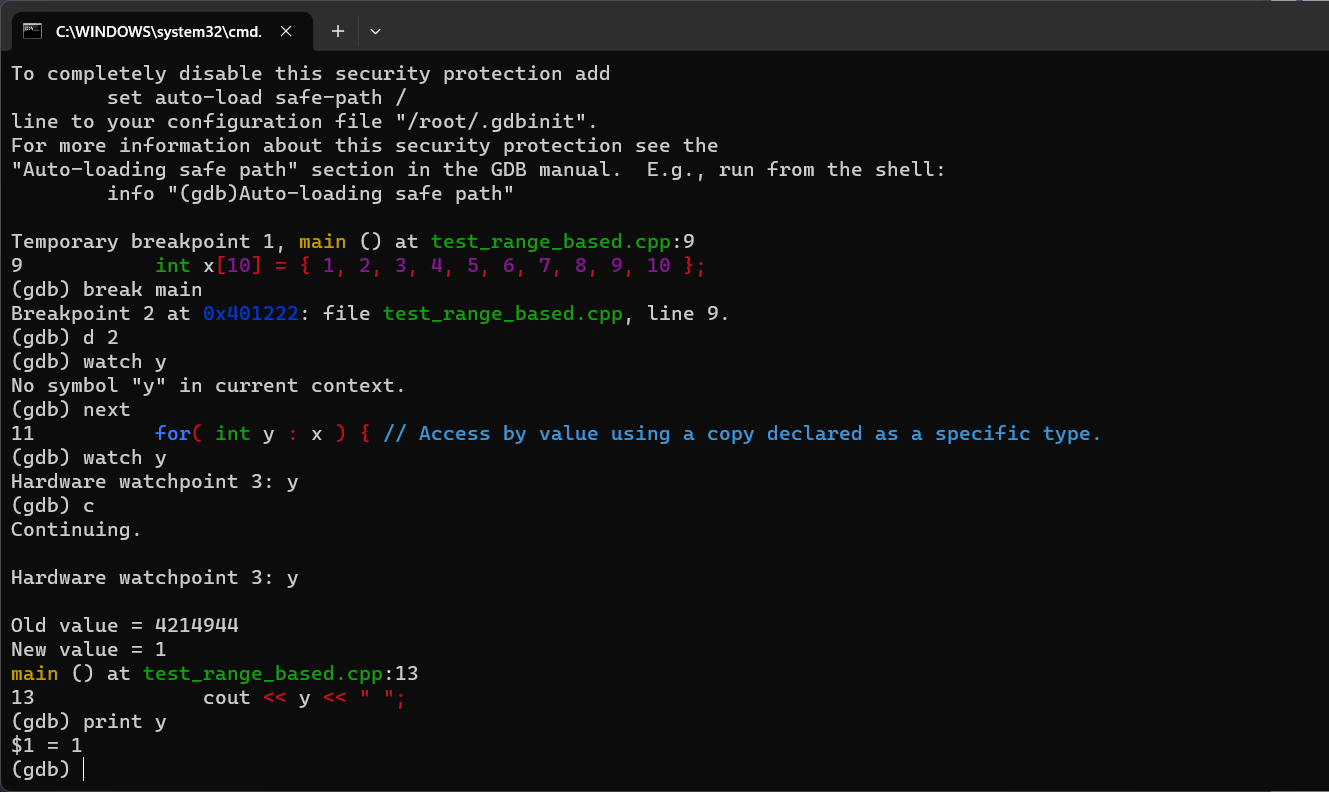


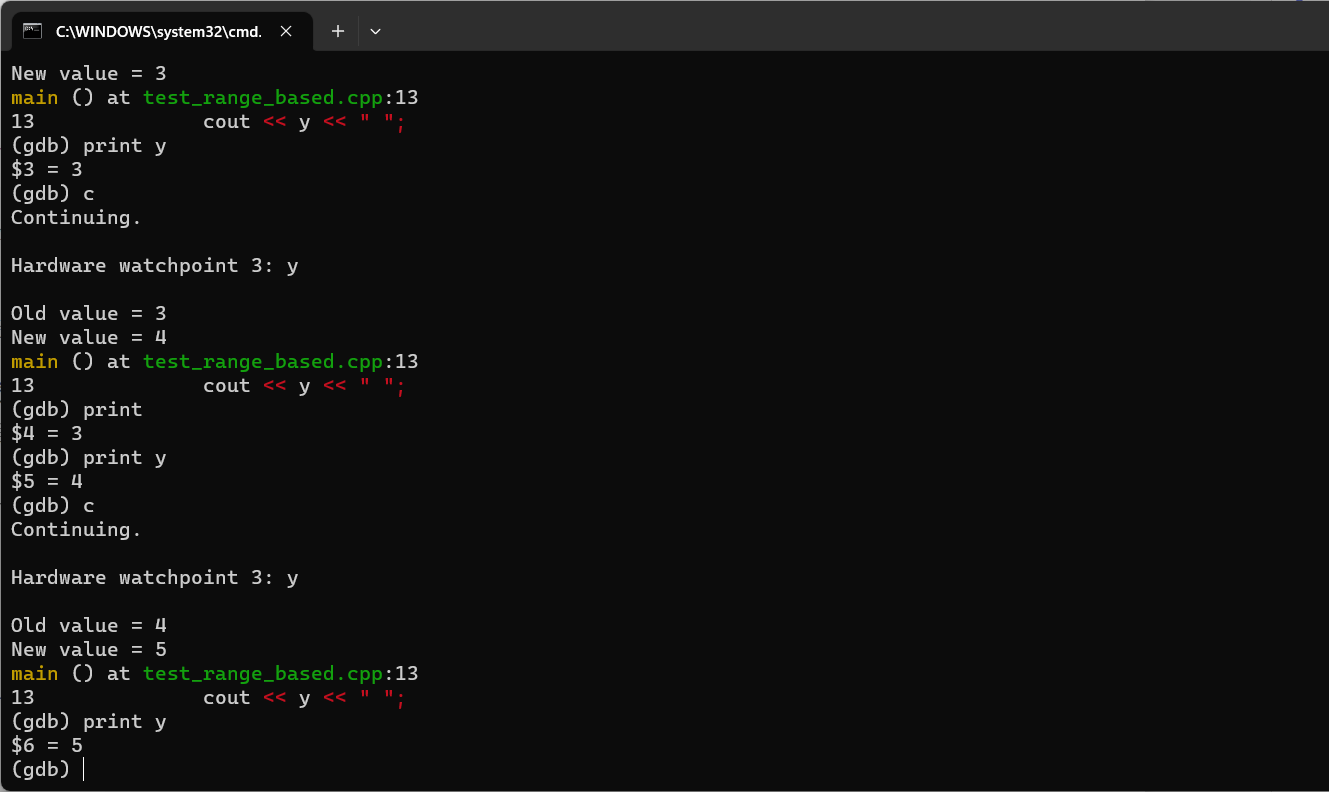
分别追踪前后两次调用函数 sumOfSquare() 时函数调用的栈帧与层级关系，利用backtrace指令

