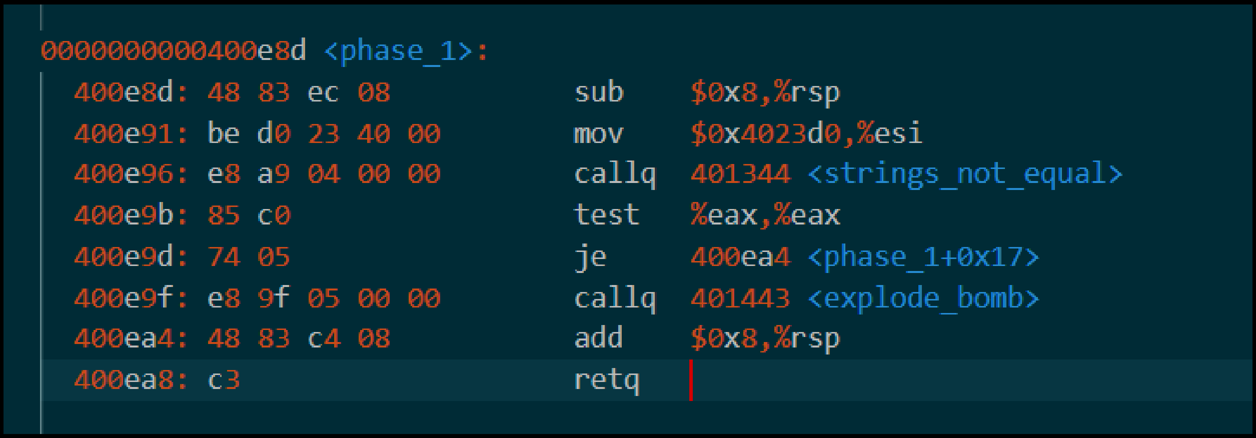
CA BOMB LAB(5)

