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 2 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

Phase 4

```
humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb
GNU gdb (Ubuntu 11.1-0ubuntu2) 11.1
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_4
Breakpoint 1 at 0x40100e
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb answers.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
2 23//test input

Breakpoint 1, 0x000000000040100e in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
=> 0x000000000040100e <+0>:     sub    $0x18,%rsp//makes stack frame
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi//answer format: %d %d
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax//eax have to contain 2
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>//if eax is NOT equal to 2 jump to explode bomb
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)// compare rsp with 14
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54> //if first input < 14, skip bomb exploded.
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx//edx=14
```

0x0000000000401049 <+59>: mov $0x0,%esi//**esi=0**
0x000000000040104e <+64>: mov (%rsp),%edi//**edi= First input**
0x0000000000401051 <+67>: call 0x400fdb <func4>//**call func4 function**
0x0000000000401056 <+72>: cmp $0x23,%eax//**compare second input with 0x23.**
0x0000000000401059 <+75>: jne 0x401062 <phase_4+84>//**if the second input is not equal to 0x23. Then bomb will exploded.**
0x000000000040105b <+77>: cmpl $0x23,0x4(%rsp)//**compare once more**
0x0000000000401060 <+82>: je 0x401067 <phase_4+89>//**if it is equal then skip bomb exploded**
0x0000000000401062 <+84>: call 0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
0x0000000000401067 <+89>: mov 0x8(%rsp),%rax
0x000000000040106c <+94>: xor %fs:0x28,%rax
0x0000000000401075 <+103>: je 0x40107c <phase_4+110>
0x0000000000401077 <+105>: call 0x400b00 <__stack_chk_fail@plt>
0x000000000040107c <+110>: add $0x18,%rsp
0x0000000000401080 <+114>: ret
End of assembler dump.

//Try first input

humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb
GNU gdb (Ubuntu 11.1-0ubuntu2) 11.1
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_4
Breakpoint 1 at 0x40100e
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb answers.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
2 23// **test input**

```
Breakpoint 1, 0x000000000040100e in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
=> 0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) u* 0x0000000000401034
0x0000000000401034 in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
=> 0x0000000000401034 <+38>:    cmp    $0x2,%eax//eax is compared with 2
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>//if not equal, bomb exploded
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
```

```
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401037 in phase_4 ()
//Checking eax or rax value, the rax or eax value is equal to 2, it skip the bomb exploded
(gdb) i r
rax            0x2                 2// eax or rax value is 2
rbx            0x7fffffffdef8      140737488346872
rcx            0x0                 0
rdx            0x17                23
rsi            0x17                23
rdi            0x7fffffffd760      140737488344928
rbp            0x2                 0x2
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x401037            0x401037 <phase_4+41>
eflags         0x246               [ PF ZF 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
0x0000000000401039 in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
```

```
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
=> 0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)//0xe = 14
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x000000000040103d in phase_4 ()
//First input should below 0xe(14)
(gdb) x/d $rsp
0x7fffffffddb0:          2
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
=> 0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
```

```
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401044 in phase_4 ()
(gdb) ni
0x0000000000401049 in phase_4 ()
//Since my first input is below 14, it skip bomb exploded.
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
=> 0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x000000000040104e in phase_4 ()
(gdb) disas
```

```
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
=> 0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401051 in phase_4 ()
(gdb) ni
0x0000000000401056 in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
```

```
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
=> 0x0000000000401056 <+72>:    cmp    $0x23,%eax//comparing eax value with 0x23.
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401059 in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
=> 0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
```

//Checking value of rax, rax is not equal to 0x23(2*16^1+3*16^0=35). So it will explode the bomb, Input is wrong

```
(gdb) i r
```

```
rax            0xd                13
rbx            0x7fffffffdef8     140737488346872
rcx            0x0                0
rdx            0x2                2
rsi            0x2                2
rdi            0x2                2
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x401059           0x401059 <phase_4+75>
eflags         0x283              [ CF 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
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
=> 0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
```

```
   0x0000000000401075 <+103>:    je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:    call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:    add    $0x18,%rsp
   0x0000000000401080 <+114>:    ret
End of assembler dump.
(gdb)
```