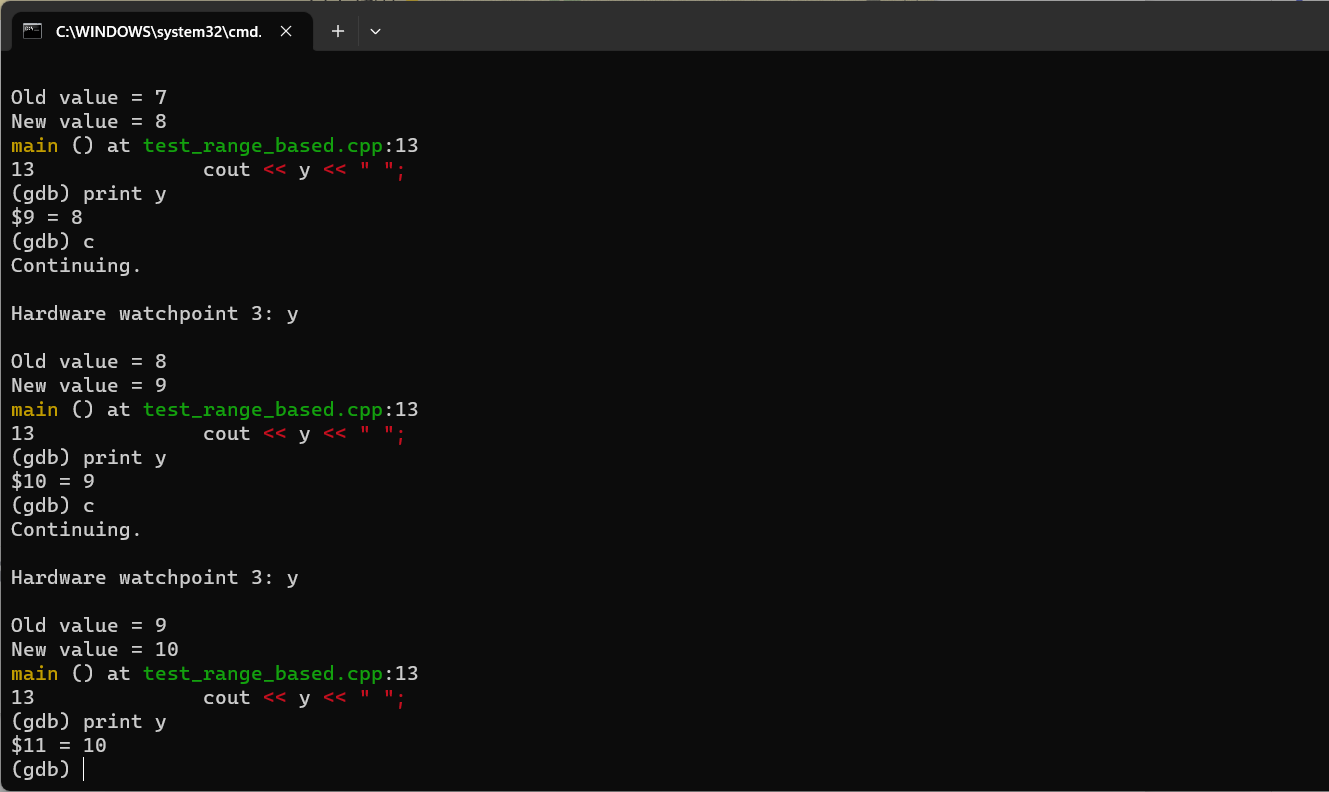
追踪每次循环的变量 y 的具体变量值：



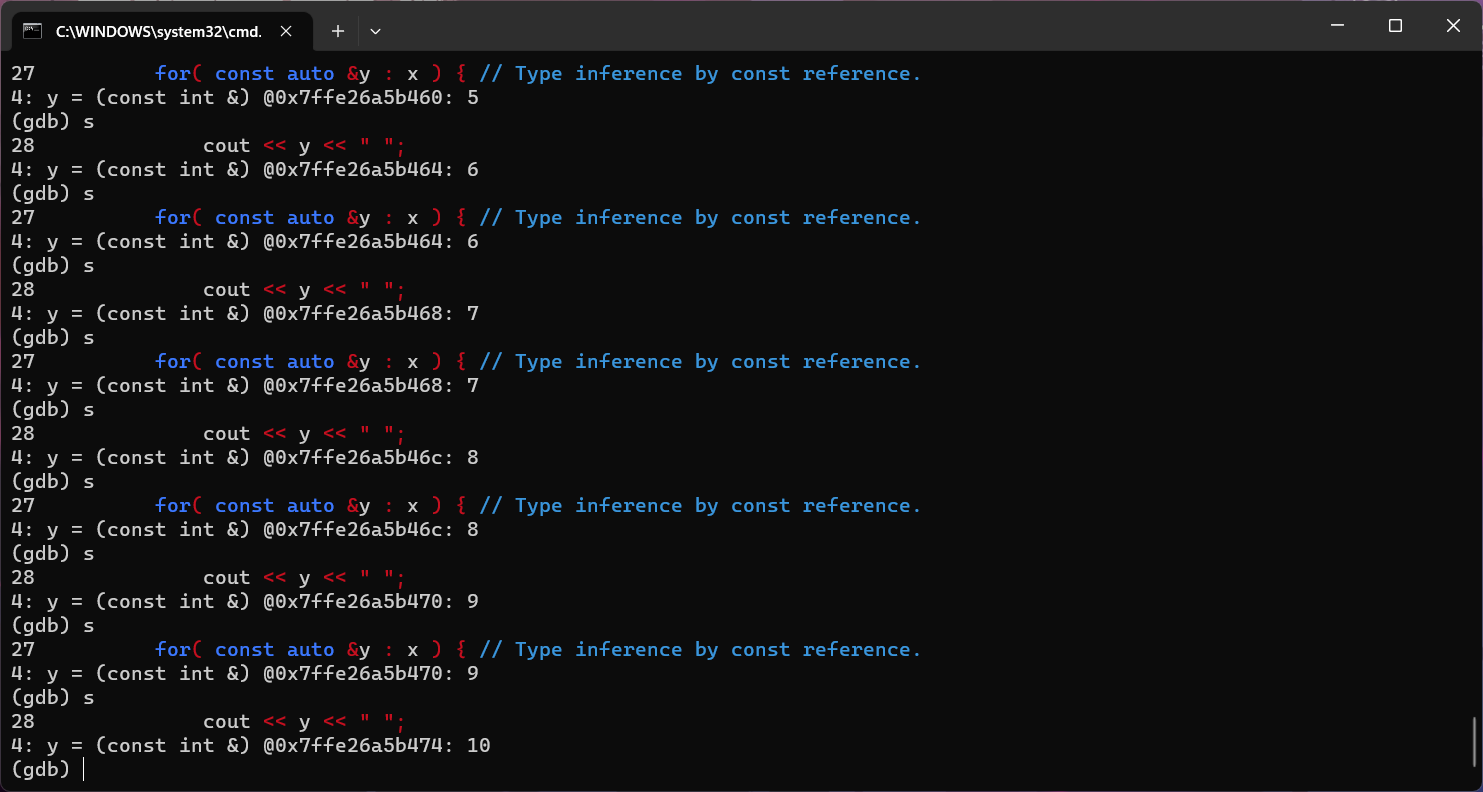
