Valgrind

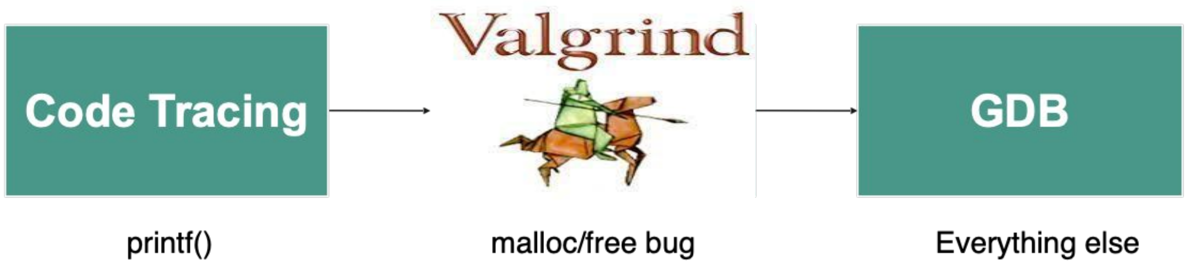
Fall 2024 - COMP 201

Lab 4

KOÇ UNIVERSITY

What is Valgrind?

- An open source system memory debugger
- Used for memory leak detection and profiling



How to use?

```
$ gcc -g -o out sample.c
```

-g	Enabling the Valgrind
out	Output file
sample.c	The program for compile

- Using -O0 is also a good idea!

- Valgrind usage:

  ```
  $ valgrind ./out
  ```

  `$ man valgrind` to play around with options

Errors that Valgrind can detect and report:

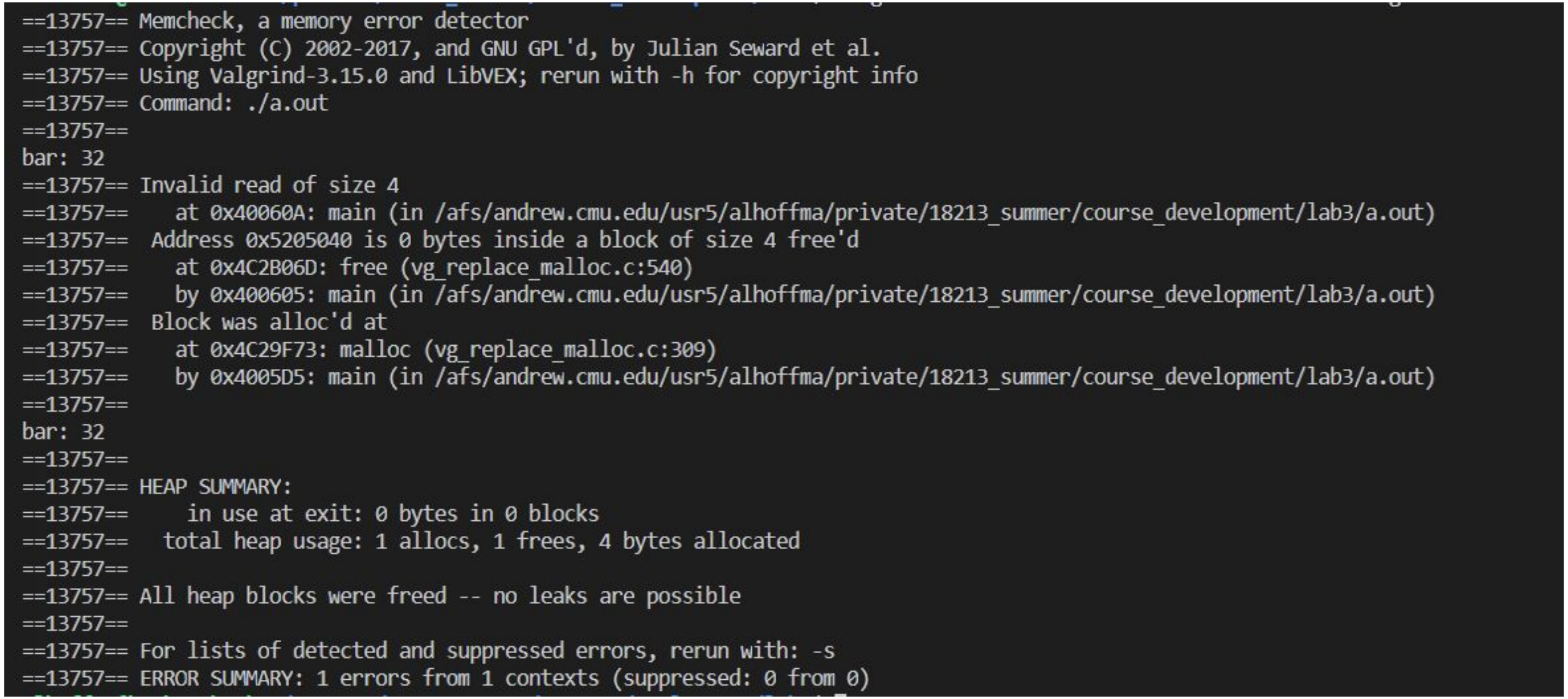
- **Invalid read/write errors**
  - Reads or writes to a memory address which you did not allocate
- **Use of an uninitialized value**
  - Code uses a declared variable before any kind of explicit assignment
- **Invalid free error**
  - Code attempts to delete allocated memory twice
  - Delete memory that was not allocated