//Try second input

```
humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$
gdb bomb
GNU gdb (Ubuntu 11.1-0ubuntu2) 11.1
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_4
Breakpoint 1 at 0x40100e
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment
1/bomb001/bomb answers.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
8 35//test input

Breakpoint 1, 0x000000000040100e in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
=> 0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
```

```
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) u*  0x0000000000401034
0x0000000000401034 in phase_4 ()
(gdb) i r
rax            0x2                 2//checking eax or rax value to compare with 0x2
rbx            0x7fffffffdef8      140737488346872
rcx            0x0                 0
rdx            0x23                35
rsi            0x23                35
rdi            0x7fffffffd760      140737488344928
rbp            0x2                 0x2
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x401034            0x401034 <phase_4+38>
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) disas
Dump of assembler code for function phase_4:
```

```
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
=> 0x0000000000401034 <+38>:    cmp    $0x2,%eax//compare
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401037 in phase_4 ()
(gdb) ni
0x0000000000401039 in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
=> 0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
```

```
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
```

//Checking rsp(first input), first input < 0xe(14). The rsp value is 8, which is less than 14.

```
(gdb) x/d $rsp
0x7fffffffddb0:         8
(gdb) ni
0x000000000040103d in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
=> 0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
```

```
0x0000000000401044 in phase_4 ()
```
//Since the rsp is less than 0xe(14), it skip bomb explode.
```
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
=> 0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401049 in phase_4 ()
(gdb) ni
0x000000000040104e in phase_4 ()
(gdb) ni
0x0000000000401051 in phase_4 ()
(gdb) disas
```
//function func4 is calling.
```
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
```

```
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
=> 0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x0000000000401056 in phase_4 ()
(gdb) ni
//checking rax or eax value, eax or rax value is 35. eax value is compared with
0x23(2*16^1+3*16^0=35)
0x0000000000401059 in phase_4 ()
(gdb) i r
rax            0x23               35
rbx            0x7fffffffdef8     140737488346872
rcx            0x0                0
rdx            0x8                8
rsi            0x8                8
rdi            0x8                8
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x401059           0x401059 <phase_4+75>
eflags         0x246              [ PF ZF 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) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
=> 0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
(gdb) ni
0x000000000040105b in phase_4 ()
//Since the eax is equal to 0x23, it skip the explode bomb
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
```

```
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
=> 0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:   call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:   add    $0x18,%rsp
   0x0000000000401080 <+114>:   ret
End of assembler dump.
```
//Second input is 35
```
(gdb) x/d 0x4+$rsp
0x7fffffffddb4:         35
(gdb) ni
0x0000000000401060 in phase_4 ()
(gdb) ni
0x0000000000401067 in phase_4 ()
```
//Since my second input (35) is equal to 0x23. It skip bomb explode.
```
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:     sub    $0x18,%rsp
   0x0000000000401012 <+4>:     mov    %fs:0x28,%rax
   0x000000000040101b <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:    xor    %eax,%eax
   0x0000000000401022 <+20>:    lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:    mov    %rsp,%rdx
   0x000000000040102a <+28>:    mov    $0x4025cf,%esi
   0x000000000040102f <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:    cmp    $0x2,%eax
   0x0000000000401037 <+41>:    jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:    cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:    jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:    call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:    mov    $0xe,%edx