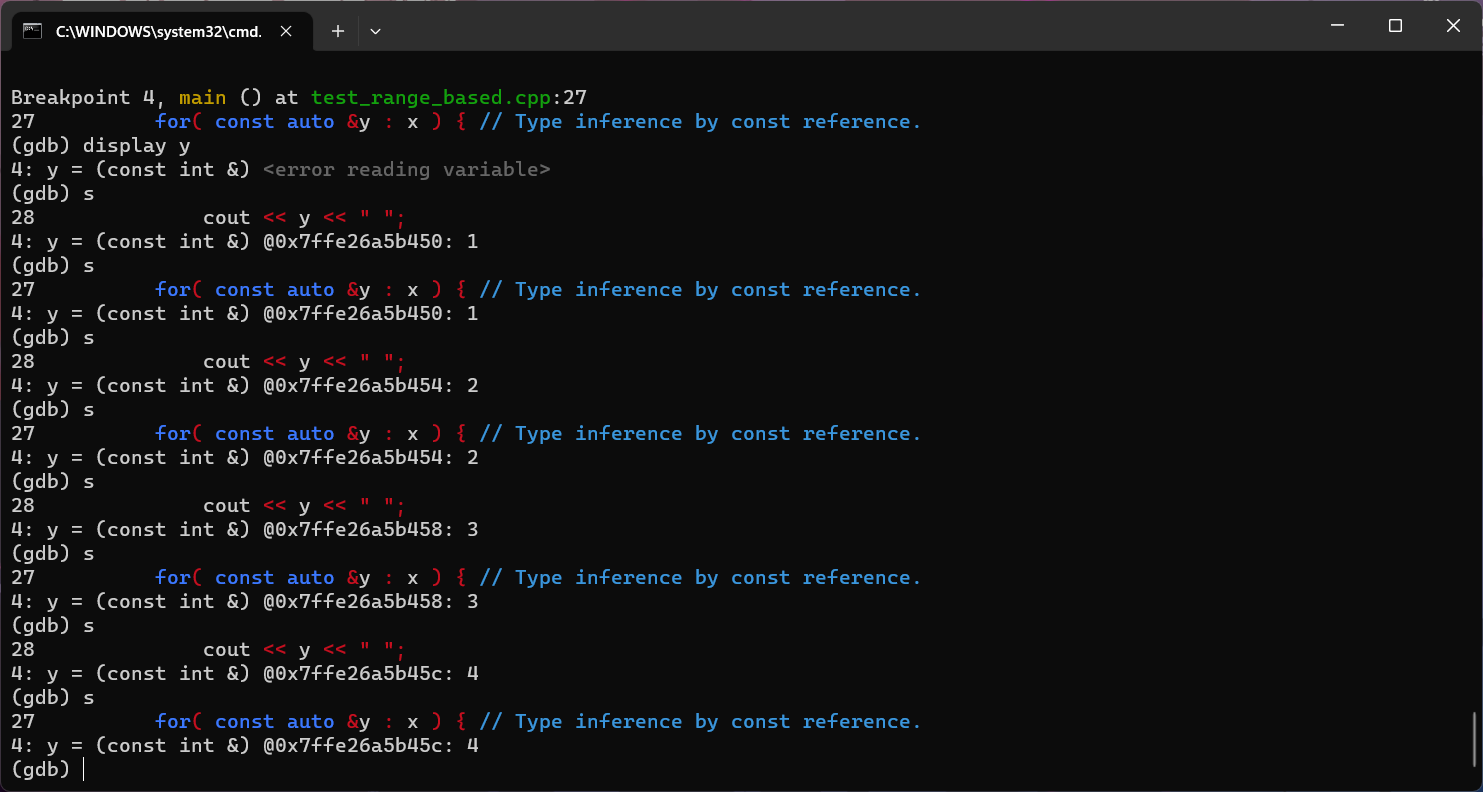
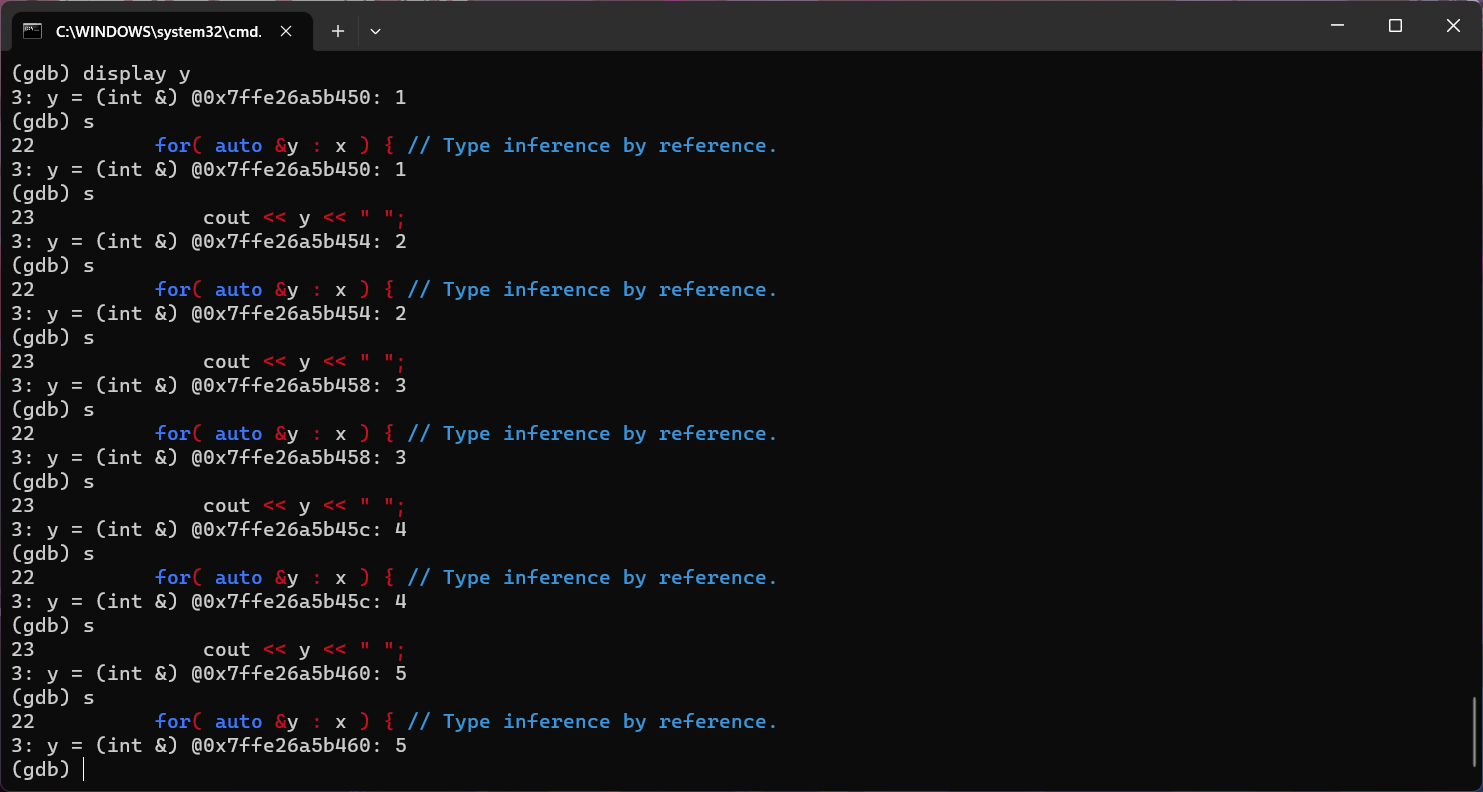
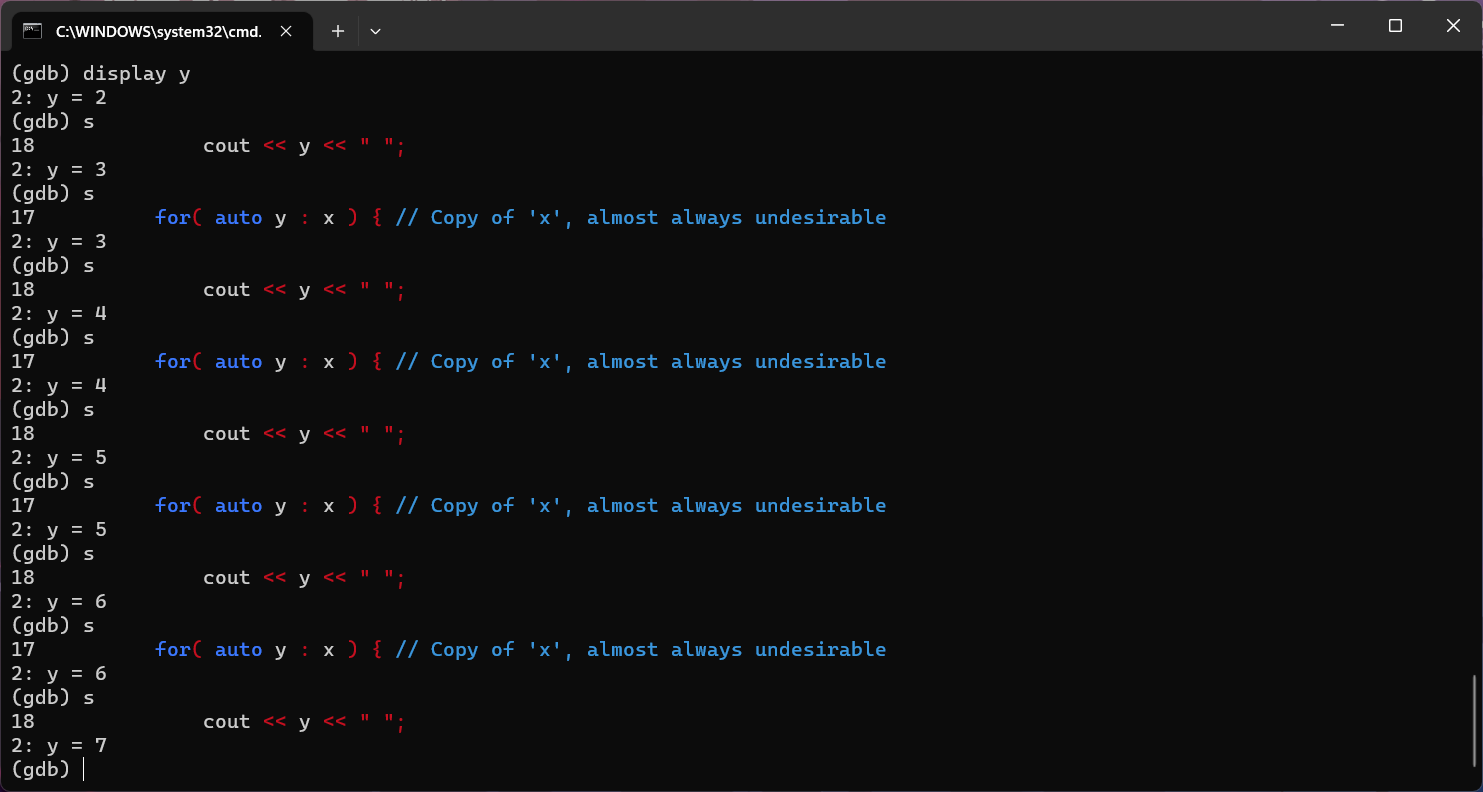
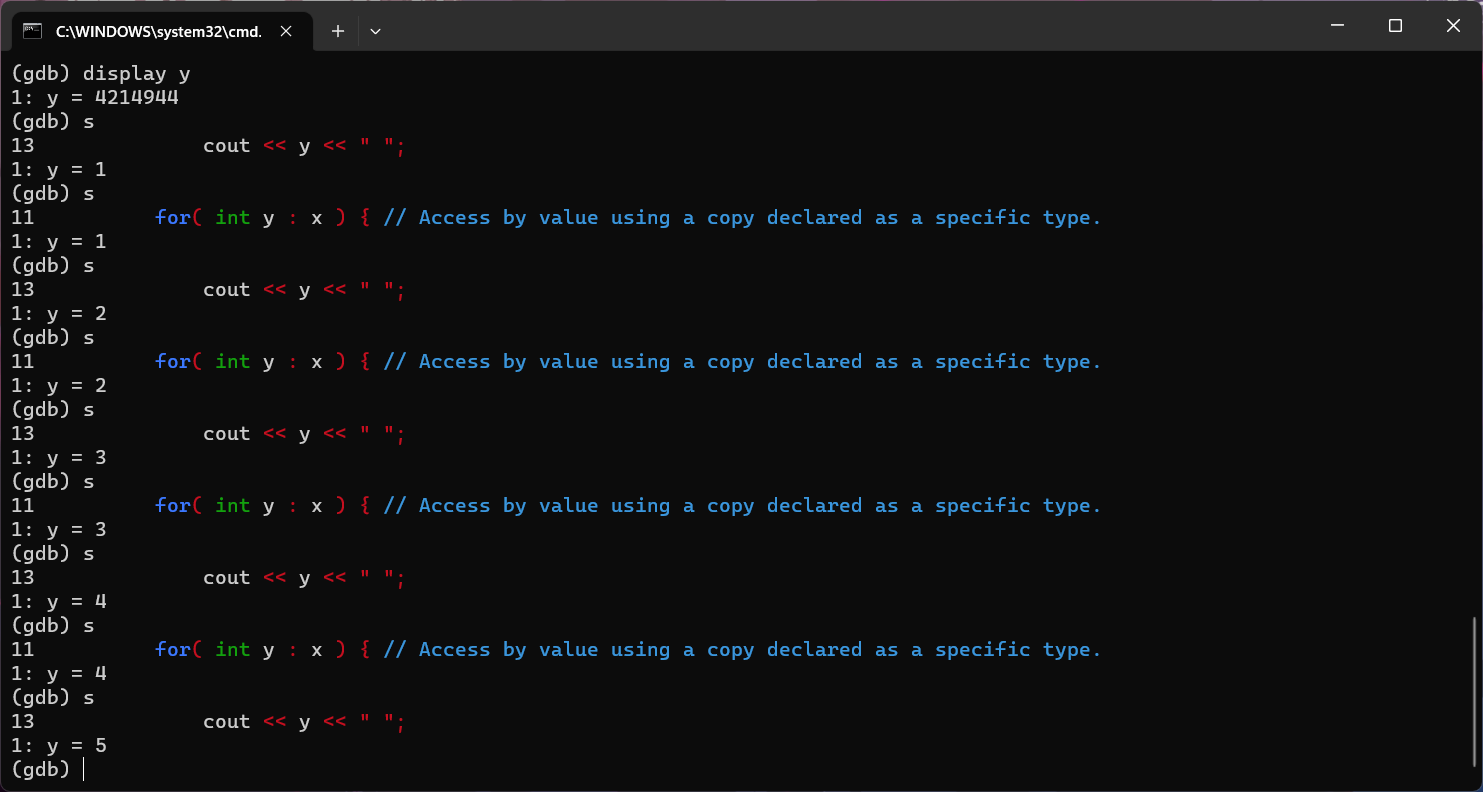




