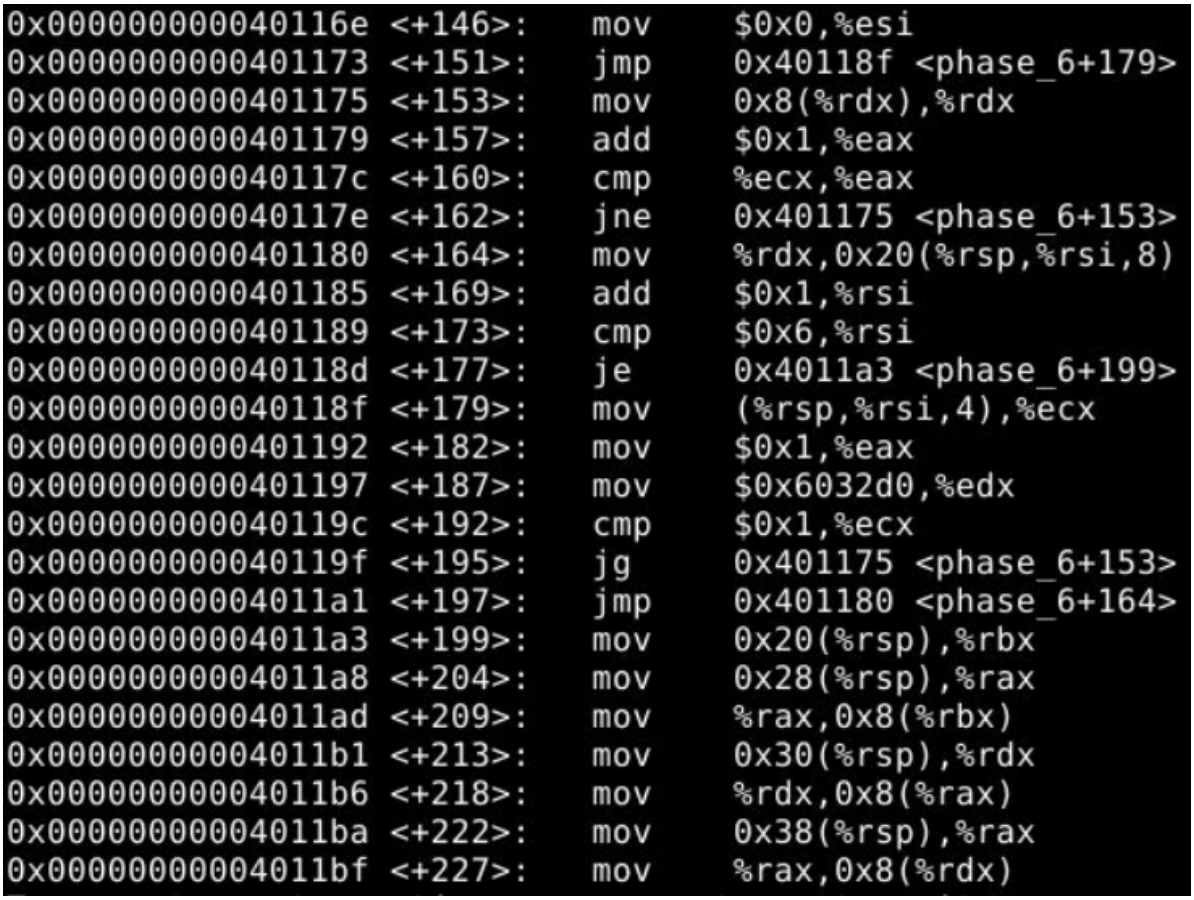
反汇编：

```
0x00000000004010dc <+0>:     push   %r14
0x00000000004010de <+2>:     push   %r13
0x00000000004010e0 <+4>:     push   %r12
0x00000000004010e2 <+6>:     push   %rbp
0x00000000004010e3 <+7>:     push   %rbx
0x00000000004010e4 <+8>:     sub    $0x60,%rsp
0x00000000004010e8 <+12>:    mov    %fs:0x28,%rax
0x00000000004010f1 <+21>:    mov    %rax,0x58(%rsp)
0x00000000004010f6 <+26>:    xor    %eax,%eax
0x00000000004010f8 <+28>:    mov    %rsp,%rsi
0x00000000004010fb <+31>:    callq  0x401469 <read_six_numbers>
0x0000000000401100 <+36>:    mov    %rsp,%r12
0x0000000000401103 <+39>:    mov    %rsp,%r13
0x0000000000401106 <+42>:    mov    $0x0,%r14d
0x000000000040110c <+48>:    jmp    0x401133 <phase_6+87>
0x000000000040110e <+50>:    callq  0x401447 <explode_bomb>
0x0000000000401113 <+55>:    jmp    0x401142 <phase_6+102>
0x0000000000401115 <+57>:    add    $0x1,%ebx
0x0000000000401118 <+60>:    cmp    $0x5,%ebx
0x000000000040111b <+63>:    jg     0x40112f <phase_6+83>
0x000000000040111d <+65>:    movslq %ebx,%rax
0x0000000000401120 <+68>:    mov    (%rsp,%rax,4),%eax
```

```
0x0000000000401123 <+71>:    cmp    %eax,0x0(%rbp)
0x0000000000401126 <+74>:    jne    0x401115 <phase_6+57>
0x0000000000401128 <+76>:    callq  0x401447 <explode_bomb>
0x000000000040112d <+81>:    jmp    0x401115 <phase_6+57>
0x000000000040112f <+83>:    add    $0x4,%r13
0x0000000000401133 <+87>:    mov    %r13,%rbp
0x0000000000401136 <+90>:    mov    0x0(%r13),%eax
0x000000000040113a <+94>:    sub    $0x1,%eax
0x000000000040113d <+97>:    cmp    $0x5,%eax
0x0000000000401140 <+100>:   ja     0x40110e <phase_6+50>
0x0000000000401142 <+102>:   add    $0x1,%r14d
0x0000000000401146 <+106>:   cmp    $0x6,%r14d
0x000000000040114a <+110>:   je     0x401151 <phase_6+117>
0x000000000040114c <+112>:   mov    %r14d,%ebx
0x000000000040114f <+115>:   jmp    0x40111d <phase_6+65>
0x0000000000401151 <+117>:   lea    0x18(%r12),%rcx