Invalid read & writes

- Reading freed variables
- Reading uninitialized variables
- Writing to uninitialized memory
  - By writing too much data to allocated memory

```
int foo (int y) {

int *bar =malloc(sizeof(int));
*bar = y;

free(bar)

printf(“bar: %d \n”, * bar);
return y;

}
```

Invalid read & writes



```
==13757== Memcheck, a memory error detector
==13757== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==13757== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==13757== Command: ./a.out
==13757==
bar: 32
==13757== Invalid read of size 4
==13757==    at 0x40060A: main (in /afs/andrew.cmu.edu/usr5/alhoffma/private/18213_summer/course_development/lab3/a.out)
==13757==  Address 0x5205040 is 0 bytes inside a block of size 4 free'd
==13757==    at 0x4C2B06D: free (vg_replace_malloc.c:540)
==13757==    by 0x400605: main (in /afs/andrew.cmu.edu/usr5/alhoffma/private/18213_summer/course_development/lab3/a.out)
==13757==  Block was alloc'd at
==13757==    at 0x4C29F73: malloc (vg_replace_malloc.c:309)
==13757==    by 0x4005D5: main (in /afs/andrew.cmu.edu/usr5/alhoffma/private/18213_summer/course_development/lab3/a.out)
==13757==
bar: 32
==13757==
==13757== HEAP SUMMARY:
==13757==     in use at exit: 0 bytes in 0 blocks
==13757==   total heap usage: 1 allocs, 1 frees, 4 bytes allocated
==13757==
==13757== All heap blocks were freed -- no leaks are possible
