Phase 1

humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb

GNU gdb (Ubuntu 10.1-2ubuntu2) 10.1.90.20210411-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb…
//set breakpoints
(gdb) b phase_1
Breakpoint 1 at 0x400e8d
(gdb) r
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
hi
Breakpoint 1, 0x0000000000400e8d in phase_1 ()

(gdb) disas
Dump of assembler code for function phase_1:
=> 0x0000000000400e8d <+0>: sub $0x8,%rsp **//building stack frame with 8 more bytes**
   0x0000000000400e91 <+4>: mov $0x4023d0,%esi **//what is this being moved?**
   0x0000000000400e96 <+9>: call 0x40133e <strings_not_equal>**//will compare input string with answer**
   0x0000000000400e9b <+14>: test %eax,%eax
   0x0000000000400e9d <+16>: je 0x400ea4 <phase_1+23>
   0x0000000000400e9f <+18>: call 0x40143d <explode_bomb>
   0x0000000000400ea4 <+23>: add $0x8,%rsp
   0x0000000000400ea8 <+27>: ret
End of assembler dump.
//Let inspect what is being moved from address 0x4023d0. We know it has to be a string of some sort so we use '/s'.

(gdb) x/s 0x4023d0
0x4023d0: "The moon unit will be divided into two divisions."

So this string is being moved into %esi, and will be passed into <string_not_equal>. Let's inspect what <string_not_equal> does:

```
//next instruction to line 2
(gdb) ni 2
0x0000000000400e96 in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
   0x0000000000400e8d <+0>:     sub    $0x8,%rsp
   0x0000000000400e91 <+4>:     mov    $0x4023d0,%esi
=> 0x0000000000400e96 <+9>:     call   0x40133e <strings_not_equal>
   0x0000000000400e9b <+14>:    test   %eax,%eax
   0x0000000000400e9d <+16>:    je     0x400ea4 <phase_1+23>
   0x0000000000400e9f <+18>:    call   0x40143d <explode_bomb>
   0x0000000000400ea4 <+23>:    add    $0x8,%rsp
   0x0000000000400ea8 <+27>:    ret
End of assembler dump.
(gdb) si
0x000000000040133e in strings_not_equal ()
(gdb) disas
Dump of assembler code for function strings_not_equal:
=> 0x000000000040133e <+0>:     push   %r12
   0x0000000000401340 <+2>:     push   %rbp
   0x0000000000401341 <+3>:     push   %rbx
   0x0000000000401342 <+4>:     mov    %rdi,%rbx
   0x0000000000401345 <+7>:     mov    %rsi,%rbp
   0x0000000000401348 <+10>:    call   0x401320 <string_length>
   0x000000000040134d <+15>:    mov    %eax,%r12d
   0x0000000000401350 <+18>:    mov    %rbp,%rdi
   0x0000000000401353 <+21>:    call   0x401320 <string_length>
   0x0000000000401358 <+26>:    mov    $0x1,%edx
   0x000000000040135d <+31>:    cmp    %eax,%r12d
   0x0000000000401360 <+34>:    jne    0x40139e <strings_not_equal+96>
   0x0000000000401362 <+36>:    movzbl (%rbx),%eax
   0x0000000000401365 <+39>:    test   %al,%al
   0x0000000000401367 <+41>:    je     0x40138b <strings_not_equal+77>
   0x0000000000401369 <+43>:    cmp    0x0(%rbp),%al
   0x000000000040136c <+46>:    je     0x401375 <strings_not_equal+55>
   0x000000000040136e <+48>:    jmp    0x401392 <strings_not_equal+84>
   0x0000000000401370 <+50>:    cmp    0x0(%rbp),%al
   0x0000000000401373 <+53>:    jne    0x401399 <strings_not_equal+91>
   0x0000000000401375 <+55>:    add    $0x1,%rbx
   0x0000000000401379 <+59>:    add    $0x1,%rbp
   0x000000000040137d <+63>:    movzbl (%rbx),%eax
   0x0000000000401380 <+66>:    test   %al,%al
   0x0000000000401382 <+68>:    jne    0x401370 <strings_not_equal+50>
   0x0000000000401384 <+70>:    mov    $0x0,%edx
   0x0000000000401389 <+75>:    jmp    0x40139e <strings_not_equal+96>
   0x000000000040138b <+77>:    mov    $0x0,%edx
   0x0000000000401390 <+82>:    jmp    0x40139e <strings_not_equal+96>
   0x0000000000401392 <+84>:    mov    $0x1,%edx
   0x0000000000401397 <+89>:    jmp    0x40139e <strings_not_equal+96>
   0x0000000000401399 <+91>:    mov    $0x1,%edx
   0x000000000040139e <+96>:    mov    %edx,%eax
   0x00000000004013a0 <+98>:    pop    %rbx
```

```
   0x00000000004013a1 <+99>:    pop    %rbp
   0x00000000004013a2 <+100>:   pop    %r12
   0x00000000004013a4 <+102>:   ret
End of assembler dump.
```

<String_not_equal> does not have a call to bomb, so it is okay to execute. Looking at %eax, we see it is = 1. So it will call the bomb.