0x0000000000401156 <+122>:   mov    $0x7,%edx
0x000000000040115b <+127>:   mov    %edx,%eax
0x000000000040115d <+129>:   sub    (%r12),%eax
0x0000000000401161 <+133>:   mov    %eax,(%r12)
0x0000000000401165 <+137>:   add    $0x4,%r12
0x0000000000401169 <+141>:   cmp    %r12,%rcx
0x000000000040116c <+144>:   jne    0x40115b <phase_6+127>
```



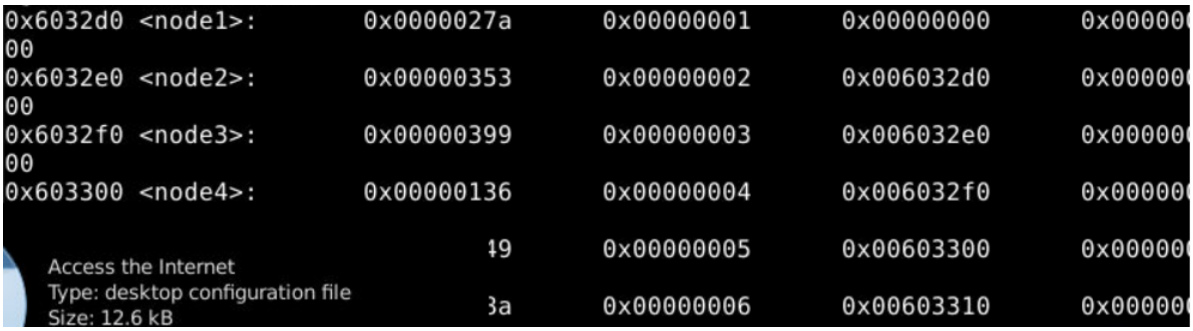
```
0x000000000040116e <+146>:    mov    $0x0,%esi
0x0000000000401173 <+151>:    jmp    0x40118f <phase_6+179>
0x0000000000401175 <+153>:    mov    0x8(%rdx),%rdx
0x0000000000401179 <+157>:    add    $0x1,%eax
0x000000000040117c <+160>:    cmp    %ecx,%eax
0x000000000040117e <+162>:    jne    0x401175 <phase_6+153>
0x0000000000401180 <+164>:    mov    %rdx,0x20(%rsp,%rsi,8)
0x0000000000401185 <+169>:    add    $0x1,%rsi
0x0000000000401189 <+173>:    cmp    $0x6,%rsi
0x000000000040118d <+177>:    je     0x4011a3 <phase_6+199>
0x000000000040118f <+179>:    mov    (%rsp,%rsi,4),%ecx
0x0000000000401192 <+182>:    mov    $0x1,%eax
0x0000000000401197 <+187>:    mov    $0x6032d0,%edx
0x000000000040119c <+192>:    cmp    $0x1,%ecx
0x000000000040119f <+195>:    jg     0x401175 <phase_6+153>
0x00000000004011a1 <+197>:    jmp    0x401180 <phase_6+164>
0x00000000004011a3 <+199>:    mov    0x20(%rsp),%rbx
0x00000000004011a8 <+204>:    mov    0x28(%rsp),%rax
0x00000000004011ad <+209>:    mov    %rax,0x8(%rbx)
0x00000000004011b1 <+213>:    mov    0x30(%rsp),%rdx
0x00000000004011b6 <+218>:    mov    %rdx,0x8(%rax)
0x00000000004011ba <+222>:    mov    0x38(%rsp),%rax
0x00000000004011bf <+227>:    mov    %rax,0x8(%rdx)
```

```
rsp -= 0x60
eax = 0
rsi = rsp
call <read_six_numbers>      // 读入6个数进来
r12 = rsp                    // r12是第一个数的地址
r13 = rsp
r14d = 0x0

1:rbp = r13
eax = *r13
eax -= 0x1
if(eax > 0x5){       // 用的ja无符号比较，所以也能排除负数 要求eax <= 5
    call <explode_bomb>
}
r14d += 0x1          // r14d用来计数，r13用来存rsp数组中的数的地址；eax为当前数组中的数-1