YASH GUPTA
S20200010234

1) PHASE-1

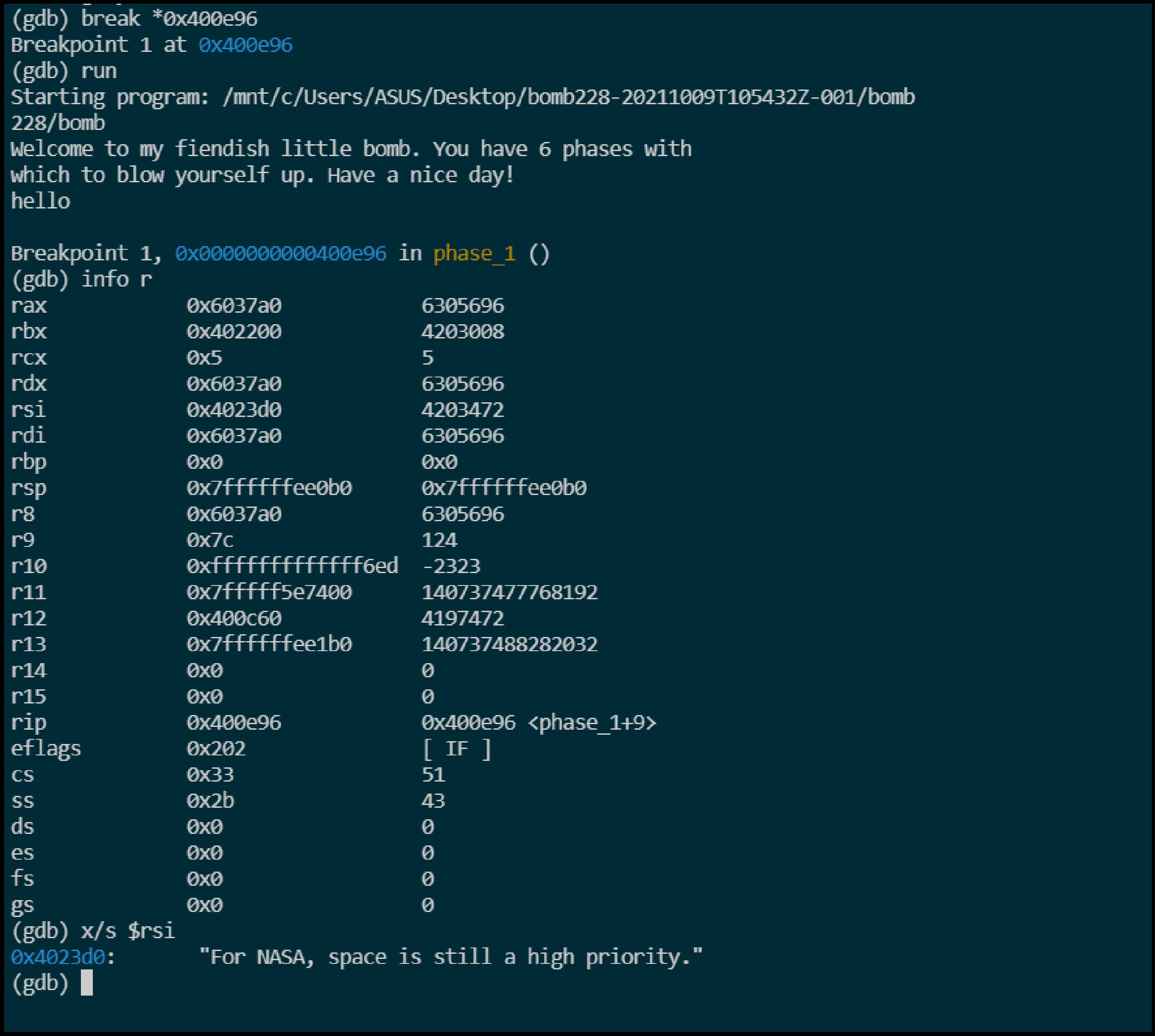
>Set the break point at <String not equal> i.e address 0x400e96



```
0000000000400e8d <phase_1>:
  400e8d: 48 83 ec 08             sub    $0x8,%rsp
  400e91: be d0 23 40 00          mov    $0x4023d0,%esi
  400e96: e8 a9 04 00 00          callq  401344 <strings_not_equal>
  400e9b: 85 c0                   test   %eax,%eax
  400e9d: 74 05                   je     400ea4 <phase_1+0x17>
  400e9f: e8 9f 05 00 00          callq  401443 <explode_bomb>
  400ea4: 48 83 c4 08             add    $0x8,%rsp
  400ea8: c3                      retq
```

> Look for the register which contains the correct string value

> Check the value stored in register that is the correct key for phase 1



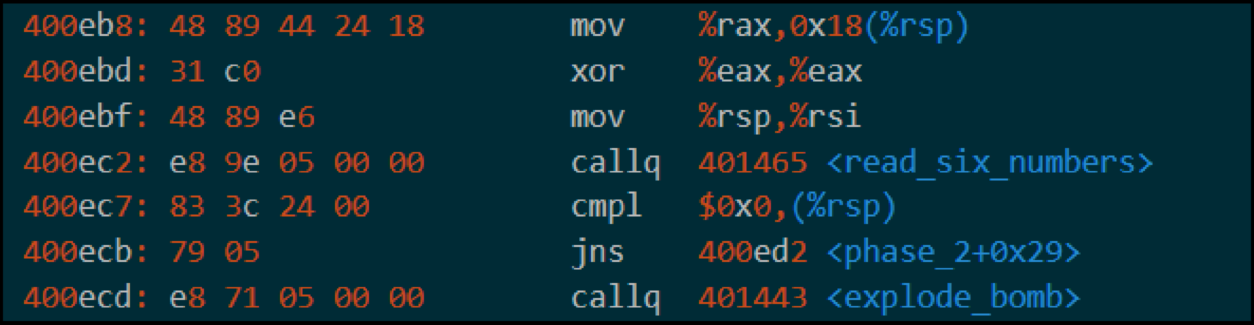
```
(gdb) break *0x400e96
Breakpoint 1 at 0x400e96
(gdb) run
Starting program: /mnt/c/Users/ASUS/Desktop/bomb228-20211009T105432Z-001/bomb
228/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
hello

Breakpoint 1, 0x0000000000400e96 in phase_1 ()
(gdb) info r
rax            0x6037a0            6305696
rbx            0x402200            4203008
rcx            0x5                 5
rdx            0x6037a0            6305696
rsi            0x4023d0            4203472
rdi            0x6037a0            6305696
rbp            0x0                 0x0
rsp            0x7fffffffee0b0     0x7fffffffee0b0
r8             0x6037a0            6305696
r9             0x7c                124
r10            0xfffffffffffff6ed  -2323
r11            0x7fffff5e7400      140737477768192
r12            0x400c60            4197472
r13            0x7ffffffee1b0      140737488282032
r14            0x0                 0
r15            0x0                 0
rip            0x400e96            0x400e96 <phase_1+9>
eflags         0x202               [ IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
gs             0x0                 0
(gdb) x/s $rsi
0x4023d0:       "For NASA, space is still a high priority."
(gdb) 
```