   0x0000000000401049 <+59>:    mov    $0x0,%esi
   0x000000000040104e <+64>:    mov    (%rsp),%edi
   0x0000000000401051 <+67>:    call   0x400fdb <func4>
   0x0000000000401056 <+72>:    cmp    $0x23,%eax
   0x0000000000401059 <+75>:    jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:    cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:    je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:    call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x0000000000401067 <+89>:    mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:    xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:   je     0x40107c <phase_4+110>
```

```
   0x0000000000401077 <+105>:    call   0x400b00 <__stack_chk_fail@plt>
   0x000000000040107c <+110>:    add    $0x18,%rsp
   0x0000000000401080 <+114>:    ret
End of assembler dump.
(gdb) ni
0x000000000040106c in phase_4 ()
(gdb) ni
0x0000000000401075 in phase_4 ()
(gdb) ni
0x000000000040107c in phase_4 ()
(gdb) disas
Dump of assembler code for function phase_4:
   0x000000000040100e <+0>:      sub    $0x18,%rsp
   0x0000000000401012 <+4>:      mov    %fs:0x28,%rax
   0x000000000040101b <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000401020 <+18>:     xor    %eax,%eax
   0x0000000000401022 <+20>:     lea    0x4(%rsp),%rcx
   0x0000000000401027 <+25>:     mov    %rsp,%rdx
   0x000000000040102a <+28>:     mov    $0x4025cf,%esi
   0x000000000040102f <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x0000000000401034 <+38>:     cmp    $0x2,%eax
   0x0000000000401037 <+41>:     jne    0x40103f <phase_4+49>
   0x0000000000401039 <+43>:     cmpl   $0xe,(%rsp)
   0x000000000040103d <+47>:     jbe    0x401044 <phase_4+54>
   0x000000000040103f <+49>:     call   0x40143d <explode_bomb>
   0x0000000000401044 <+54>:     mov    $0xe,%edx
   0x0000000000401049 <+59>:     mov    $0x0,%esi
   0x000000000040104e <+64>:     mov    (%rsp),%edi
   0x0000000000401051 <+67>:     call   0x400fdb <func4>
   0x0000000000401056 <+72>:     cmp    $0x23,%eax
   0x0000000000401059 <+75>:     jne    0x401062 <phase_4+84>
   0x000000000040105b <+77>:     cmpl   $0x23,0x4(%rsp)
   0x0000000000401060 <+82>:     je     0x401067 <phase_4+89>
   0x0000000000401062 <+84>:     call   0x40143d <explode_bomb>
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000401067 <+89>:     mov    0x8(%rsp),%rax
   0x000000000040106c <+94>:     xor    %fs:0x28,%rax
   0x0000000000401075 <+103>:    je     0x40107c <phase_4+110>
   0x0000000000401077 <+105>:    call   0x400b00 <__stack_chk_fail@plt>
=> 0x000000000040107c <+110>:    add    $0x18,%rsp
   0x0000000000401080 <+114>:    ret
End of assembler dump.
(gdb) ni
0x0000000000401080 in phase_4 ()
(gdb) ni
main (argc=<optimized out>, argv=<optimized out>) at bomb.c:96
96          phase_defused();
```
//run the program again without breakpoints and phase 4 is defused.
```
(gdb) r answers.txt
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
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
8 35
So you got that one. Try this one.

- **So solution in phase 4 is:** 8 35

Phase 5

humble@humble-Vostro-3583:~/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001$ gdb bomb
GNU gdb (Ubuntu 11.1-0ubuntu2) 11.1
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_5
Breakpoint 1 at 0x401081
(gdb) r answers.txt
Starting program: /home/humble/Desktop/References/3rd Year/ITS304/Assignment 1/bomb001/bomb answers.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
So you got that one. Try this one.
5 1//**test input**

Breakpoint 1, 0x0000000000401081 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
=> 0x0000000000401081 <+0>: sub $0x18,%rsp//**makes stack frame**
  0x0000000000401085 <+4>: mov %fs:0x28,%rax
  0x000000000040108e <+13>: mov %rax,0x8(%rsp)
  0x0000000000401093 <+18>: xor %eax,%eax
  0x0000000000401095 <+20>: lea 0x4(%rsp),%rcx
  0x000000000040109a <+25>: mov %rsp,%rdx
  0x000000000040109d <+28>: mov $0x4025cf,%esi//**answer format: %d %d**
  0x00000000004010a2 <+33>: call 0x400bb0 <__isoc99_sscanf@plt>
  0x00000000004010a7 <+38>: cmp $0x1,%eax//**eax holds the number of inputs, compare eax with 0x1**
  0x00000000004010aa <+41>: jg 0x4010b1 <phase_5+48>
  0x00000000004010ac <+43>: call 0x40143d <explode_bomb>
  0x00000000004010b1 <+48>: mov (%rsp),%eax
  0x00000000004010b4 <+51>: and $0xf,%eax