==13757==
==13757== For lists of detected and suppressed errors, rerun with: -s
==13757== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

Memory Errors Vs. Memory Leaks

- **Memory leaks:**
  - A program dynamically allocates memory and does not free it
  - Won't cause a program to misbehave, crash, or give wrong answers

- **Memory errors:**
  - Is a red alert.
  - Reading uninitialized memory
  - Writing past the end of a piece of memory,
  - Accessing freed memory, etc
  - Can have significant consequences.
  - Memory errors should never be treated casually or ignored

Types of Memory Leaks

- Still Reachable
  - Memory plock is still pointed at, programmer could go back and free it before exiting

- Indirectly Lost
  - Block is lost because the blocks that point to it are themselves lost

- Definitely Lost
  - No pointer to the block can be found

- Possibly Lost
  - Pointer exists but it points to an internal part of the memory block

Memory Leaks

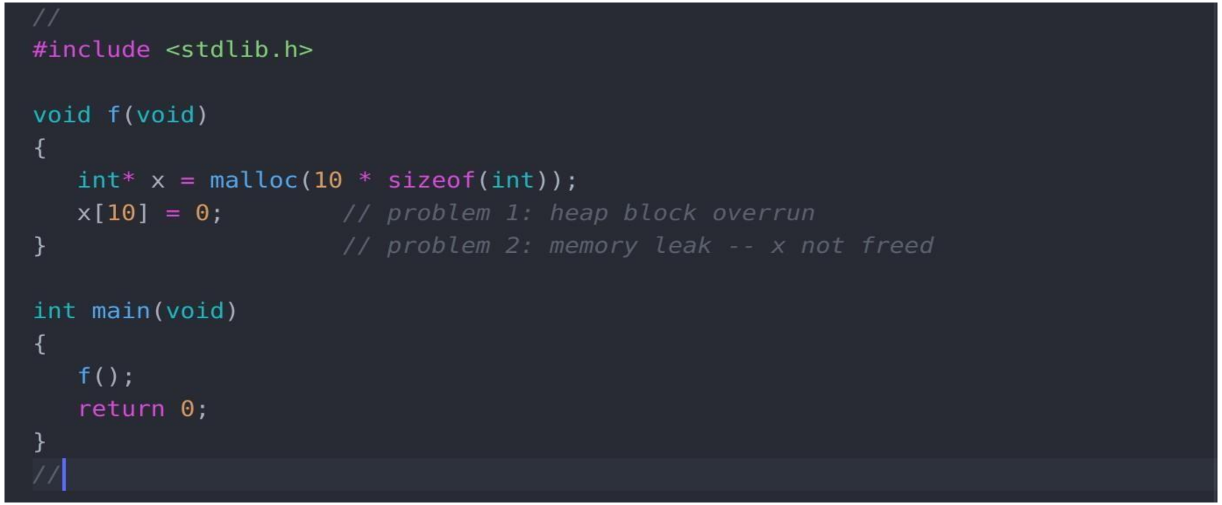
- Memory that is allocated should always be freed

```
int foo (int y) {

int *bar =malloc(sizeof(int));
*bar = y;

printf(“bar: %d \n”, * bar);
return y;

}
```

Example: sample.c

- With a memory error and a memory leak

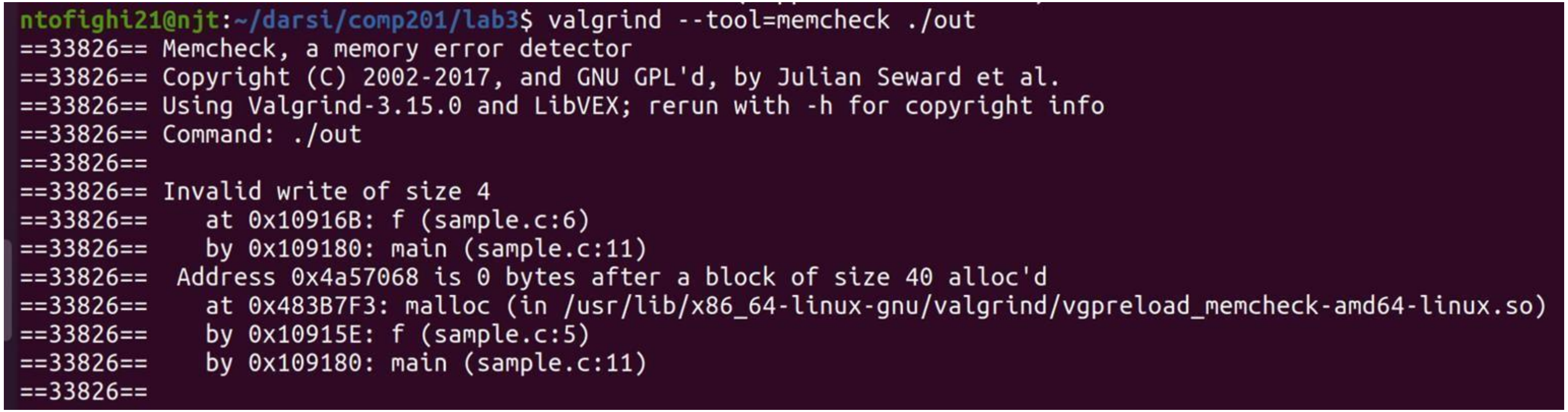
```
$ gcc -g -o out sample.c
$ valgrind ./out
```