>Hence the string value =”For NASA, space is still a high priority.”

2) PHASE-2

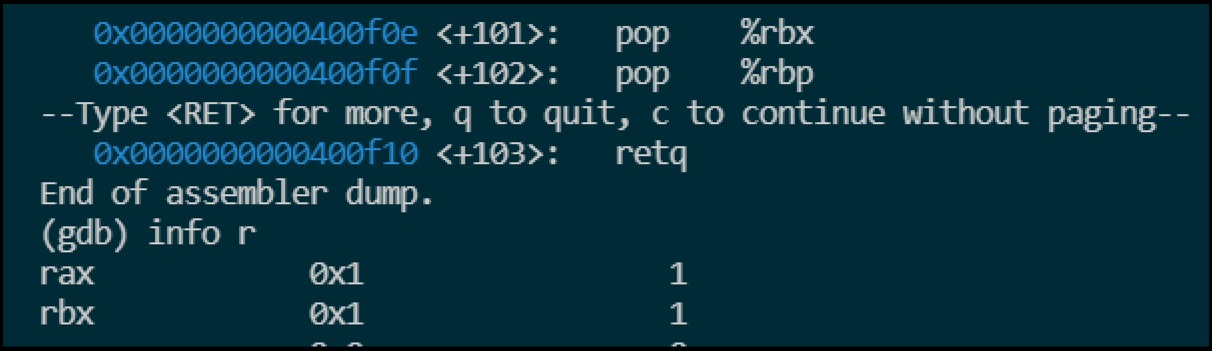
>Assuming 6 numbers must be taken.



```
400eb8: 48 89 44 24 18          mov    %rax,0x18(%rsp)
400ebd: 31 c0                   xor    %eax,%eax
400ebf: 48 89 e6                mov    %rsp,%rsi
400ec2: e8 9e 05 00 00          callq  401465 <read_six_numbers>
400ec7: 83 3c 24 00             cmpl   $0x0,(%rsp)
400ecb: 79 05                   jns    400ed2 <phase_2+0x29>
400ecd: e8 71 05 00 00          callq  401443 <explode_bomb>
```

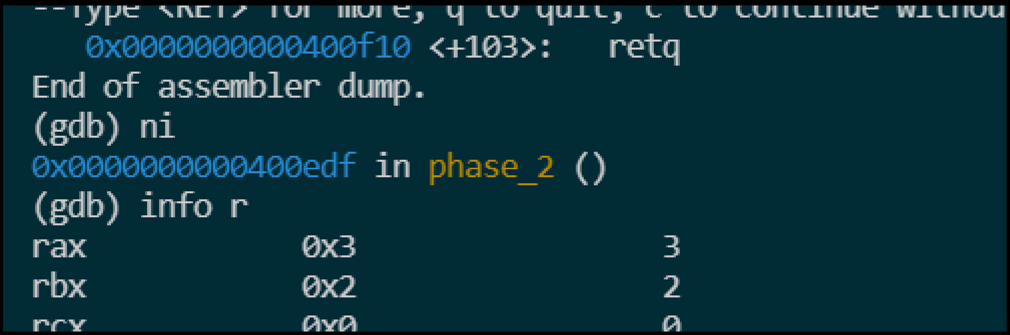
>From this we can conclude that the first number is 0 or else the bomb will explode.