```
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax//first should below 0xf(15). Compare
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>//if eax and 0xf is equal then it
calls the exploded bomb
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax//compare eax with 0xf
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>//if eax is not equal to 0xf,
then jump into 72. The loop starts and the instructions are in looping until the eax value is
equal to 0xf. When the eax value is equal to 0xf, then loop will stop and execute the next
instructions.
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx//edx holds or keep tracking, the number
of loops taking place.
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx//0x4(%rsp) holds input that I
entered and ecx holds the real input. And compare, if it is equal then it skip the explode bomb.
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) x/s 0x4025cf
0x4025cf:       "%d %d"//input format
(gdb) disas
Dump of assembler code for function phase_5:
=> 0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
```

```
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u*  0x00000000004010a7
0x00000000004010a7 in phase_5 ()
(gdb) ni
0x00000000004010aa in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
=> 0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
```

```
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
```

//checking eax or rax value, it should greater than 1.

```
End of assembler dump.
(gdb) i r
rax            0x2                2
rbx            0x7fffffffdef8     140737488346872
rcx            0x0                0
rdx            0x1                1
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010aa           0x4010aa <phase_5+41>
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) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
=> 0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
```

```
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010b1 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
=> 0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
```

```
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010b4 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
=> 0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
```

```
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
(gdb) u *0x00000000004010ba
0x00000000004010ba in phase_5 ()
(gdb) ni
0x00000000004010bd in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:      sub    $0x18,%rsp
   0x0000000000401085 <+4>:      mov    %fs:0x28,%rax
   0x000000000040108e <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:     xor    %eax,%eax
   0x0000000000401095 <+20>:     lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:     mov    %rsp,%rdx
   0x000000000040109d <+28>:     mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:     cmp    $0x1,%eax
   0x00000000004010aa <+41>:     jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:     call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:     mov    (%rsp),%eax
   0x00000000004010b4 <+51>:     and    $0xf,%eax
   0x00000000004010b7 <+54>:     mov    %eax,(%rsp)
   0x00000000004010ba <+57>:     cmp    $0xf,%eax
=> 0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:     mov    $0x0,%ecx
   0x00000000004010c4 <+67>:     mov    $0x0,%edx
   0x00000000004010c9 <+72>:     add    $0x1,%edx
   0x00000000004010cc <+75>:     cltq
   0x00000000004010ce <+77>:     mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:     add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:     cmp    $0xf,%eax
   0x00000000004010da <+89>:     jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:     movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:     cmp    $0xf,%edx
   0x00000000004010e6 <+101>:    jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:    cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:    je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
```

//checking eax or rax value, compare eax or rax with 0xf. It should not be equal.

```
(gdb) i r
rax            0x5                5
rbx            0x7fffffffdef8     140737488346872
rcx            0x0                0
rdx            0x1                1
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010bd           0x4010bd <phase_5+60>
eflags         0x297              [ CF PF 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
(gdb) ni
0x00000000004010bf in phase_5 ()
```

//Since the eax value and 0xf is not equal, it executes the next instructions.

```
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
=> 0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
```

```
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
```

```
   0x00000000004010ee <+109>:     call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:     mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:     xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:     je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:     call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:     add    $0x18,%rsp
   0x000000000040110c <+139>:     ret
```

//checking the eax or rax value at 1st iteration.

```
End of assembler dump.
(gdb) i r
rax            0xc                12
rbx            0x7fffffffdef8     140737488346872
rcx            0xc                12
rdx            0x1                1
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
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) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
```

```
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010cc in phase_5 ()
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
```

```
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 2nd iteration.

```
(gdb) i r
rax            0x3                3
rbx            0x7fffffffdef8     140737488346872
rcx            0xf                15
rdx            0x2                2
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
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) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
```

```
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
```

```
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 3rd iteration.