if(r14d == 0x6){     // 扫到最后一个数
    rcx = r12 + 0x18
    edx = 0x7            // 把所有数转化成7-x
    do {
        eax = edx
        eax -= *r12
        *(r12) = eax
        r12 += 0x4
    }while(rcx != r12);
    esi = 0x0;
```

```
    while(1){
        ecx = *(rsp + 4 * rsi)
        eax = 0x1
        edx = 0x6032d0
        if(ecx > 0x1){
            do{
                rdx = *(rdx + 0x8)
                eax += 0x1
            }while(ecx != eax)
        }
        *(rsp + 8 * rsi + 0x20) = rdx
        rsi += 0x1
        if(rsi == 0x6){
            rbx = *(rsp + 0x20)
                rax = *(rsp + 0x28)
                *(rbx + 0x8) = rax
                rdx = *(rsp + 0x30)
                *(rax + 0x8) = rdx
                rdx = *(rsp + 0x40)
                *(rax + 0x8) = rdx
                rax = *(rsp + 0x48)
                *(rdx + 0x8) = rax
                *(rax + 0x8) = 0x0
                ebp = 0x5

                2:rax = *(rbx + 0x8)
                    eax = *(rax)
                    if(*rbx >= eax){
                        rbx = *(rbx + 0x8)
                            ebp -= 0x1
                            if(ebp == 0x0){
                                retq
                            }
                        else {
                            goto 2;
                        }
                    }
                    else {
                        call <explode_bomb>
                    }
        }
    }
}
else {  // rbp依次放的是第1，2，3，4，5个数；这段代码用于检验6个数互不相同，否则bomb
    ebx = r14d
    while(1){
        rax = ebx
```

```
        eax = *(rsp + 4 * rax)
        if(*rbp != eax){
            ebx += 0x1
            if(ebx > 0x5){
                r13 += 0x4
                goto 1;
                break;
            }
        }
        else {
            call <explode_bomb>
        }
    }
}
```

0x6032d0是一个链表1-6



输入的一串数将第三列的顺序链表重新按输入输入接上，最后是让按着重组后的链表顺序使得对应的第一列数从大到小排，所以应该是3 2 1 5 4 6；由于有一个7-x，所以答案是4 5 6 2 3 1