>Run gdb and add a breakpoint at before theexplode_bomb call in phase 2



```
   0x0000000000400f0e <+101>:   pop    %rbx
   0x0000000000400f0f <+102>:   pop    %rbp
--Type <RET> for more, q to quit, c to continue without paging--
   0x0000000000400f10 <+103>:   retq
End of assembler dump.
(gdb) info r
rax            0x1                 1
rbx            0x1                 1
rcx            0x0                 0
```

>Check the register value for rax to get the next value. Then update the input given with the found value



```
--Type <RET> for more, q to quit, c to continue withou
   0x0000000000400f10 <+103>:   retq
End of assembler dump.
(gdb) ni
0x0000000000400edf in phase_2 ()
(gdb) info r
rax            0x3                 3
rbx            0x2                 2
rcx            0x0                 0
```

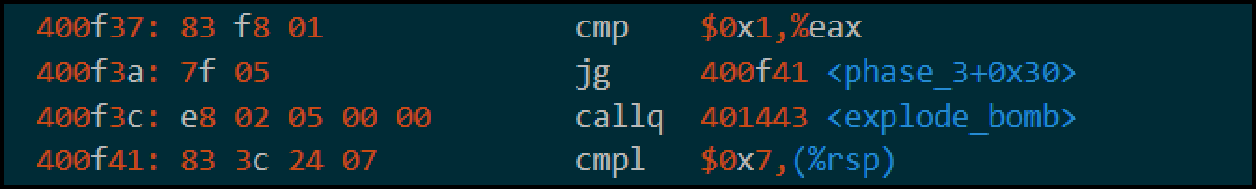
>Similarly repeat this process and find the value of all 6 digits

>After solving you will get the key as “0 1 3 6 10 15”

3) PHASE -3

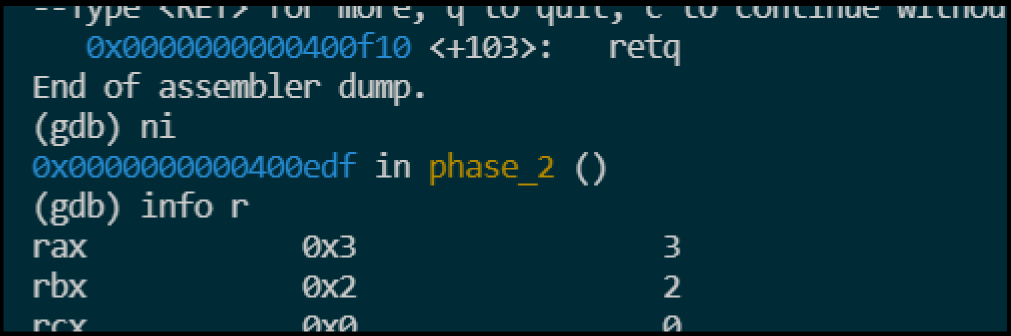
>By looking at the code we can conclude that the input consist two integers

>First integer is in the range of 1 to 7

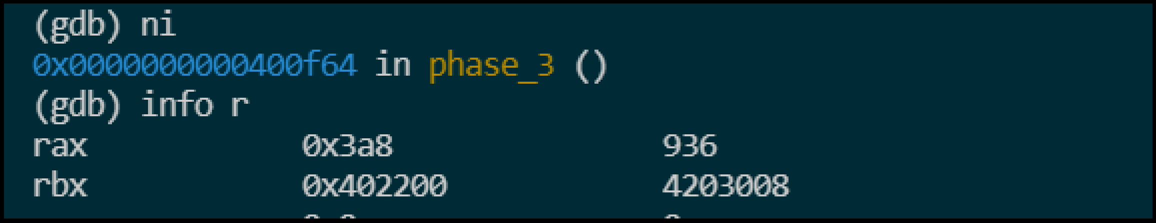


```
  400f37: 83 f8 01              cmp    $0x1,%eax
  400f3a: 7f 05                 jg     400f41 <phase_3+0x30>
  400f3c: e8 02 05 00 00        callq  401443 <explode_bomb>
  400f41: 83 3c 24 07           cmpl   $0x7,(%rsp)
```

>By checking the value at registers we get both the values



```
--Type <RET> for more, q to quit, c to continue withou
   0x0000000000400f10 <+103>:   retq
End of assembler dump.
(gdb) ni
0x0000000000400edf in phase_2 ()
(gdb) info r
rax            0x3                 3
rbx            0x2                 2
rcx            0x0                 0
```