```
(gdb) i r
rax            0x7                7
rbx            0x7fffffffdef8     140737488346872
rcx            0x16               22
rdx            0x3                3
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
eflags         0x212              [ AF 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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
```

```
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 4th iteration.

```
(gdb) i r
rax            0xb                11
rbx            0x7fffffffdef8     140737488346872
rcx            0x21               33
rdx            0x4                4
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
```

```
r13            0x400d56          4197718
r14            0x0               0
r15            0x7ffff7ffbc40    140737354120256
rip            0x4010d7          0x4010d7 <phase_5+86>
eflags         0x216             [ PF AF 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
0x00000000004010da in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
=> 0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
```

```
   0x000000000040110c <+139>:    ret
End of assembler dump.
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
```

```
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 5th iteration.

```
(gdb) i r
rax            0xd                13
rbx            0x7fffffffdef8     140737488346872
rcx            0x2e               46
rdx            0x5                5
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
```

```
rip            0x4010d7         0x4010d7 <phase_5+86>
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) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
```

```
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 6th iteration.

```
(gdb) i r
rax            0x9                9
rbx            0x7fffffffdef8     140737488346872
rcx            0x37               55
rdx            0x6                6
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
```

```
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x4010d7            0x4010d7 <phase_5+86>
eflags         0x212               [ AF 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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
```

```
   0x00000000004010e8 <+103>:    cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:    je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:      sub    $0x18,%rsp
   0x0000000000401085 <+4>:      mov    %fs:0x28,%rax
   0x000000000040108e <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:     xor    %eax,%eax
   0x0000000000401095 <+20>:     lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:     mov    %rsp,%rdx
   0x000000000040109d <+28>:     mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:     cmp    $0x1,%eax
   0x00000000004010aa <+41>:     jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:     call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:     mov    (%rsp),%eax
   0x00000000004010b4 <+51>:     and    $0xf,%eax
   0x00000000004010b7 <+54>:     mov    %eax,(%rsp)
   0x00000000004010ba <+57>:     cmp    $0xf,%eax
   0x00000000004010bd <+60>:     je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:     mov    $0x0,%ecx
   0x00000000004010c4 <+67>:     mov    $0x0,%edx
   0x00000000004010c9 <+72>:     add    $0x1,%edx
   0x00000000004010cc <+75>:     cltq
   0x00000000004010ce <+77>:     mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:     add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:     cmp    $0xf,%eax
   0x00000000004010da <+89>:     jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:     movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:     cmp    $0xf,%edx
   0x00000000004010e6 <+101>:    jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:    cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:    je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
```

//checking the eax or rax value at 7th iteration.

```
(gdb) i r
rax            0x4                4
rbx            0x7fffffffdef8     140737488346872
rcx            0x3b               59
rdx            0x7                7
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
```

```
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
```

//checking the eax or rax value at 8th iteration.

```
(gdb) i r
rax            0x8                8
rbx            0x7fffffffdef8     140737488346872
rcx            0x43               67
rdx            0x8                8
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
eflags         0x212              [ AF 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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
```

```
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
```

```
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 9th iteration.

```
(gdb) i r
rax            0x0                 0
rbx            0x7fffffffdef8      140737488346872
rcx            0x43                67
rdx            0x9                 9
rsi            0x1                 1
rdi            0x7fffffffd760      140737488344928
rbp            0x2                 0x2
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x4010d7            0x4010d7 <phase_5+86>
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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
```

```
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
```

```
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
```

```
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 10th iteration.

```
(gdb) i r
rax            0x1                 1
rbx            0x7fffffffdef8      140737488346872
rcx            0x4e                78
rdx            0xb                 11
rsi            0x1                 1
rdi            0x7fffffffd760      140737488344928
rbp            0x2                 0x2
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x4010d7            0x4010d7 <phase_5+86>
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) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
```

```
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
```