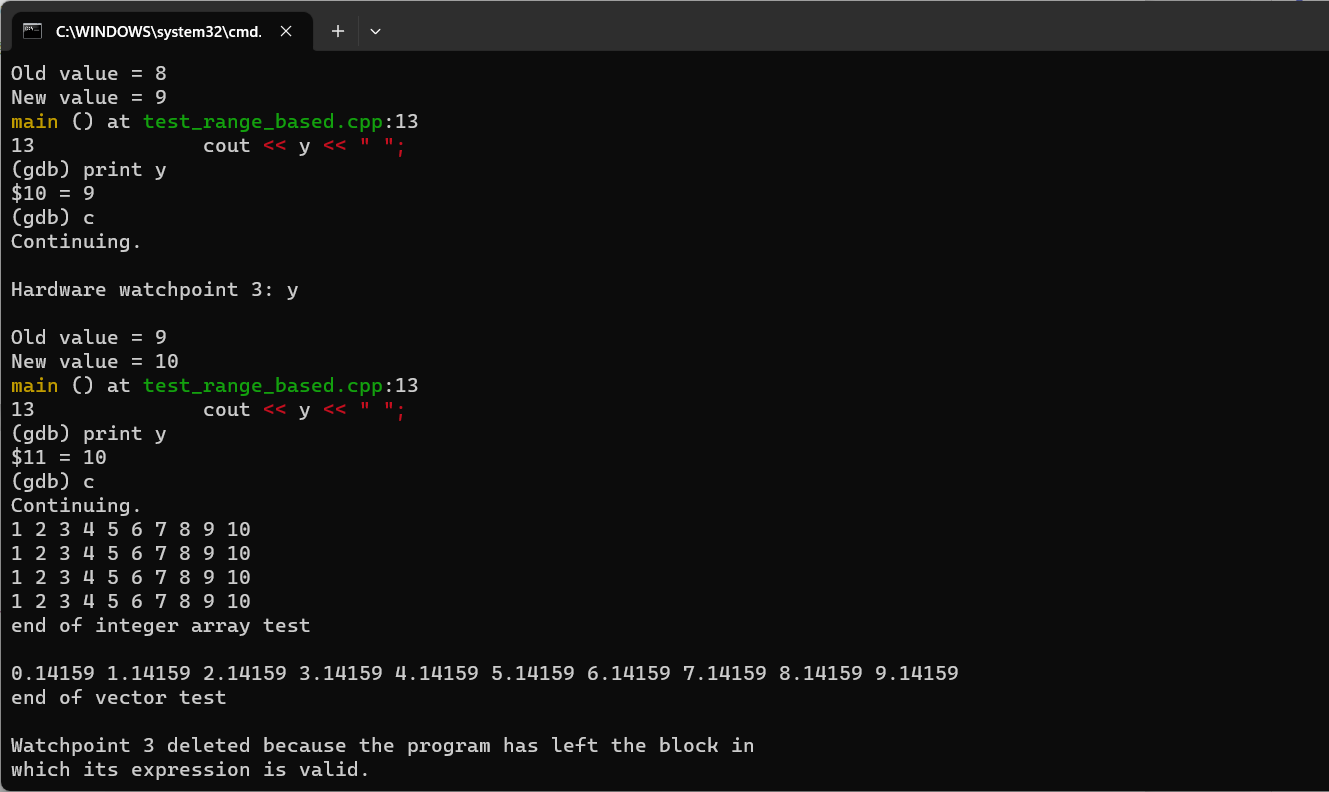




*方法一，设置y为观察点，追踪y变化*



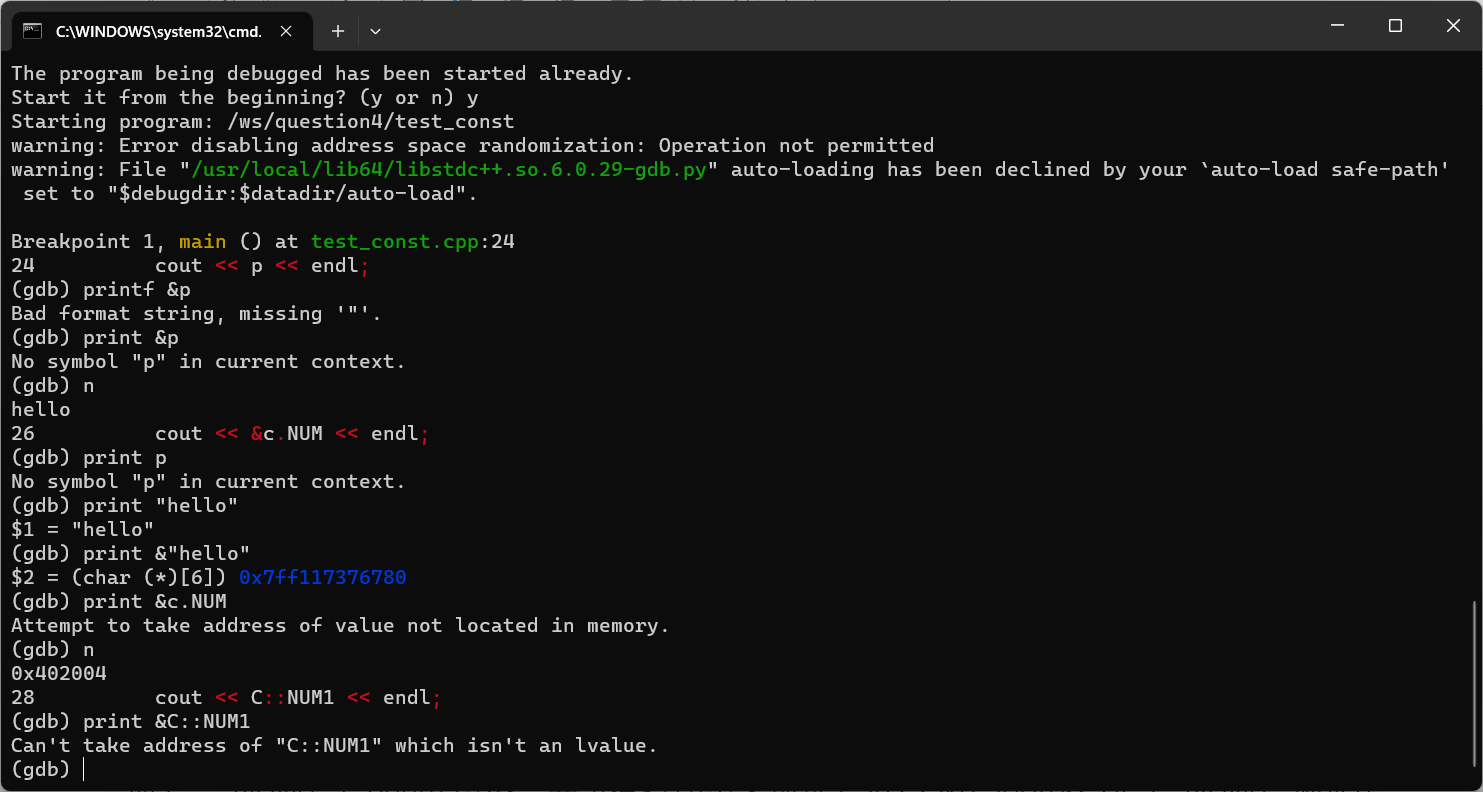