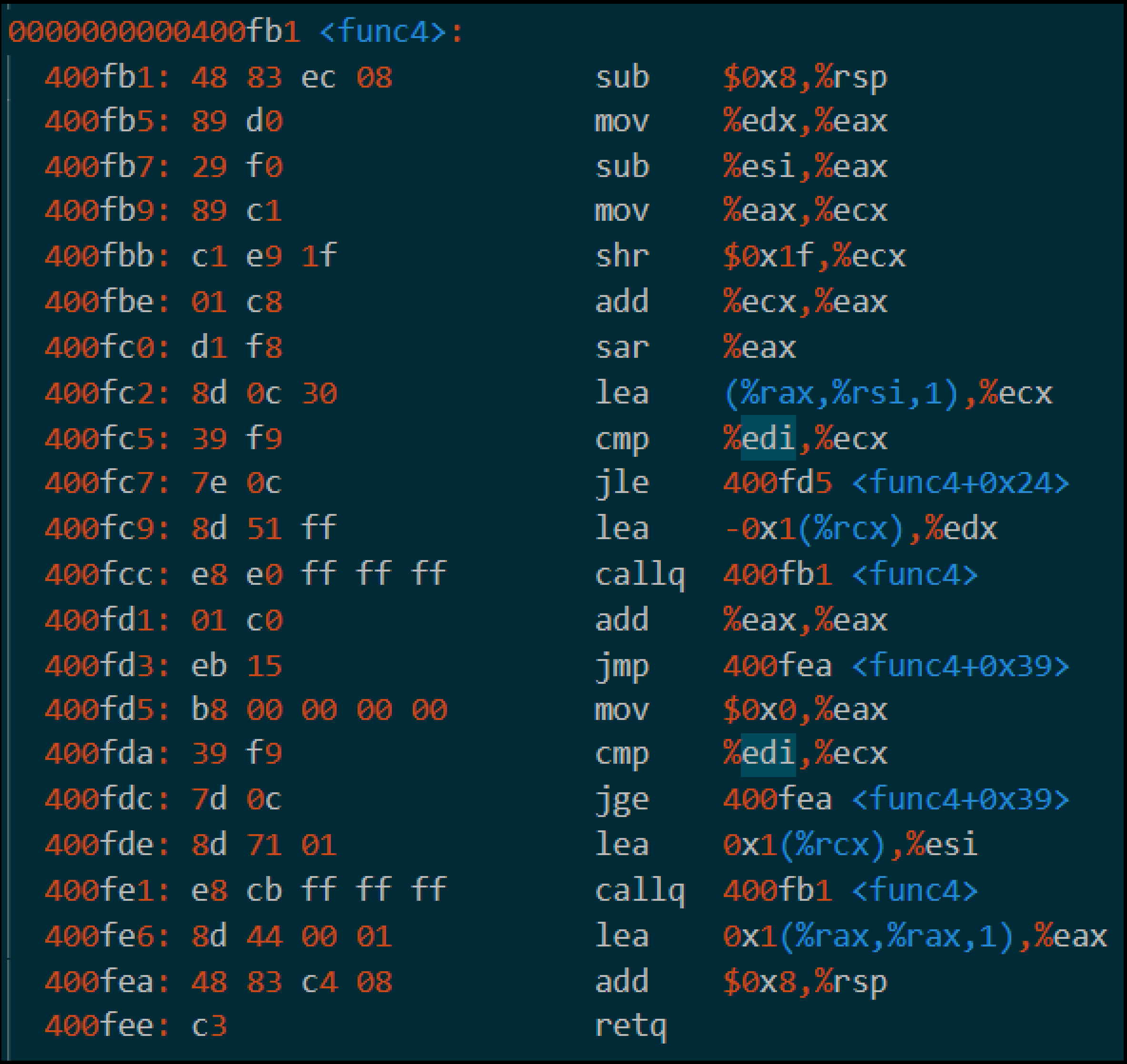


```
(gdb) ni
0x0000000000400f64 in phase_3 ()
(gdb) info r
rax            0x3a8               936
rbx            0x402200            4203008
```