```
//
#include <stdlib.h>

void f(void)
{
   int* x = malloc(10 * sizeof(int));
   x[10] = 0;        // problem 1: heap block overrun
}                    // problem 2: memory leak -- x not freed

int main(void)
{
   f();
   return 0;
}
//
```

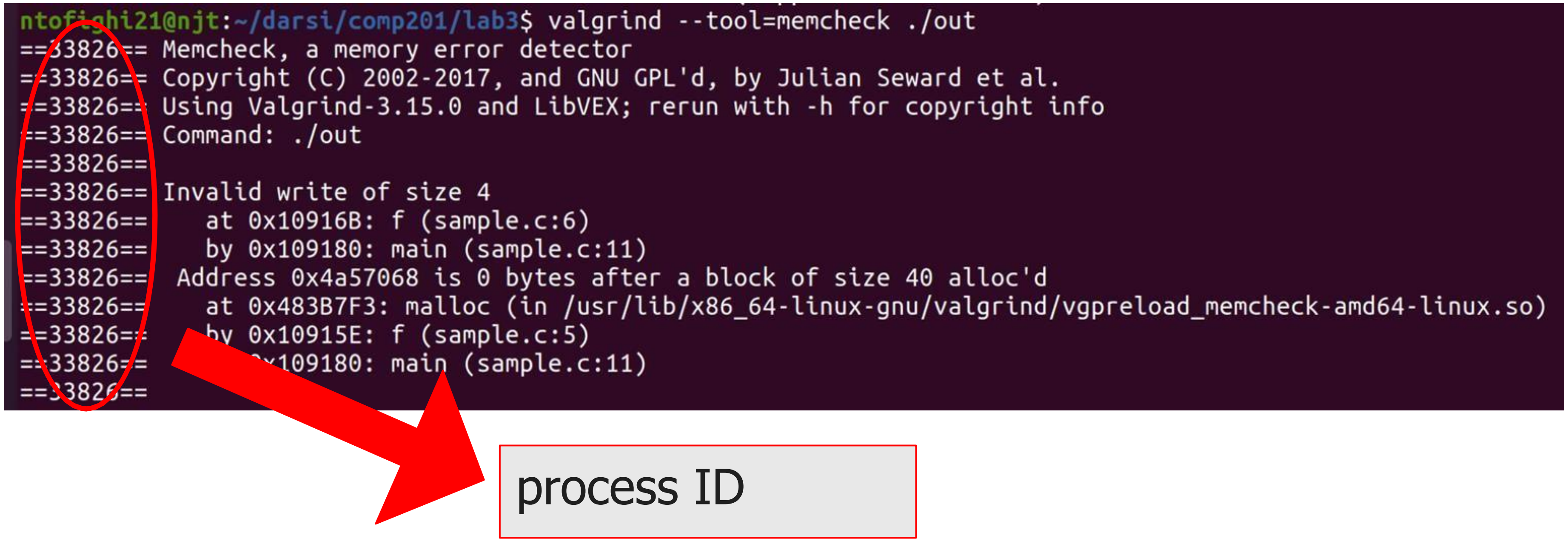
Memory error

- `valgrind --tool=memcheck ./out`