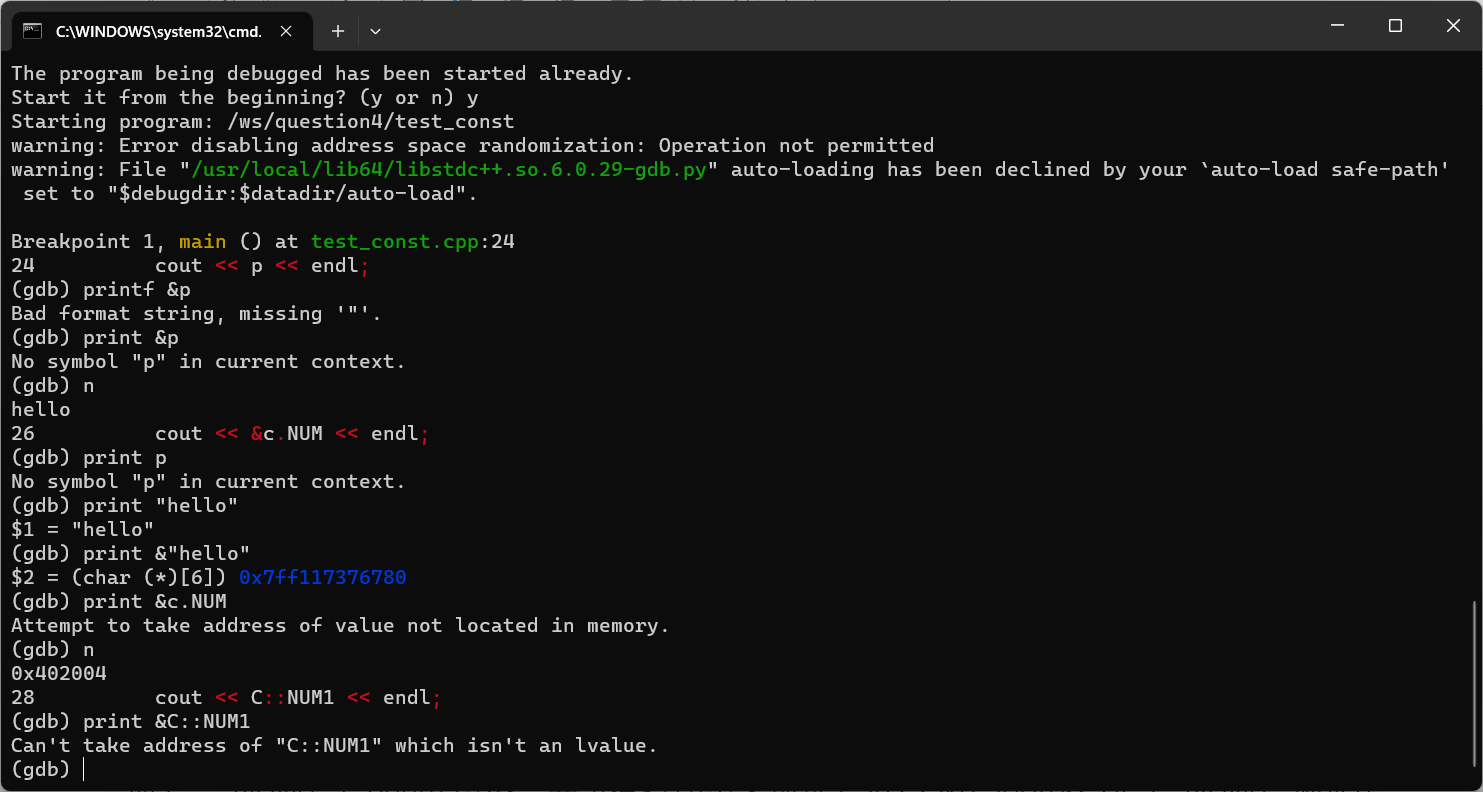
*方法二：设置断点监视y的变化，利用display命令*



打印出来j的值：vector v中所存的数被各个赋值到 j 上，由于类型是const 型，所以是只读型。

对于代码片段三，我们希望您能根据调试与代码中的提示修改代码让其正常运行，并告诉我们 const,enum，define 三者中哪一个有地址，并将其地址打印出来：

结论：enum没有地址，const和define有地址

define p为字符串“hello”后打印“hello”地址，可以看到能够输出地址，所以define是有地址的。

对于enum型，可以看到无法取地址。

在打印c.NUM的地址时可以看到显示其attemp to take the address of value not located in the memory，即获取不在内存中的值的地址。因为NUM变量仅仅实在类中被声明，并没有给其分配内存空间，因此需要在外部用const int C::NUM定义，即静态成员变量是属于类而不是属于对象的，故这么取是错误的，应改为&C::NUM。

下面是更正后的代码截图：