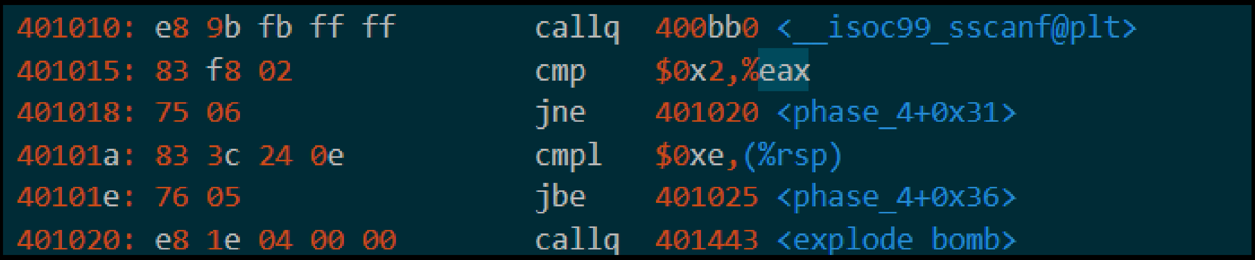
>Hence the key of this phase =”3 936”

4) PHASE 4

>After reading the whole phase_4 code, we came to know that the key contains two integers and one of them is passed through a function called func4, it returns an integer and my key will be correct only if the returned number is equal to the other number of input.



```
0000000000400fb1 <func4>:
  400fb1: 48 83 ec 08             sub    $0x8,%rsp
  400fb5: 89 d0                   mov    %edx,%eax
  400fb7: 29 f0                   sub    %esi,%eax
  400fb9: 89 c1                   mov    %eax,%ecx
  400fbb: c1 e9 1f                shr    $0x1f,%ecx
  400fbe: 01 c8                   add    %ecx,%eax
  400fc0: d1 f8                   sar    %eax
  400fc2: 8d 0c 30                lea    (%rax,%rsi,1),%ecx
  400fc5: 39 f9                   cmp    %edi,%ecx
  400fc7: 7e 0c                   jle    400fd5 <func4+0x24>
  400fc9: 8d 51 ff                lea    -0x1(%rcx),%edx
  400fcc: e8 e0 ff ff ff          callq  400fb1 <func4>
  400fd1: 01 c0                   add    %eax,%eax
  400fd3: eb 15                   jmp    400fea <func4+0x39>
  400fd5: b8 00 00 00 00          mov    $0x0,%eax
  400fda: 39 f9                   cmp    %edi,%ecx
  400fdc: 7d 0c                   jge    400fea <func4+0x39>
  400fde: 8d 71 01                lea    0x1(%rcx),%esi
  400fe1: e8 cb ff ff ff          callq  400fb1 <func4>
  400fe6: 8d 44 00 01             lea    0x1(%rax,%rax,1),%eax
  400fea: 48 83 c4 08             add    $0x8,%rsp
  400fee: c3                      retq
```

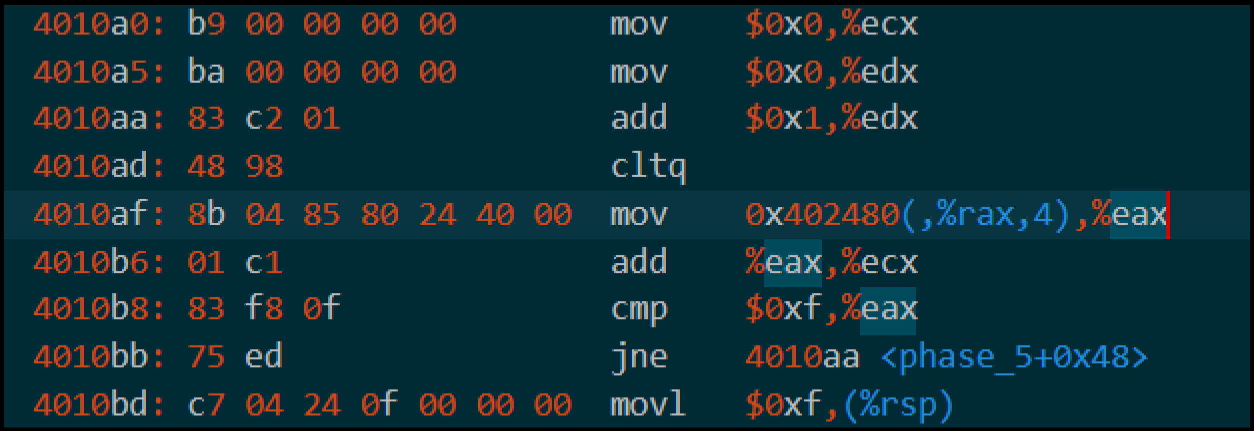


```
401010: e8 9b fb ff ff          callq  400bb0 <__isoc99_sscanf@plt>
401015: 83 f8 02                cmp    $0x2,%eax
401018: 75 06                   jne    401020 <phase_4+0x31>
40101a: 83 3c 24 0e             cmpl   $0xe,(%rsp)
40101e: 76 05                   jbe    401025 <phase_4+0x36>
401020: e8 1e 04 00 00          callq  401443 <explode_bomb>
```