```
ntofighi21@njt:~/darsi/comp201/lab3$ valgrind --tool=memcheck ./out
==33826== Memcheck, a memory error detector
==33826== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==33826== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==33826== Command: ./out
==33826==
==33826== Invalid write of size 4
==33826==    at 0x10916B: f (sample.c:6)
==33826==    by 0x109180: main (sample.c:11)
==33826==  Address 0x4a57068 is 0 bytes after a block of size 40 alloc'd
==33826==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==33826==    by 0x10915E: f (sample.c:5)
==33826==    by 0x109180: main (sample.c:11)
==33826==
```

- **valgrind --tool=memcheck ./out**



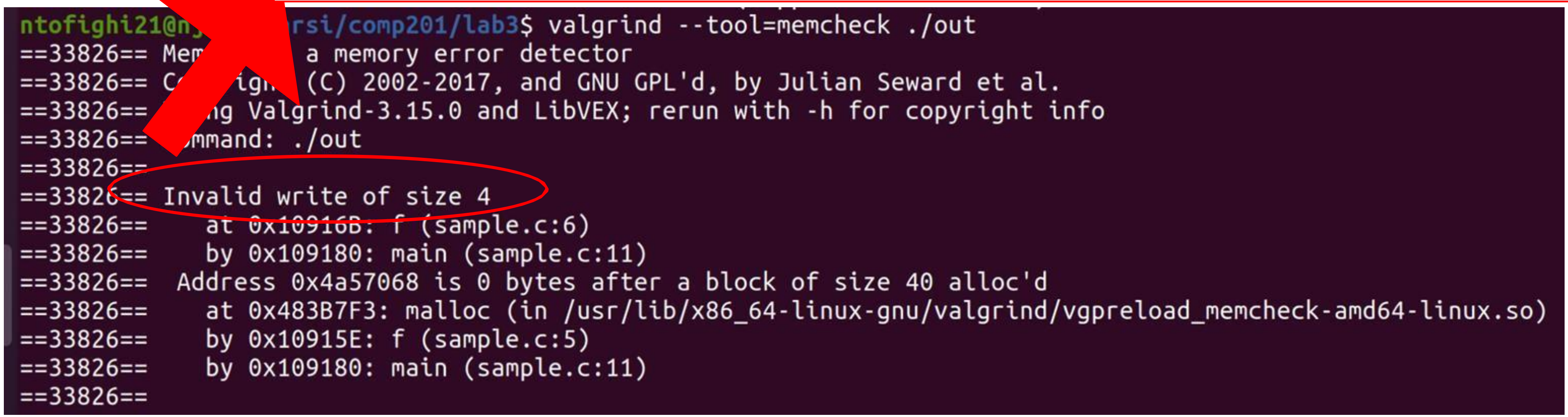
```
ntofighi21@njt:~/darsi/comp201/lab3$ valgrind --tool=memcheck ./out
==33826== Memcheck, a memory error detector
==33826== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==33826== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==33826== Command: ./out
==33826==
==33826== Invalid write of size 4
==33826==    at 0x10916B: f (sample.c:6)
==33826==    by 0x109180: main (sample.c:11)
==33826==  Address 0x4a57068 is 0 bytes after a block of size 40 alloc'd
==33826==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==33826==    by 0x10915E: f (sample.c:5)
==33826==    by 0x109180: main (sample.c:11)
==33826==
```

process ID

- **valgrind --tool=memcheck ./out**

Types of error

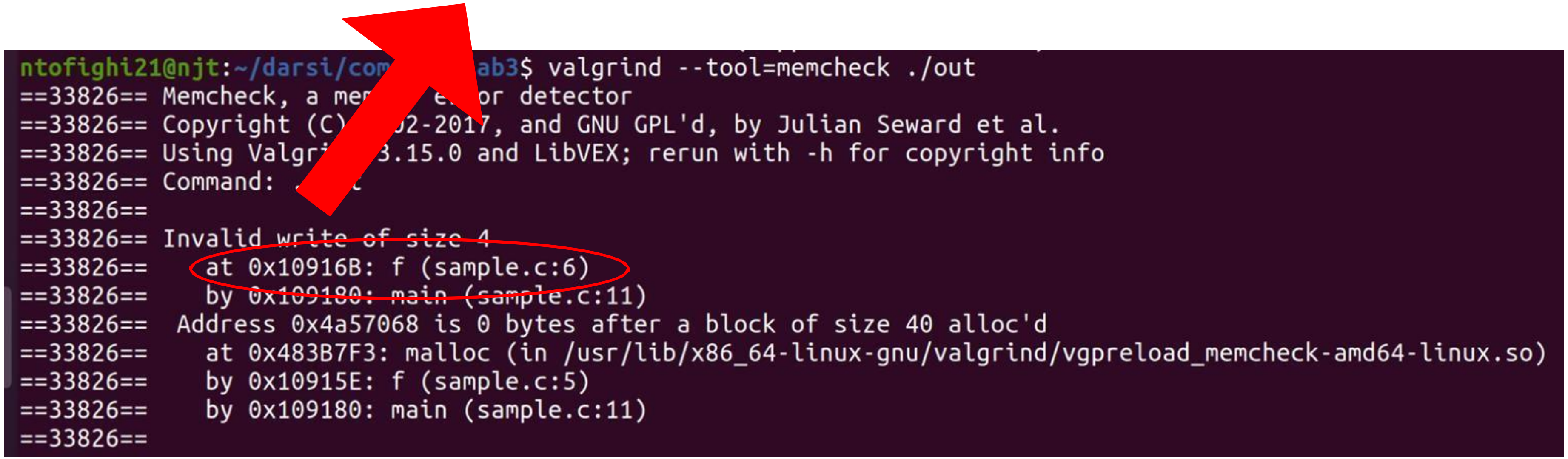
Here; The program wrote to some memory it should not have due to a heap block overrun.



