parts of phase 4 code:

Canary → checks for stack overflow

Checks exactly 2 inputs

```
Breakpoint 2, 0x00005555555557af in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
=> 0x00005555555557af <+0>:     endbr64
   0x00005555555557b3 <+4>:     sub    $0x18,%rsp
   0x00005555555557b7 <+8>:     mov    %fs:0x28,%rax
   0x00005555555557c0 <+17>:    mov    %rax,0x8(%rsp)
   0x00005555555557c5 <+22>:    xor    %eax,%eax
   0x00005555555557c7 <+24>:    mov    %rsp,%rcx
   0x00005555555557ca <+27>:    lea    0x4(%rsp),%rdx
   0x00005555555557cf <+32>:    lea    0x1c2f(%rip),%rsi        # 0x555555557405
   0x00005555555557d6 <+39>:    call   0x555555555330 <__isoc99_sscanf@plt>
   0x00005555555557db <+44>:    cmp    $0x2,%eax
   0x00005555555557de <+47>:    jne    0x5555555557eb <phase_4+60>
   0x00005555555557e0 <+49>:    mov    (%rsp),%eax
   0x00005555555557e3 <+52>:    sub    $0x2,%eax
   0x00005555555557e6 <+55>:    cmp    $0x2,%eax
   0x00005555555557e9 <+58>:    jbe    0x5555555557f0 <phase_4+65>
   0x00005555555557eb <+60>:    call   0x555555555e0f <explode_bomb>
   0x00005555555557f0 <+65>:    mov    (%rsp),%esi
   0x00005555555557f3 <+68>:    mov    $0x9,%edi
   0x00005555555557f8 <+73>:    call   0x555555555774 <func4>
   0x00005555555557fd <+78>:    cmp    %eax,0x4(%rsp)
   0x0000555555555801 <+82>:    jne    0x555555555818 <phase_4+105>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000555555555803 <+84>:    mov    0x8(%rsp),%rax
   0x0000555555555808 <+89>:    sub    %fs:0x28,%rax
   0x0000555555555811 <+98>:    jne    0x55555555581f <phase_4+112>
   0x0000555555555813 <+100>:   add    $0x18,%rsp
   0x0000555555555817 <+104>:   ret
   0x0000555555555818 <+105>:   call   0x555555555e0f <explode_bomb>
   0x000055555555581d <+110>:   jmp    0x555555555803 <phase_4+84>
   0x000055555555581f <+112>:   call   0x555555555280 <__stack_chk_fail@plt>
End of assembler dump.
(gdb)
```

rax = 352

General Idea: * looking