>After checking the value at registers we came to know the key of phase 4 =”4 2”.

5) PHASE 5

> After reading the whole phase_5 code, we came to know that the key contains two integers.

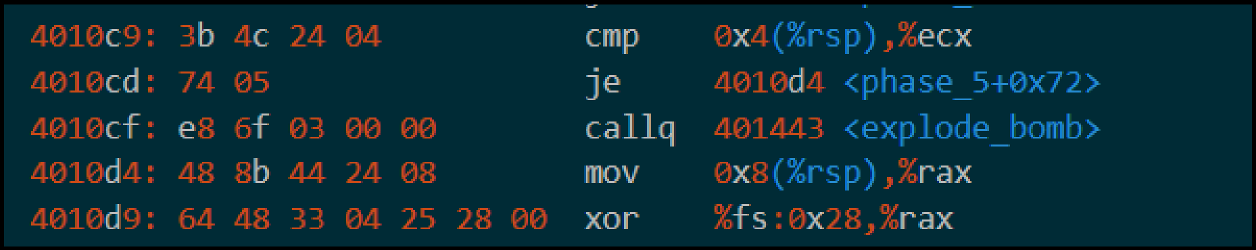


```
4010a0: b9 00 00 00 00          mov    $0x0,%ecx
4010a5: ba 00 00 00 00          mov    $0x0,%edx
4010aa: 83 c2 01                add    $0x1,%edx
4010ad: 48 98                   cltq
4010af: 8b 04 85 80 24 40 00    mov    0x402480(,%rax,4),%eax
4010b6: 01 c1                   add    %eax,%ecx
4010b8: 83 f8 0f                cmp    $0xf,%eax
4010bb: 75 ed                   jne    4010aa <phase_5+0x48>
4010bd: c7 04 24 0f 00 00 00    movl   $0xf,(%rsp)
```

> In the above set of lines there is a loop running until eax is not equal to 15 and I must make sure that our edx should also be 15 or else the bomb will explode

> So, I’ve tried different values of first argument in key and when I kept 5 as first int in the key the gdb

have surpassed the explode function, like this I got to know the first number is 5.



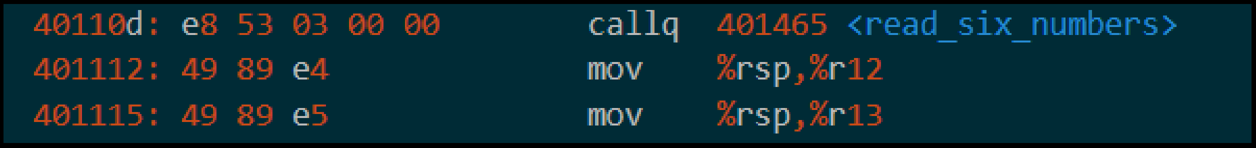
```
4010c9: 3b 4c 24 04          cmp    0x4(%rsp),%ecx
4010cd: 74 05                je     4010d4 <phase_5+0x72>
4010cf: e8 6f 03 00 00       callq  401443 <explode_bomb>
4010d4: 48 8b 44 24 08       mov    0x8(%rsp),%rax
4010d9: 64 48 33 04 25 28 00 xor    %fs:0x28,%rax
```

> In the above code it is comparing my second number with the value in ecx register So, I’ve used ‘info r’ to find what is the value stored there

> Like this I got to know my second number of key is 115.