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
hi

Breakpoint 1, 0x0000000000400e8d in phase_1 ()
(gdb) ni 3
0x0000000000400e9b in phase_1 ()
(gdb) i r
rax            0x1                1 //Will call the bomb as 1&1 will not give back zero
rbx            0x4021f0           4202992
rcx            0x2                2
rdx            0x1                1
rsi            0x4023d0           4203472
rdi            0x402401           4203521
rbp            0x0                0x0
rsp            0x7fffffffde20     0x7fffffffde20
r8             0x6037a0           6305696
r9             0x6046b0           6309552
r10            0x400669           4195945
r11            0x7ffff7def8a0     140737351973024
r12            0x400c60           4197472
r13            0x0                0
r14            0x0                0
r15            0x0                0
rip            0x400e9b           0x400e9b <phase_1+14>
eflags         0x287              [ CF PF SF IF ]
cs             0x33               51
ss             0x2b               43
ds             0x0                0
es             0x0                0
fs             0x0                0
gs             0x0                0
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb
```

Lets try to use the string we found in the disassembler code and see the value of %eax for it.

“The moon unit will be divided into two divisions.”

Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
The moon unit will be divided into two divisions.

Breakpoint 1, 0x0000000000400e8d in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x0000000000400e8d <+0>: sub $0x8,%rsp
0x0000000000400e91 <+4>: mov $0x4023d0,%esi
0x0000000000400e96 <+9>: call 0x40133e <strings_not_equal>
0x0000000000400e9b <+14>: test %eax,%eax
0x0000000000400e9d <+16>: je 0x400ea4 <phase_1+23>
0x0000000000400e9f <+18>: call 0x40143d <explode_bomb>
0x0000000000400ea4 <+23>: add $0x8,%rsp
0x0000000000400ea8 <+27>: ret
End of assembler dump.
(gdb) ni 3
0x0000000000400e9b in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
0x0000000000400e8d <+0>: sub $0x8,%rsp
0x0000000000400e91 <+4>: mov $0x4023d0,%esi
0x0000000000400e96 <+9>: call 0x40133e <strings_not_equal>
=> 0x0000000000400e9b <+14>: test %eax,%eax
0x0000000000400e9d <+16>: je 0x400ea4 <phase_1+23>
0x0000000000400e9f <+18>: call 0x40143d <explode_bomb>
0x0000000000400ea4 <+23>: add $0x8,%rsp
0x0000000000400ea8 <+27>: ret
End of assembler dump.
(gdb) i r
rax 0x0 0 **//%rax is equal to 0! which means it will jump pass the explode _bomb**
rbx 0x4021f0 4202992
rcx 0x31 49
rdx 0x0 0
rsi 0x4023d0 4203472
rdi 0x402401 4203521
rbp 0x0 0x0
rsp 0x7fffffffde20 0x7fffffffde20
r8 0x6037a0 6305696
r9 0x6046b0 6309552
r10 0x400669 4195945
r11 0x7ffff7def8a0 140737351973024
r12 0x400c60 4197472
r13 0x0 0
r14 0x0 0
r15 0x0 0
rip 0x400e9b 0x400e9b <phase_1+14>
eflags 0x246 [PF ZF IF]
cs 0x33 51

```
ss             0x2b              43
ds             0x0               0
es             0x0               0
fs             0x0               0
gs             0x0               0
```

```
//break point remove
(gdb) info break
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400e8d <phase_1>
          breakpoint already hit 1 time
(gdb) del 1
(gdb) info break
No breakpoints or watchpoints.
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
The moon unit will be divided into two divisions.
Phase 1 defused. How about the next one?
```

- **So solution in phase 1 is:** The moon unit will be divided into two divisions.

phase 2

```
humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb
GNU gdb (Ubuntu 10.1-2ubuntu2) 10.1.90.20210411-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb...
(gdb) b phase_2
Breakpoint 1 at 0x400ea9
(gdb) b explode_bomb
Breakpoint 2 at 0x40143d
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
hi//test input

Breakpoint 1, 0x0000000000400ea9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
=> 0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)//It is 1st input
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)//It is 2nd input
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
```

```
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) until * 0x0000000000400ec2
0x0000000000400ec2 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
=> 0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
```

//checking the input format

```
(gdb) si
0x000000000040145f in read_six_numbers ()
(gdb) disas
Dump of assembler code for function read_six_numbers:
=> 0x000000000040145f <+0>:     sub    $0x8,%rsp
   0x0000000000401463 <+4>:     mov    %rsi,%rdx
   0x0000000000401466 <+7>:     lea    0x4(%rsi),%rcx
   0x000000000040146a <+11>:    lea    0x14(%rsi),%rax
   0x000000000040146e <+15>:    push   %rax
   0x000000000040146f <+16>:    lea    0x10(%rsi),%rax
   0x0000000000401473 <+20>:    push   %rax
   0x0000000000401474 <+21>:    lea    0xc(%rsi),%r9
   0x0000000000401478 <+25>:    lea    0x8(%rsi),%r8
   0x000000000040147c <+29>:    mov    $0x4025c3,%esi
   0x0000000000401481 <+34>:    mov    $0x0,%eax
   0x0000000000401486 <+39>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x000000000040148b <+44>:    add    $0x10,%rsp
   0x000000000040148f <+48>:    cmp    $0x5,%eax
   0x0000000000401492 <+51>:    jg     0x401499 <read_six_numbers+58>
   0x0000000000401494 <+53>:    call   0x40143d <explode_bomb>
   0x0000000000401499 <+58>:    add    $0x8,%rsp
   0x000000000040149d <+62>:    ret
End of assembler dump.
(gdb) x/s 0x4025c3
0x4025c3:       "%d %d %d %d %d %d"//input format with six integer
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
0 1 2 3 4 5//try input with correct format

Breakpoint 1, 0x0000000000400ea9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
=> 0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
```

```
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
//checking the third input
(gdb) until *0x0000000000400ee6
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
```

```
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) i r
rax            0x1                 1 //the third input is 1 because the input store in eax for 32 bits and
rax for 64 bits
rbx            0x7fffffffdde0      140737488346592
rcx            0x0                 0
rdx            0x5                 5
rsi            0x0                 0
rdi            0x7fffffffd770      140737488344944
rbp            0x7fffffffddf0      0x7fffffffddf0
rsp            0x7fffffffdde0      0x7fffffffdde0
r8             0x1999999999999999  1844674407370955161
r9             0x0                 0
r10            0x7ffff7f49ac0      140737353390784
r11            0x7ffff7f4a3c0      140737353393088
r12            0x400c60            4197472
r13            0x0                 0
r14            0x0                 0
r15            0x0                 0
rip            0x400ee6            0x400ee6 <phase_2+61>
eflags         0x202               [ IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                 0

(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
```

```
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
=> 0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
```