//checking the eax or rax value at 11th iteration.

```
(gdb) i r
rax            0x2                2
rbx            0x7fffffffdef8     140737488346872
rcx            0x50               80
rdx            0xc                12
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
```

```
r8             0x0                 0
r9             0x0                 0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56            4197718
r14            0x0                 0
r15            0x7ffff7ffbc40      140737354120256
rip            0x4010d7            0x4010d7 <phase_5+86>
eflags         0x216               [ PF AF 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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
```

```
   0x00000000004010ec <+107>:    je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:      sub    $0x18,%rsp
   0x0000000000401085 <+4>:      mov    %fs:0x28,%rax
   0x000000000040108e <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:     xor    %eax,%eax
   0x0000000000401095 <+20>:     lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:     mov    %rsp,%rdx
   0x000000000040109d <+28>:     mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:     cmp    $0x1,%eax
   0x00000000004010aa <+41>:     jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:     call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:     mov    (%rsp),%eax
   0x00000000004010b4 <+51>:     and    $0xf,%eax
   0x00000000004010b7 <+54>:     mov    %eax,(%rsp)
   0x00000000004010ba <+57>:     cmp    $0xf,%eax
   0x00000000004010bd <+60>:     je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:     mov    $0x0,%ecx
   0x00000000004010c4 <+67>:     mov    $0x0,%edx
   0x00000000004010c9 <+72>:     add    $0x1,%edx
   0x00000000004010cc <+75>:     cltq
   0x00000000004010ce <+77>:     mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:     add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:     cmp    $0xf,%eax
   0x00000000004010da <+89>:     jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:     movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:     cmp    $0xf,%edx
   0x00000000004010e6 <+101>:    jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:    cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:    je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
```

//checking the eax or rax value at 12th iteration.

```
(gdb) i r
rax            0xe                14
rbx            0x7fffffffdef8      140737488346872
rcx            0x5e               94
rdx            0xd                13
rsi            0x1                1
rdi            0x7fffffffd760      140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0      0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0      140737353329344
r11            0x7ffff7f3b3c0      140737353331648
r12            0x7fffffffdef8      140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40      140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
```

```
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
```

```
   0x00000000004010ee <+109>:    call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:    mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:    xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:    je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:    add    $0x18,%rsp
   0x000000000040110c <+139>:    ret
End of assembler dump.
```

//checking the eax or rax value at 13th iteration.

```
(gdb) i r
rax            0x6                6
rbx            0x7fffffffdef8     140737488346872
rcx            0x64               100
rdx            0xe                14
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
eflags         0x212              [ AF 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
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010c9 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:      sub    $0x18,%rsp
   0x0000000000401085 <+4>:      mov    %fs:0x28,%rax
   0x000000000040108e <+13>:     mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:     xor    %eax,%eax
   0x0000000000401095 <+20>:     lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:     mov    %rsp,%rdx
   0x000000000040109d <+28>:     mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:     call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:     cmp    $0x1,%eax
   0x00000000004010aa <+41>:     jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:     call   0x40143d <explode_bomb>
```

```
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
=> 0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) u * 0x00000000004010d7
0x00000000004010d7 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
```

```
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
```

//checking the eax or rax value at 14th iteration.

```
(gdb) i r
rax            0xf                15
rbx            0x7fffffffdef8     140737488346872
rcx            0x73               115
rdx            0xf                15
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010d7           0x4010d7 <phase_5+86>
eflags         0x212              [ AF 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) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
```

```
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
=> 0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010da in phase_5 ()
(gdb) ni
0x00000000004010dc in phase_5 ()
```