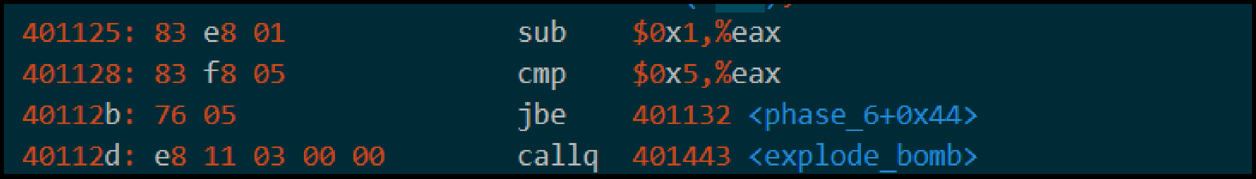
>Hence the final key of phase 5=”5 115”

6) PHASE 6



```
40110d: e8 53 03 00 00          callq  401465 <read_six_numbers>
401112: 49 89 e4                mov    %rsp,%r12
401115: 49 89 e5                mov    %rsp,%r13
```

>From this we conclude there will be an input of 6 integers.



```
401125: 83 e8 01               sub    $0x1,%eax
401128: 83 f8 05               cmp    $0x5,%eax
40112b: 76 05                  jbe    401132 <phase_6+0x44>
40112d: e8 11 03 00 00         callq  401443 <explode_bomb>
```

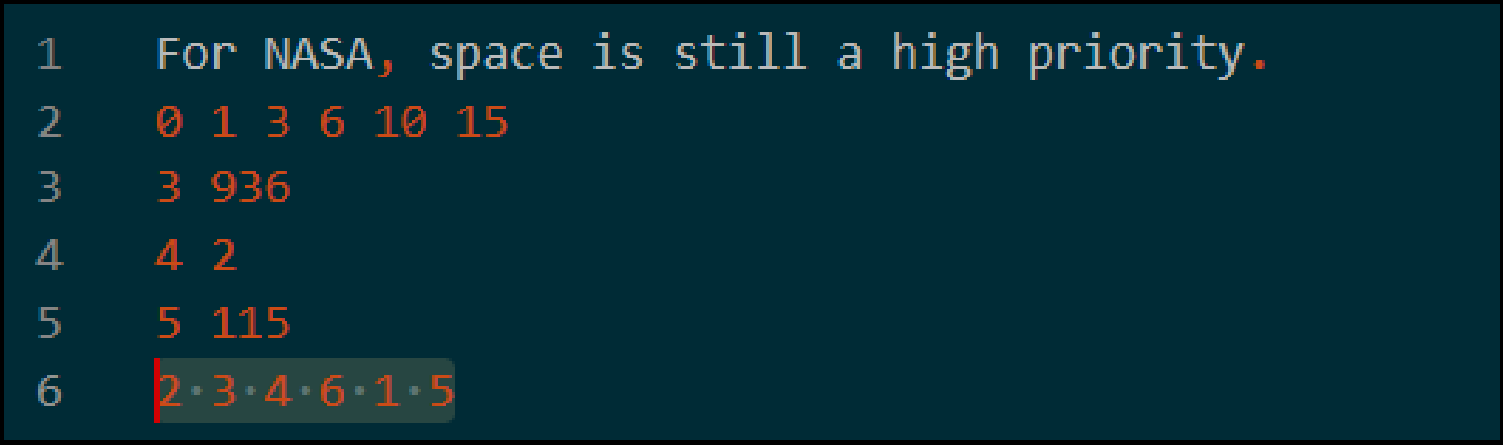
>The numbers will be between 1 to 6.

>From code we can conclude the numbers need to be unique.

>After further solving and checking the value of registers we can find each integers.

>Hence the final key i.e of phase_6 =”2 3 4 561 5”

ALL KEYS:-



```
For NASA, space is still a high priority.
0 1 3 6 10 15
3 936
4 2
5 115
2 3 4 6 1 5
```

THANK YOU