```
   0x0000000000400f12 <+105>:    pop    %rbx
   0x0000000000400f13 <+106>:    pop    %rbp
   0x0000000000400f14 <+107>:    ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:      push   %rbp
   0x0000000000400eaa <+1>:      push   %rbx
   0x0000000000400eab <+2>:      sub    $0x28,%rsp
   0x0000000000400eaf <+6>:      mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:     mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:     xor    %eax,%eax
   0x0000000000400ebf <+22>:     mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:     call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:     cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:     jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:     cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:     je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:     call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:     mov    %rsp,%rbx
   0x0000000000400edc <+51>:     lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:     mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:     add    (%rbx),%eax
   0x0000000000400ee6 <+61>:     cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:     je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:     call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:     add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:     cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:     jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:     mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:     xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:     je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:    add    $0x28,%rsp
   0x0000000000400f12 <+105>:    pop    %rbx
   0x0000000000400f13 <+106>:    pop    %rbp
   0x0000000000400f14 <+107>:    ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:      push   %rbp
   0x0000000000400eaa <+1>:      push   %rbx
   0x0000000000400eab <+2>:      sub    $0x28,%rsp
   0x0000000000400eaf <+6>:      mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:     mov    %rax,0x18(%rsp)
```

```
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
```

```
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) i r
rax            0x2                 2 // the 4th input is 2 .

rbx            0x7fffffffdde4      140737488346596
rcx            0x0                 0
rdx            0x5                 5
rsi            0x0                 0
rdi            0x7fffffffd770      140737488344944
rbp            0x7fffffffddf0      0x7fffffffddf0
rsp            0x7fffffffdde0      0x7fffffffdde0
r8             0x1999999999999999  1844674407370955161
r9             0x0                 0
r10            0x7ffff7f49ac0      140737353390784
r11            0x7ffff7f4a3c0      140737353393088
r12            0x400c60            4197472
r13            0x0                 0
r14            0x0                 0
r15            0x0                 0
rip            0x400ee6            0x400ee6 <phase_2+61>
eflags         0x202               [ IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                 0
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
0 1 1 2 4 5
Breakpoint 1, 0x0000000000400ea9 in phase_2 ()
```

//checking the 5th input

```
(gdb) disas
Dump of assembler code for function phase_2:
=> 0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) until
0x0000000000400eaa in phase_2 ()
(gdb) until * 0x0000000000400ee6
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
```

```
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
```

```
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
```

```
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
```

```
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
```

```
   0x0000000000400f12 <+105>:     pop    %rbx
   0x0000000000400f13 <+106>:     pop    %rbp
   0x0000000000400f14 <+107>:     ret
End of assembler dump.
(gdb) i r
rax            0x2                2
rbx            0x7fffffffdde4     140737488346596
rcx            0x0                0
rdx            0x5                5
rsi            0x0                0
rdi            0x7fffffffd770     140737488344944
rbp            0x7fffffffddf0     0x7fffffffddf0
rsp            0x7fffffffdde0     0x7fffffffdde0
r8             0x1999999999999999 1844674407370955161
r9             0x0                0
r10            0x7ffff7f49ac0     140737353390784
r11            0x7ffff7f4a3c0     140737353393088
r12            0x400c60           4197472
r13            0x0                0
r14            0x0                0
r15            0x0                0
rip            0x400ee6           0x400ee6 <phase_2+61>
eflags         0x202              [ IF ]
cs             0x33               51
ss             0x2b               43
ds             0x0                0
es             0x0                0
fs             0x0                0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                0
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
```

```
=> 0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
```

```
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
```

```
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
```

```
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
=> 0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
//bomb is explode due to the wrong inputs
0x0000000000400eeb in phase_2 ()
(gdb) disas
```

```
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
=> 0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
```

//again run to check the 5th inputs

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
0 1 1 2 4 5//check the 5th inputs

Breakpoint 1, 0x0000000000400ea9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
=> 0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
```

```
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) until *0x0000000000400ef4
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
```

```
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
```

```
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()

(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
=> 0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
```

```
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
```

```
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) ni

(gdb) until *0x0000000000400ef4
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
```

```
   0x0000000000400f14 <+107>:     ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:       push   %rbp
   0x0000000000400eaa <+1>:       push   %rbx
   0x0000000000400eab <+2>:       sub    $0x28,%rsp
   0x0000000000400eaf <+6>:       mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:      mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:      xor    %eax,%eax
   0x0000000000400ebf <+22>:      mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:      call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:      cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:      jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:      cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:      je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:      call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:      mov    %rsp,%rbx
   0x0000000000400edc <+51>:      lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:      mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:      add    (%rbx),%eax
   0x0000000000400ee6 <+61>:      cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:      je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:      call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:      add    $0x4,%rbx
   0x0000000000400ef4 <+75>:      cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:      jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:      mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:      xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:      je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:      call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:     add    $0x28,%rsp
   0x0000000000400f12 <+105>:     pop    %rbx
   0x0000000000400f13 <+106>:     pop    %rbp
   0x0000000000400f14 <+107>:     ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:       push   %rbp
   0x0000000000400eaa <+1>:       push   %rbx
   0x0000000000400eab <+2>:       sub    $0x28,%rsp
   0x0000000000400eaf <+6>:       mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:      mov    %rax,0x18(%rsp)
```

```
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
=> 0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
```

```
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
```

```
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
```

```
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
```

```
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) i r
rax            0x3              3 //the 5th inputs is 3

rbx            0x7fffffffdde8   140737488346600
rcx            0x0              0
rdx            0x5              5
rsi            0x0              0
rdi            0x7fffffffd770   140737488344944
rbp            0x7fffffffddf0   0x7fffffffddf0
rsp            0x7fffffffdde0   0x7fffffffdde0
r8             0x1999999999999999  1844674407370955161
r9             0x0              0
r10            0x7ffff7f49ac0   140737353390784
r11            0x7ffff7f4a3c0   140737353393088
r12            0x400c60         4197472
r13            0x0              0
r14            0x0              0
r15            0x0              0
rip            0x400ee6         0x400ee6 <phase_2+61>
eflags         0x206            [ PF IF ]
cs             0x33             51
ss             0x2b             43
ds             0x0              0
es             0x0              0
fs             0x0              0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0              0
//checking for last inputs
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
0 1 1 2 3 6

Breakpoint 1, 0x0000000000400ea9 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
=> 0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
```

```
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) until *0x0000000000400ef4
0x0000000000400ef4 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
```

```
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
=> 0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
```

```
End of assembler dump.
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
```

```
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
```

```
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) disas
```

```
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
```

```
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
```

```
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.

(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee9 in phase_2 ()
(gdb) ni
0x0000000000400ef0 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
```

```
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
=> 0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ef4 in phase_2 ()
(gdb) ni
0x0000000000400ef7 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
```

```
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400ee1 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
=> 0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
```

```
End of assembler dump.
(gdb) ni
0x0000000000400ee4 in phase_2 ()
(gdb) ni
0x0000000000400ee6 in phase_2 ()
(gdb) disas
Dump of assembler code for function phase_2:
   0x0000000000400ea9 <+0>:     push   %rbp
   0x0000000000400eaa <+1>:     push   %rbx
   0x0000000000400eab <+2>:     sub    $0x28,%rsp
   0x0000000000400eaf <+6>:     mov    %fs:0x28,%rax
   0x0000000000400eb8 <+15>:    mov    %rax,0x18(%rsp)
   0x0000000000400ebd <+20>:    xor    %eax,%eax
   0x0000000000400ebf <+22>:    mov    %rsp,%rsi
   0x0000000000400ec2 <+25>:    call   0x40145f <read_six_numbers>
   0x0000000000400ec7 <+30>:    cmpl   $0x0,(%rsp)
   0x0000000000400ecb <+34>:    jne    0x400ed4 <phase_2+43>
   0x0000000000400ecd <+36>:    cmpl   $0x1,0x4(%rsp)
   0x0000000000400ed2 <+41>:    je     0x400ed9 <phase_2+48>
   0x0000000000400ed4 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
=> 0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x40143d <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) i r
rax            0x5                 5 // last input is 5
rbx            0x7fffffffddec      140737488346604
rcx            0x0                 0
rdx            0x6                 6
rsi            0x0                 0
rdi            0x7fffffffd770      140737488344944
rbp            0x7fffffffddf0      0x7fffffffddf0
rsp            0x7fffffffdde0      0x7fffffffdde0
r8             0x1999999999999999  1844674407370955161
r9             0x0                 0
r10            0x7ffff7f49ac0      140737353390784
```

```
r11            0x7ffff7f4a3c0      140737353393088
r12            0x400c60            4197472
r13            0x0                 0
r14            0x0                 0
r15            0x0                 0
rip            0x400ee6            0x400ee6 <phase_2+61>
eflags         0x206               [ PF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                 0
```