```
ntofighi21@n[illegible]rsi/comp201/lab3$ valgrind --tool=memcheck ./out
==33826== Mem[illegible] a memory error detector
==33826== C[illegible]igh[illegible] (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==33826== [illegible]ng Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==33826== [illegible]mmand: ./out
==33826==
==33826== Invalid write of size 4
==33826==    at 0x10916B: f (sample.c:6)
==33826==    by 0x109180: main (sample.c:11)
==33826==  Address 0x4a57068 is 0 bytes after a block of size 40 alloc'd
==33826==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==33826==    by 0x10915E: f (sample.c:5)
==33826==    by 0x109180: main (sample.c:11)
==33826==
```

- ➢ **valgrind --tool=memcheck ./out**

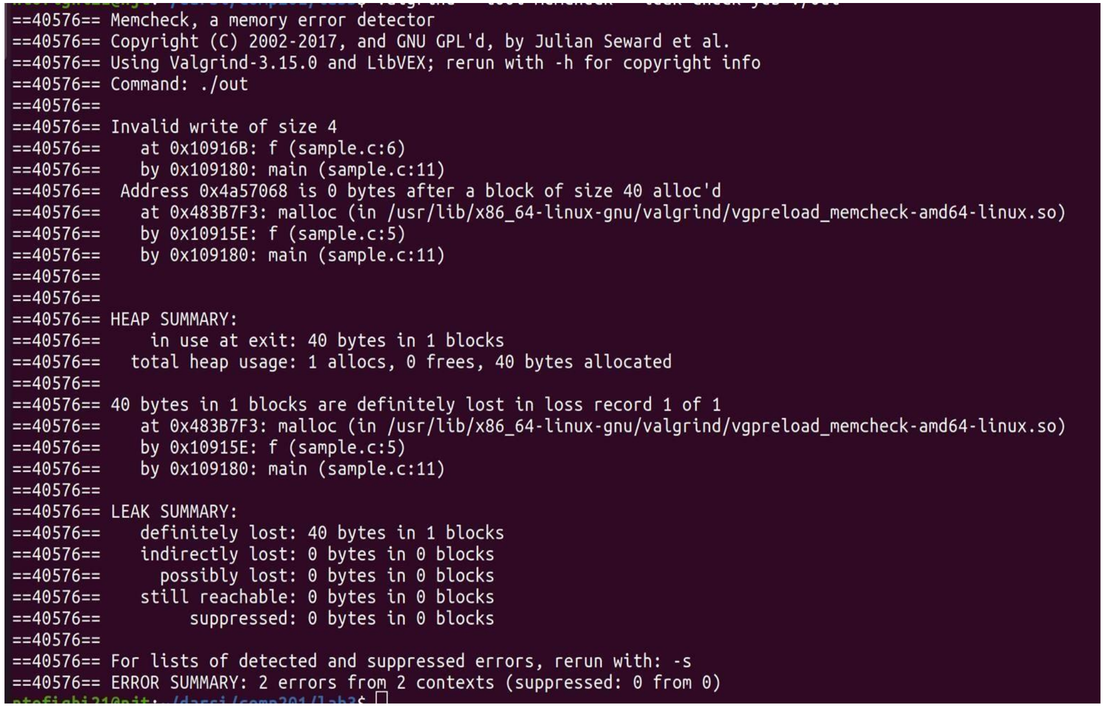
Stack trace → where the problem occurred.



