test a, b → a AND b → If both A and B are zero → set zero flag

cmp a, b → b - a → sets zero and sign flag

3 helped us avoid bomb →

rsp has our input

```
cmpl $0x7, (%rsp)
ja   explode bomb
```

integer must be 7 or below

↳ our first number in our input!

```
lea    0x1AB6(%rip), %rdx
movslq (%rdx, %rax, 4), %rax
```

0x555555557180
+ 0x C
0x555555557180C

This is the address before the 3.4 gets added with this and gets sent to %rax.

0x3.4 → 0011 << 2 → 1100 → C

After checking address values:
0xffffe582 is what gets stored into %rax

```
  0x555555557180   %rdx
+ 0xffffffffe582   %rax
  0x555555555732   address
```

19-16 = 3
21-16 = 5
21-16 = 5

2 8
0010 1000
32 + 8 = 40

```
mov  $0x2e4, %eax
then jumps to ... 710
cmp  %eax, 0x4(%rsp)
```

↳ our second # from the input!

```
jne  explode bomb
```

↳ If our second # is not equal to what's in eax, the bomb explodes!

∴ We have two values for this function
value 1: How far are we going to jump the first time (gives a different value we are moving into %eax)
value 2: The value in %eax

0x2e4

0010 1110 0100

512+128+64+32+4 =

36

100

228

```
  512
+ 228
  740
```

740 → Required 2nd input

Answer (one out of 8): 3 740