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
0 1 1 2 3 5
That's number 2.  Keep going!
```

- **So solution in phase 3 is:** 0 1 1 2 3 5

Phase 3

humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb
GNU gdb (Ubuntu 10.1-2ubuntu2) 10.1.90.20210411-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb...
(gdb) b phase_3
Breakpoint 1 at 0x400f15
(gdb) b explode_bomb
Breakpoint 2 at 0x40143d
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2. Keep going!
hi//**Test Input**

Breakpoint 1, 0x0000000000400f15 in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
=> 0x0000000000400f15 <+0>: sub $0x18,%rsp //**makes stack**
  0x0000000000400f19 <+4>: mov %fs:0x28,%rax
  0x0000000000400f22 <+13>: mov %rax,0x8(%rsp)
  0x0000000000400f27 <+18>: xor %eax,%eax
  0x0000000000400f29 <+20>: lea 0x4(%rsp),%rcx
  0x0000000000400f2e <+25>: mov %rsp,%rdx
  0x0000000000400f31 <+28>: mov $0x4025cf,%esi //**what we are moving from 0x4025cf**
  0x0000000000400f36 <+33>: call 0x400bb0 <__isoc99_sscanf@plt>//**functions that takes input**
  0x0000000000400f3b <+38>: cmp $0x1,%eax //**comparing 1 with result from input**
  0x0000000000400f3e <+41>: jg 0x400f45 <phase_3+48> //**if %eax > 1 the jump pass explode bomb function**
  0x0000000000400f40 <+43>: call 0x40143d <explode_bomb>
  0x0000000000400f45 <+48>: cmpl $0x7,(%rsp)//**comparing 7 with the value at rsp**
  0x0000000000400f49 <+52>: ja 0x400fa6 <phase_3+145> //**bomb explode, if it is greater than 7**

```
   0x0000000000400f4b <+54>:    mov    (%rsp),%eax
   0x0000000000400f4e <+57>:    jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:    mov    $0x134,%eax
   0x0000000000400f5a <+69>:    jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:    mov    $0x0,%eax
   0x0000000000400f61 <+76>:    sub    $0x85,%eax
   0x0000000000400f66 <+81>:    jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:    mov    $0x0,%eax
   0x0000000000400f6d <+88>:    add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:    jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:    mov    $0x0,%eax
   0x0000000000400f79 <+100>:   sub    $0x68,%eax
   0x0000000000400f7c <+103>:   jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:   mov    $0x0,%eax
   0x0000000000400f83 <+110>:   add    $0x68,%eax
   0x0000000000400f86 <+113>:   jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:   mov    $0x0,%eax
   0x0000000000400f8d <+120>:   sub    $0x68,%eax
   0x0000000000400f90 <+123>:   jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:   mov    $0x0,%eax
   0x0000000000400f97 <+130>:   add    $0x68,%eax
   0x0000000000400f9a <+133>:   jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:   mov    $0x0,%eax
   0x0000000000400fa1 <+140>:   sub    $0x68,%eax
   0x0000000000400fa4 <+143>:   jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:   call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:   mov    $0x0,%eax
   0x0000000000400fb0 <+155>:   cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:   jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:   cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:   je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:   call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:   mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:   xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:   je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:   add    $0x18,%rsp
   0x0000000000400fda <+197>:   ret
End of assembler dump.
```

Lets move the pointer and check what is there in 0x4025cf

