```
akhilesh@akhilesh-VirtualBox:~/Desktop/422104/week7$ gcc -g sum.c -o sum
akhilesh@akhilesh-VirtualBox:~/Desktop/422104/week7$ ./sum
0
1
2
3
4
5
6
7
Segmentation fault (core dumped)
akhilesh@akhilesh-VirtualBox:~/Desktop/422104/week7$ gdb sum
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 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 sum...
(gdb) run
Starting program: /home/akhilesh/Desktop/422104/week7/sum
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
0
1
2
3
4
5
6
7

Program received signal SIGSEGV, Segmentation fault.
0x0000555555555196 in main () at sum.c:12
12          printf("%d\n", *ptr);
(gdb) list
7               printf("%d\n", i);
8           }
9
10
11          int *ptr = NULL;
12          printf("%d\n", *ptr);
13
```

```
13
14
15
16          return 0;
(gdb)
17      }
18
19
(gdb) ru
The program being debugged has been started already.
Start it from the beginning? (y or n)
Please answer y or n.
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/akhilesh/Desktop/422104/week7/sum
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
0
1
2
3
4
5
6
7

Program received signal SIGSEGV, Segmentation fault.
0x0000555555555196 in main () at sum.c:12
12          printf("%d\n", *ptr);
(gdb) break main
Breakpoint 1 at 0x555555555155: file sum.c, line 5.
(gdb) break 11
Breakpoint 2 at 0x55555555518a: file sum.c, line 11.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/akhilesh/Desktop/422104/week7/sum
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at sum.c:5
5           int n = 8;
(gdb) print n
$1 = 0
(gdb) next
6           for(int i=0; i<n; i++){
(gdb) print n
$2 = 8
```

```
(gdb) next
7               printf("%d\n", i);
(gdb) next

0
6           for(int i=0; i<n; i++){
(gdb) next
7               printf("%d\n", i);
(gdb) print i
$3 = 1
(gdb) continue
Continuing.
1
2
3
4
5
6
7

Breakpoint 2, main () at sum.c:11
11          int *ptr = NULL;
(gdb) next
12          printf("%d\n", *ptr);
(gdb) next

Program received signal SIGSEGV, Segmentation fault.
0x0000555555555196 in main () at sum.c:12
12          printf("%d\n", *ptr);
(gdb) next

Program terminated with signal SIGSEGV, Segmentation fault.
The program no longer exists.
(gdb) disassemble
No frame selected.
(gdb) run
Starting program: /home/akhilesh/Desktop/422104/week7/sum
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at sum.c:5
5           int n = 8;
```

```
(gdb) disassemble
No frame selected.
(gdb) run
Starting program: /home/akhilesh/Desktop/422104/week7/sum
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at sum.c:5
5           int n = 8;
(gdb) disassemble
Dump of assembler code for function main:
   0x0000555555555149 <+0>:     endbr64
   0x000055555555514d <+4>:     push   %rbp
   0x000055555555514e <+5>:     mov    %rsp,%rbp
   0x0000555555555151 <+8>:     sub    $0x10,%rsp
=> 0x0000555555555155 <+12>:    movl   $0x8,-0xc(%rbp)
   0x000055555555515c <+19>:    movl   $0x0,-0x10(%rbp)
   0x0000555555555163 <+26>:    jmp    0x555555555182 <main+57>
   0x0000555555555165 <+28>:    mov    -0x10(%rbp),%eax
   0x0000555555555168 <+31>:    mov    %eax,%esi
   0x000055555555516a <+33>:    lea    0xe93(%rip),%rax        # 0x555555556004
   0x0000555555555171 <+40>:    mov    %rax,%rdi
   0x0000555555555174 <+43>:    mov    $0x0,%eax
   0x0000555555555179 <+48>:    call   0x555555555050 <printf@plt>
   0x000055555555517e <+53>:    addl   $0x1,-0x10(%rbp)
   0x0000555555555182 <+57>:    mov    -0x10(%rbp),%eax
   0x0000555555555185 <+60>:    cmp    -0xc(%rbp),%eax
   0x0000555555555188 <+63>:    jl     0x555555555165 <main+28>
   0x000055555555518a <+65>:    movq   $0x0,-0x8(%rbp)
   0x0000555555555192 <+73>:    mov    -0x8(%rbp),%rax
   0x0000555555555196 <+77>:    mov    (%rax),%eax
   0x0000555555555198 <+79>:    mov    %eax,%esi
   0x000055555555519a <+81>:    lea    0xe63(%rip),%rax        # 0x555555556004
   0x00005555555551a1 <+88>:    mov    %rax,%rdi
   0x00005555555551a4 <+91>:    mov    $0x0,%eax
   0x00005555555551a9 <+96>:    call   0x555555555050 <printf@plt>
   0x00005555555551ae <+101>:   mov    $0x0,%eax
   0x00005555555551b3 <+106>:   leave
   0x00005555555551b4 <+107>:   ret
End of assembler dump.
```