```
ntofighi21@njt:~/darsi/com      ab3$ valgrind --tool=memcheck ./out
==33826== Memcheck, a mem    e  or detector
==33826== Copyright (C)  02-2017, and GNU GPL'd, by Julian Seward et al.
==33826== Using Valgri  3.15.0 and LibVEX; rerun with -h for copyright info
==33826== Command: .  t
==33826==
==33826== Invalid write of size 4
==33826==    at 0x10916B: f (sample.c:6)
==33826==    by 0x109180: main (sample.c:11)
==33826==  Address 0x4a57068 is 0 bytes after a block of size 40 alloc'd
==33826==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==33826==    by 0x10915E: f (sample.c:5)
==33826==    by 0x109180: main (sample.c:11)
==33826==
```

Memory error

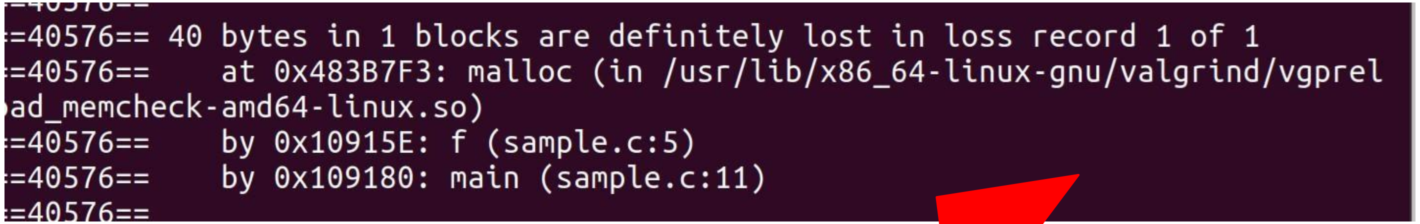
➢ **`valgrind --tool=memcheck --leak-check=yes ./out`**



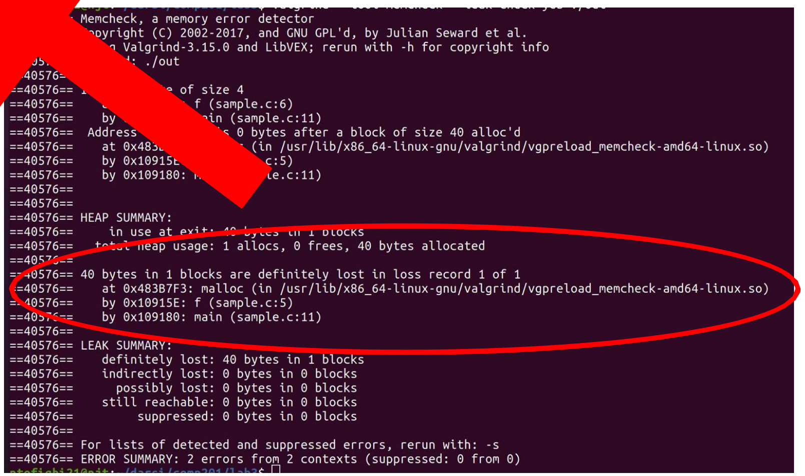
```
==40576== Memcheck, a memory error detector
==40576== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==40576== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==40576== Command: ./out
==40576==
==40576== Invalid write of size 4
==40576==    at 0x10916B: f (sample.c:6)
==40576==    by 0x109180: main (sample.c:11)
==40576==  Address 0x4a57068 is 0 bytes after a block of size 40 alloc'd
==40576==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==40576==    by 0x10915E: f (sample.c:5)
==40576==    by 0x109180: main (sample.c:11)
==40576==
==40576==
==40576== HEAP SUMMARY:
==40576==     in use at exit: 40 bytes in 1 blocks
==40576==   total heap usage: 1 allocs, 0 frees, 40 bytes allocated
==40576==
==40576== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
==40576==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==40576==    by 0x10915E: f (sample.c:5)
==40576==    by 0x109180: main (sample.c:11)
==40576==
==40576== LEAK SUMMARY:
==40576==    definitely lost: 40 bytes in 1 blocks
==40576==    indirectly lost: 0 bytes in 0 blocks
==40576==      possibly lost: 0 bytes in 0 blocks
==40576==    still reachable: 0 bytes in 0 blocks
==40576==         suppressed: 0 bytes in 0 blocks
==40576==
==40576== For lists of detected and suppressed errors, rerun with: -s
==40576== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
```

Memory error

➢ `valgrind --tool=memcheck --leak-check=yes ./out`