```
(gdb) x/s 0x4025cf
0x4025cf:       "%d %d"//input format

(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
```

which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2. Keep going!
2 45//**test input with correct format**

//Checking the first input

Breakpoint 1, 0x0000000000400f15 in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
=> 0x0000000000400f15 <+0>: sub $0x18,%rsp
  0x0000000000400f19 <+4>: mov %fs:0x28,%rax
  0x0000000000400f22 <+13>: mov %rax,0x8(%rsp)
  0x0000000000400f27 <+18>: xor %eax,%eax
  0x0000000000400f29 <+20>: lea 0x4(%rsp),%rcx
  0x0000000000400f2e <+25>: mov %rsp,%rdx
  0x0000000000400f31 <+28>: mov $0x4025cf,%esi
  0x0000000000400f36 <+33>: call 0x400bb0 <__isoc99_sscanf@plt>
  0x0000000000400f3b <+38>: cmp $0x1,%eax
  0x0000000000400f3e <+41>: jg 0x400f45 <phase_3+48>
  0x0000000000400f40 <+43>: call 0x40143d <explode_bomb>
  0x0000000000400f45 <+48>: cmpl $0x7,(%rsp)
  0x0000000000400f49 <+52>: ja 0x400fa6 <phase_3+145>
  0x0000000000400f4b <+54>: mov (%rsp),%eax
  0x0000000000400f4e <+57>: jmp *0x402440(,%rax,8)
  0x0000000000400f55 <+64>: mov $0x134,%eax
  0x0000000000400f5a <+69>: jmp 0x400f61 <phase_3+76>
  0x0000000000400f5c <+71>: mov $0x0,%eax
  0x0000000000400f61 <+76>: sub $0x85,%eax
  0x0000000000400f66 <+81>: jmp 0x400f6d <phase_3+88>
  0x0000000000400f68 <+83>: mov $0x0,%eax
  0x0000000000400f6d <+88>: add $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
  0x0000000000400f72 <+93>: jmp 0x400f79 <phase_3+100>
  0x0000000000400f74 <+95>: mov $0x0,%eax
  0x0000000000400f79 <+100>: sub $0x68,%eax
  0x0000000000400f7c <+103>: jmp 0x400f83 <phase_3+110>
  0x0000000000400f7e <+105>: mov $0x0,%eax
  0x0000000000400f83 <+110>: add $0x68,%eax
  0x0000000000400f86 <+113>: jmp 0x400f8d <phase_3+120>
  0x0000000000400f88 <+115>: mov $0x0,%eax
  0x0000000000400f8d <+120>: sub $0x68,%eax
  0x0000000000400f90 <+123>: jmp 0x400f97 <phase_3+130>
  0x0000000000400f92 <+125>: mov $0x0,%eax
  0x0000000000400f97 <+130>: add $0x68,%eax
  0x0000000000400f9a <+133>: jmp 0x400fa1 <phase_3+140>
  0x0000000000400f9c <+135>: mov $0x0,%eax
  0x0000000000400fa1 <+140>: sub $0x68,%eax
  0x0000000000400fa4 <+143>: jmp 0x400fb0 <phase_3+155>
  0x0000000000400fa6 <+145>: call 0x40143d <explode_bomb>
  0x0000000000400fab <+150>: mov $0x0,%eax
  0x0000000000400fb0 <+155>: cmpl $0x5,(%rsp)

```
   0x0000000000400fb4 <+159>:     jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:     cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:     je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:     call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:     mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:     xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:     je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:     add    $0x18,%rsp
   0x0000000000400fda <+197>:     ret
End of assembler dump.

(gdb) until * 0x0000000000400f3b
0x0000000000400f3b in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:       sub    $0x18,%rsp
   0x0000000000400f19 <+4>:       mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:      mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:      xor    %eax,%eax
   0x0000000000400f29 <+20>:      lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:      mov    %rsp,%rdx
   0x0000000000400f31 <+28>:      mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:      call   0x400bb0 <__isoc99_sscanf@plt>
=> 0x0000000000400f3b <+38>:      cmp    $0x1,%eax
   0x0000000000400f3e <+41>:      jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:      call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:      cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:      ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:      mov    (%rsp),%eax
   0x0000000000400f4e <+57>:      jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:      mov    $0x134,%eax
   0x0000000000400f5a <+69>:      jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:      mov    $0x0,%eax
   0x0000000000400f61 <+76>:      sub    $0x85,%eax
   0x0000000000400f66 <+81>:      jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:      mov    $0x0,%eax
   0x0000000000400f6d <+88>:      add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:      jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:      mov    $0x0,%eax
   0x0000000000400f79 <+100>:     sub    $0x68,%eax
   0x0000000000400f7c <+103>:     jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:     mov    $0x0,%eax
   0x0000000000400f83 <+110>:     add    $0x68,%eax
   0x0000000000400f86 <+113>:     jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:     mov    $0x0,%eax
   0x0000000000400f8d <+120>:     sub    $0x68,%eax
   0x0000000000400f90 <+123>:     jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:     mov    $0x0,%eax
   0x0000000000400f97 <+130>:     add    $0x68,%eax
```