//Since the eax value is equal to 0xf, the loop stops and executes the next instructions.

```
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
```

```
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
=> 0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010e3 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
```

```
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
=> 0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) i r
rax            0xf                15
rbx            0x7fffffffdef8     140737488346872
rcx            0x73               115
rdx            0xf                15
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010e3           0x4010e3 <phase_5+98>
eflags         0x246              [ PF ZF 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
0x00000000004010e6 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
```

```
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
=> 0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) ni
0x00000000004010e8 in phase_5 ()
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
```

```
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
=> 0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
   0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
//checking the second input, and 0x4(%rsp) holds my second inputs.
(gdb) x/d 0x4+$rsp
0x7fffffffddb4:          1
(gdb) i r
rax            0xf                15
rbx            0x7fffffffdef8     140737488346872
rcx            0x73               115//this ecx or rcx holds the second input, the second input is 115.
rdx            0xf                15
rsi            0x1                1
rdi            0x7fffffffd760     140737488344928
rbp            0x2                0x2
rsp            0x7fffffffddb0     0x7fffffffddb0
r8             0x0                0
r9             0x0                0
r10            0x7ffff7f3aac0     140737353329344
r11            0x7ffff7f3b3c0     140737353331648
r12            0x7fffffffdef8     140737488346872
r13            0x400d56           4197718
r14            0x0                0
r15            0x7ffff7ffbc40     140737354120256
rip            0x4010e8           0x4010e8 <phase_5+103>
eflags         0x246              [ PF ZF 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
0x00000000004010ec in phase_5 ()
```

//Since I kept second input as 1, and it is not equal to the system input. So the bomb explode function is called.

```
(gdb) disas
Dump of assembler code for function phase_5:
   0x0000000000401081 <+0>:     sub    $0x18,%rsp
   0x0000000000401085 <+4>:     mov    %fs:0x28,%rax
   0x000000000040108e <+13>:    mov    %rax,0x8(%rsp)
   0x0000000000401093 <+18>:    xor    %eax,%eax
   0x0000000000401095 <+20>:    lea    0x4(%rsp),%rcx
   0x000000000040109a <+25>:    mov    %rsp,%rdx
   0x000000000040109d <+28>:    mov    $0x4025cf,%esi
   0x00000000004010a2 <+33>:    call   0x400bb0 <__isoc99_sscanf@plt>
   0x00000000004010a7 <+38>:    cmp    $0x1,%eax
   0x00000000004010aa <+41>:    jg     0x4010b1 <phase_5+48>
   0x00000000004010ac <+43>:    call   0x40143d <explode_bomb>
   0x00000000004010b1 <+48>:    mov    (%rsp),%eax
   0x00000000004010b4 <+51>:    and    $0xf,%eax
   0x00000000004010b7 <+54>:    mov    %eax,(%rsp)
   0x00000000004010ba <+57>:    cmp    $0xf,%eax
   0x00000000004010bd <+60>:    je     0x4010ee <phase_5+109>
   0x00000000004010bf <+62>:    mov    $0x0,%ecx
   0x00000000004010c4 <+67>:    mov    $0x0,%edx
   0x00000000004010c9 <+72>:    add    $0x1,%edx
   0x00000000004010cc <+75>:    cltq
   0x00000000004010ce <+77>:    mov    0x402480(,%rax,4),%eax
   0x00000000004010d5 <+84>:    add    %eax,%ecx
--Type <RET> for more, q to quit, c to continue without paging--
   0x00000000004010d7 <+86>:    cmp    $0xf,%eax
   0x00000000004010da <+89>:    jne    0x4010c9 <phase_5+72>
   0x00000000004010dc <+91>:    movl   $0xf,(%rsp)
   0x00000000004010e3 <+98>:    cmp    $0xf,%edx
   0x00000000004010e6 <+101>:   jne    0x4010ee <phase_5+109>
   0x00000000004010e8 <+103>:   cmp    0x4(%rsp),%ecx
=> 0x00000000004010ec <+107>:   je     0x4010f3 <phase_5+114>
   0x00000000004010ee <+109>:   call   0x40143d <explode_bomb>
   0x00000000004010f3 <+114>:   mov    0x8(%rsp),%rax
   0x00000000004010f8 <+119>:   xor    %fs:0x28,%rax
   0x0000000000401101 <+128>:   je     0x401108 <phase_5+135>
   0x0000000000401103 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401108 <+135>:   add    $0x18,%rsp
   0x000000000040110c <+139>:   ret
End of assembler dump.
(gdb) r answers.txt
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
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
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!
Halfway there!
So you got that one. Try this one.
5 115// **correct input**
Good work! On to the next…

- **So solution in phase 5 is:** 5 115