```
=40576== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
=40576==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgprel
ad_memcheck-amd64-linux.so)
=40576==    by 0x10915E: f (sample.c:5)
=40576==    by 0x109180: main (sample.c:11)
=40576==
```



```
  Memcheck, a memory error detector
  pyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
  g Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
  d: ./out
40576=
==40576== I       e of size 4
==40576==    a       f (sample.c:6)
==40576==    by       in (sample.c:11)
==40576==  Address       s 0 bytes after a block of size 40 alloc'd
==40576==    at 0x483       (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==40576==    by 0x10915E      c:5)
==40576==    by 0x109180: m      le.c:11)
==40576==
==40576==
==40576== HEAP SUMMARY:
==40576==     in use at exit: 40 bytes in 1 blocks
==40576==   total heap usage: 1 allocs, 0 frees, 40 bytes allocated
==40576==
==40576== 40 bytes in 1 blocks are definitely lost in loss record 1 of 1
==40576==    at 0x483B7F3: malloc (in /usr/lib/x86_64-linux-gnu/valgrind/vgpreload_memcheck-amd64-linux.so)
==40576==    by 0x10915E: f (sample.c:5)
==40576==    by 0x109180: main (sample.c:11)
==40576==
==40576== LEAK SUMMARY:
==40576==    definitely lost: 40 bytes in 1 blocks
==40576==    indirectly lost: 0 bytes in 0 blocks
==40576==      possibly lost: 0 bytes in 0 blocks
==40576==    still reachable: 0 bytes in 0 blocks
==40576==         suppressed: 0 bytes in 0 blocks
==40576==
==40576== For lists of detected and suppressed errors, rerun with: -s
==40576== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
```

Useful links

- Valgrind and GDB in close cooperation
- Valgrind User Manual

Now In-Lab Exercise!