```
   0x0000000000400f9a <+133>:     jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:     mov    $0x0,%eax
   0x0000000000400fa1 <+140>:     sub    $0x68,%eax
   0x0000000000400fa4 <+143>:     jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:     call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:     mov    $0x0,%eax
   0x0000000000400fb0 <+155>:     cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:     jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:     cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:     je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:     call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:     mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:     xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:     je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:     add    $0x18,%rsp
   0x0000000000400fda <+197>:     ret
End of assembler dump.
(gdb) i r
rax            0x2                2 //The correct input for the first one is 2. Since eax holds the input
and it store in rax at 64 bits
rbx            0x7fffffffdf18     140737488346904
rcx            0x0                0
rdx            0x2d               45
rsi            0x0                0
rdi            0x7fffffffd7b0     140737488345008
rbp            0x0                0x0
rsp            0x7fffffffde00     0x7fffffffde00
r8             0x1999999999999999 1844674407370955161
r9             0x0                0
r10            0x7ffff7f49ac0     140737353390784
r11            0x7ffff7f4a3c0     140737353393088
r12            0x400c60           4197472
r13            0x0                0
r14            0x0                0
r15            0x0                0
rip            0x400f3b           0x400f3b <phase_3+38>
eflags         0x202              [ IF ]
cs             0x33               51
ss             0x2b               43
ds             0x0                0
es             0x0                0
fs             0x0                0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                0
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
```

which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2. Keep going!
2 90 // **check input for first by entering the correct input that fetch from above**

Breakpoint 1, 0x0000000000400f15 in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
=> 0x0000000000400f15 <+0>: sub $0x18,%rsp
  0x0000000000400f19 <+4>: mov %fs:0x28,%rax
  0x0000000000400f22 <+13>: mov %rax,0x8(%rsp)
  0x0000000000400f27 <+18>: xor %eax,%eax
  0x0000000000400f29 <+20>: lea 0x4(%rsp),%rcx
  0x0000000000400f2e <+25>: mov %rsp,%rdx
  0x0000000000400f31 <+28>: mov $0x4025cf,%esi
  0x0000000000400f36 <+33>: call 0x400bb0 <__isoc99_sscanf@plt>
  0x0000000000400f3b <+38>: cmp $0x1,%eax
  0x0000000000400f3e <+41>: jg 0x400f45 <phase_3+48>
  0x0000000000400f40 <+43>: call 0x40143d <explode_bomb>
  0x0000000000400f45 <+48>: cmpl $0x7,(%rsp)
  0x0000000000400f49 <+52>: ja 0x400fa6 <phase_3+145>
  0x0000000000400f4b <+54>: mov (%rsp),%eax
  0x0000000000400f4e <+57>: jmp *0x402440(,%rax,8)
  0x0000000000400f55 <+64>: mov $0x134,%eax
  0x0000000000400f5a <+69>: jmp 0x400f61 <phase_3+76>
  0x0000000000400f5c <+71>: mov $0x0,%eax
  0x0000000000400f61 <+76>: sub $0x85,%eax
  0x0000000000400f66 <+81>: jmp 0x400f6d <phase_3+88>
  0x0000000000400f68 <+83>: mov $0x0,%eax
  0x0000000000400f6d <+88>: add $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
  0x0000000000400f72 <+93>: jmp 0x400f79 <phase_3+100>
  0x0000000000400f74 <+95>: mov $0x0,%eax
  0x0000000000400f79 <+100>: sub $0x68,%eax
  0x0000000000400f7c <+103>: jmp 0x400f83 <phase_3+110>
  0x0000000000400f7e <+105>: mov $0x0,%eax
  0x0000000000400f83 <+110>: add $0x68,%eax
  0x0000000000400f86 <+113>: jmp 0x400f8d <phase_3+120>
  0x0000000000400f88 <+115>: mov $0x0,%eax
  0x0000000000400f8d <+120>: sub $0x68,%eax
  0x0000000000400f90 <+123>: jmp 0x400f97 <phase_3+130>
  0x0000000000400f92 <+125>: mov $0x0,%eax
  0x0000000000400f97 <+130>: add $0x68,%eax
  0x0000000000400f9a <+133>: jmp 0x400fa1 <phase_3+140>
  0x0000000000400f9c <+135>: mov $0x0,%eax
  0x0000000000400fa1 <+140>: sub $0x68,%eax
  0x0000000000400fa4 <+143>: jmp 0x400fb0 <phase_3+155>
  0x0000000000400fa6 <+145>: call 0x40143d <explode_bomb>
  0x0000000000400fab <+150>: mov $0x0,%eax
  0x0000000000400fb0 <+155>: cmpl $0x5,(%rsp)
  0x0000000000400fb4 <+159>: jg 0x400fbc <phase_3+167>
  0x0000000000400fb6 <+161>: cmp 0x4(%rsp),%eax

```
   0x0000000000400fba <+165>:     je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:     call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:     mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:     xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:     je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:     add    $0x18,%rsp
   0x0000000000400fda <+197>:     ret
End of assembler dump.
(gdb) until *0x0000000000400f3b
0x0000000000400f3b in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:       sub    $0x18,%rsp
   0x0000000000400f19 <+4>:       mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:      mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:      xor    %eax,%eax
   0x0000000000400f29 <+20>:      lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:      mov    %rsp,%rdx
   0x0000000000400f31 <+28>:      mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:      call   0x400bb0 <__isoc99_sscanf@plt>
=> 0x0000000000400f3b <+38>:      cmp    $0x1,%eax
   0x0000000000400f3e <+41>:      jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:      call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:      cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:      ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:      mov    (%rsp),%eax
   0x0000000000400f4e <+57>:      jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:      mov    $0x134,%eax
   0x0000000000400f5a <+69>:      jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:      mov    $0x0,%eax
   0x0000000000400f61 <+76>:      sub    $0x85,%eax
   0x0000000000400f66 <+81>:      jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:      mov    $0x0,%eax
   0x0000000000400f6d <+88>:      add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:      jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:      mov    $0x0,%eax
   0x0000000000400f79 <+100>:     sub    $0x68,%eax
   0x0000000000400f7c <+103>:     jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:     mov    $0x0,%eax
   0x0000000000400f83 <+110>:     add    $0x68,%eax
   0x0000000000400f86 <+113>:     jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:     mov    $0x0,%eax
   0x0000000000400f8d <+120>:     sub    $0x68,%eax
   0x0000000000400f90 <+123>:     jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:     mov    $0x0,%eax
   0x0000000000400f97 <+130>:     add    $0x68,%eax
   0x0000000000400f9a <+133>:     jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:     mov    $0x0,%eax
   0x0000000000400fa1 <+140>:     sub    $0x68,%eax
```

```
   0x0000000000400fa4 <+143>:     jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:     call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:     mov    $0x0,%eax
   0x0000000000400fb0 <+155>:     cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:     jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:     cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:     je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:     call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:     mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:     xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:     je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:     add    $0x18,%rsp
   0x0000000000400fda <+197>:     ret
End of assembler dump.
(gdb) ni
0x0000000000400f3e in phase_3 ()
(gdb) ni
0x0000000000400f45 in phase_3 ()
// It pass the explode bomb, since the input is correct
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:       sub    $0x18,%rsp
   0x0000000000400f19 <+4>:       mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:      mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:      xor    %eax,%eax
   0x0000000000400f29 <+20>:      lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:      mov    %rsp,%rdx
   0x0000000000400f31 <+28>:      mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:      call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000400f3b <+38>:      cmp    $0x1,%eax
   0x0000000000400f3e <+41>:      jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:      call   0x40143d <explode_bomb>
=> 0x0000000000400f45 <+48>:     cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:      ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:      mov    (%rsp),%eax
   0x0000000000400f4e <+57>:      jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:      mov    $0x134,%eax
   0x0000000000400f5a <+69>:      jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:      mov    $0x0,%eax
   0x0000000000400f61 <+76>:      sub    $0x85,%eax
   0x0000000000400f66 <+81>:      jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:      mov    $0x0,%eax
   0x0000000000400f6d <+88>:      add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:      jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:      mov    $0x0,%eax
   0x0000000000400f79 <+100>:     sub    $0x68,%eax
   0x0000000000400f7c <+103>:     jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:     mov    $0x0,%eax
   0x0000000000400f83 <+110>:     add    $0x68,%eax
```

```
   0x0000000000400f86 <+113>:    jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:    mov    $0x0,%eax
   0x0000000000400f8d <+120>:    sub    $0x68,%eax
   0x0000000000400f90 <+123>:    jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:    mov    $0x0,%eax
   0x0000000000400f97 <+130>:    add    $0x68,%eax
   0x0000000000400f9a <+133>:    jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:    mov    $0x0,%eax
   0x0000000000400fa1 <+140>:    sub    $0x68,%eax
   0x0000000000400fa4 <+143>:    jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:    call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:    mov    $0x0,%eax
   0x0000000000400fb0 <+155>:    cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:    jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:    cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:    je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:    mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:    xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:    je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:    add    $0x18,%rsp
   0x0000000000400fda <+197>:    ret
End of assembler dump.
(gdb) ni
0x0000000000400f49 in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:      sub    $0x18,%rsp
   0x0000000000400f19 <+4>:      mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:     xor    %eax,%eax
   0x0000000000400f29 <+20>:     lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:     mov    %rsp,%rdx
   0x0000000000400f31 <+28>:     mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000400f3b <+38>:     cmp    $0x1,%eax
   0x0000000000400f3e <+41>:     jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:     call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:     cmpl   $0x7,(%rsp)
=> 0x0000000000400f49 <+52>:     ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:     mov    (%rsp),%eax
   0x0000000000400f4e <+57>:     jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:     mov    $0x134,%eax
   0x0000000000400f5a <+69>:     jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:     mov    $0x0,%eax
   0x0000000000400f61 <+76>:     sub    $0x85,%eax
   0x0000000000400f66 <+81>:     jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:     mov    $0x0,%eax
   0x0000000000400f6d <+88>:     add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
```

```
   0x0000000000400f72 <+93>:    jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:    mov    $0x0,%eax
   0x0000000000400f79 <+100>:   sub    $0x68,%eax
   0x0000000000400f7c <+103>:   jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:   mov    $0x0,%eax
   0x0000000000400f83 <+110>:   add    $0x68,%eax
   0x0000000000400f86 <+113>:   jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:   mov    $0x0,%eax
   0x0000000000400f8d <+120>:   sub    $0x68,%eax
   0x0000000000400f90 <+123>:   jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:   mov    $0x0,%eax
   0x0000000000400f97 <+130>:   add    $0x68,%eax
   0x0000000000400f9a <+133>:   jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:   mov    $0x0,%eax
   0x0000000000400fa1 <+140>:   sub    $0x68,%eax
   0x0000000000400fa4 <+143>:   jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:   call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:   mov    $0x0,%eax
   0x0000000000400fb0 <+155>:   cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:   jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:   cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:   je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:   call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:   mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:   xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:   je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:   add    $0x18,%rsp
   0x0000000000400fda <+197>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400f4b in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:     sub    $0x18,%rsp
   0x0000000000400f19 <+4>:     mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:    xor    %eax,%eax
   0x0000000000400f29 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:    mov    %rsp,%rdx
   0x0000000000400f31 <+28>:    mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000400f3b <+38>:    cmp    $0x1,%eax
   0x0000000000400f3e <+41>:    jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:    cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:    ja     0x400fa6 <phase_3+145>
=> 0x0000000000400f4b <+54>:    mov    (%rsp),%eax
   0x0000000000400f4e <+57>:    jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:    mov    $0x134,%eax
   0x0000000000400f5a <+69>:    jmp    0x400f61 <phase_3+76>
```

```
   0x0000000000400f5c <+71>:    mov    $0x0,%eax
   0x0000000000400f61 <+76>:    sub    $0x85,%eax
   0x0000000000400f66 <+81>:    jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:    mov    $0x0,%eax
   0x0000000000400f6d <+88>:    add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:    jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:    mov    $0x0,%eax
   0x0000000000400f79 <+100>:   sub    $0x68,%eax
   0x0000000000400f7c <+103>:   jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:   mov    $0x0,%eax
   0x0000000000400f83 <+110>:   add    $0x68,%eax
   0x0000000000400f86 <+113>:   jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:   mov    $0x0,%eax
   0x0000000000400f8d <+120>:   sub    $0x68,%eax
   0x0000000000400f90 <+123>:   jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:   mov    $0x0,%eax
   0x0000000000400f97 <+130>:   add    $0x68,%eax
   0x0000000000400f9a <+133>:   jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:   mov    $0x0,%eax
   0x0000000000400fa1 <+140>:   sub    $0x68,%eax
   0x0000000000400fa4 <+143>:   jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:   call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:   mov    $0x0,%eax
   0x0000000000400fb0 <+155>:   cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:   jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:   cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:   je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:   call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:   mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:   xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:   je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:   add    $0x18,%rsp
   0x0000000000400fda <+197>:   ret
End of assembler dump.
(gdb) until
0x0000000000400f4e in phase_3 ()
```

//To next compare instruction

```
(gdb) until * 0x0000000000400fb0
0x0000000000400fb0 in phase_3 ()
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:     sub    $0x18,%rsp
   0x0000000000400f19 <+4>:     mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:    xor    %eax,%eax
   0x0000000000400f29 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:    mov    %rsp,%rdx
   0x0000000000400f31 <+28>:    mov    $0x4025cf,%esi
```

```
   0x0000000000400f36 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000400f3b <+38>:    cmp    $0x1,%eax
   0x0000000000400f3e <+41>:    jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:    cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:    ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:    mov    (%rsp),%eax
   0x0000000000400f4e <+57>:    jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:    mov    $0x134,%eax
   0x0000000000400f5a <+69>:    jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:    mov    $0x0,%eax
   0x0000000000400f61 <+76>:    sub    $0x85,%eax
   0x0000000000400f66 <+81>:    jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:    mov    $0x0,%eax
   0x0000000000400f6d <+88>:    add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:    jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:    mov    $0x0,%eax
   0x0000000000400f79 <+100>:   sub    $0x68,%eax
   0x0000000000400f7c <+103>:   jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:   mov    $0x0,%eax
   0x0000000000400f83 <+110>:   add    $0x68,%eax
   0x0000000000400f86 <+113>:   jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:   mov    $0x0,%eax
   0x0000000000400f8d <+120>:   sub    $0x68,%eax
   0x0000000000400f90 <+123>:   jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:   mov    $0x0,%eax
   0x0000000000400f97 <+130>:   add    $0x68,%eax
   0x0000000000400f9a <+133>:   jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:   mov    $0x0,%eax
   0x0000000000400fa1 <+140>:   sub    $0x68,%eax
   0x0000000000400fa4 <+143>:   jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:   call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:   mov    $0x0,%eax
=> 0x0000000000400fb0 <+155>:   cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:   jg     0x400fbc <phase_3+167>
   0x0000000000400fb6 <+161>:   cmp    0x4(%rsp),%eax
   0x0000000000400fba <+165>:   je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:   call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:   mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:   xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:   je     0x400fd6 <phase_3+193>
   0x0000000000400fd1 <+188>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:   add    $0x18,%rsp
   0x0000000000400fda <+197>:   ret
End of assembler dump.
(gdb) ni
0x0000000000400fb4 in phase_3 ()
(gdb) ni
0x0000000000400fb6 in phase_3 ()
```

//To check the compare instruction and checking the second input

```
(gdb) disas
Dump of assembler code for function phase_3:
   0x0000000000400f15 <+0>:     sub    $0x18,%rsp
   0x0000000000400f19 <+4>:     mov    %fs:0x28,%rax
   0x0000000000400f22 <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000400f27 <+18>:    xor    %eax,%eax
   0x0000000000400f29 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000400f2e <+25>:    mov    %rsp,%rdx
   0x0000000000400f31 <+28>:    mov    $0x4025cf,%esi
   0x0000000000400f36 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000400f3b <+38>:    cmp    $0x1,%eax
   0x0000000000400f3e <+41>:    jg     0x400f45 <phase_3+48>
   0x0000000000400f40 <+43>:    call   0x40143d <explode_bomb>
   0x0000000000400f45 <+48>:    cmpl   $0x7,(%rsp)
   0x0000000000400f49 <+52>:    ja     0x400fa6 <phase_3+145>
   0x0000000000400f4b <+54>:    mov    (%rsp),%eax
   0x0000000000400f4e <+57>:    jmp    *0x402440(,%rax,8)
   0x0000000000400f55 <+64>:    mov    $0x134,%eax
   0x0000000000400f5a <+69>:    jmp    0x400f61 <phase_3+76>
   0x0000000000400f5c <+71>:    mov    $0x0,%eax
   0x0000000000400f61 <+76>:    sub    $0x85,%eax
   0x0000000000400f66 <+81>:    jmp    0x400f6d <phase_3+88>
   0x0000000000400f68 <+83>:    mov    $0x0,%eax
   0x0000000000400f6d <+88>:    add    $0x201,%eax
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f72 <+93>:    jmp    0x400f79 <phase_3+100>
   0x0000000000400f74 <+95>:    mov    $0x0,%eax
   0x0000000000400f79 <+100>:   sub    $0x68,%eax
   0x0000000000400f7c <+103>:   jmp    0x400f83 <phase_3+110>
   0x0000000000400f7e <+105>:   mov    $0x0,%eax
   0x0000000000400f83 <+110>:   add    $0x68,%eax
   0x0000000000400f86 <+113>:   jmp    0x400f8d <phase_3+120>
   0x0000000000400f88 <+115>:   mov    $0x0,%eax
   0x0000000000400f8d <+120>:   sub    $0x68,%eax
   0x0000000000400f90 <+123>:   jmp    0x400f97 <phase_3+130>
   0x0000000000400f92 <+125>:   mov    $0x0,%eax
   0x0000000000400f97 <+130>:   add    $0x68,%eax
   0x0000000000400f9a <+133>:   jmp    0x400fa1 <phase_3+140>
   0x0000000000400f9c <+135>:   mov    $0x0,%eax
   0x0000000000400fa1 <+140>:   sub    $0x68,%eax
   0x0000000000400fa4 <+143>:   jmp    0x400fb0 <phase_3+155>
   0x0000000000400fa6 <+145>:   call   0x40143d <explode_bomb>
   0x0000000000400fab <+150>:   mov    $0x0,%eax
   0x0000000000400fb0 <+155>:   cmpl   $0x5,(%rsp)
   0x0000000000400fb4 <+159>:   jg     0x400fbc <phase_3+167>
=> 0x0000000000400fb6 <+161>:   cmp    0x4(%rsp),%eax//checking the cmp instruction
   0x0000000000400fba <+165>:   je     0x400fc1 <phase_3+172>
   0x0000000000400fbc <+167>:   call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400fc1 <+172>:   mov    0x8(%rsp),%rax
   0x0000000000400fc6 <+177>:   xor    %fs:0x28,%rax
   0x0000000000400fcf <+186>:   je     0x400fd6 <phase_3+193>
```

```
   0x0000000000400fd1 <+188>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400fd6 <+193>:    add    $0x18,%rsp
   0x0000000000400fda <+197>:    ret
End of assembler dump.
(gdb) i r
rax            0x199               409 //The second input is 409 because eax holds the correct input.
The rax is 64 bits and eax is for 32 bits therefore 409 the input store in rax.
rbx            0x7fffffffdf18      140737488346904
rcx            0x0                 0
rdx            0x5a                90
rsi            0x0                 0
rdi            0x7fffffffd7b0      140737488345008
rbp            0x0                 0x0
rsp            0x7fffffffde00      0x7fffffffde00
r8             0x1999999999999999  1844674407370955161
r9             0x0                 0
r10            0x7ffff7f49ac0      140737353390784
r11            0x7ffff7f4a3c0      140737353393088
r12            0x400c60            4197472
r13            0x0                 0
r14            0x0                 0
r15            0x0                 0
rip            0x400fb6            0x400fb6 <phase_3+161>
eflags         0x293               [ CF AF SF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
--Type <RET> for more, q to quit, c to continue without paging--
gs             0x0                 0

//run to check the input
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
2 409 // The correct input that fetch from above is entering to defuse the bomb phase three
Halfway there!
```

- **So solution in phase 3 is:** 2 and 409