

jithu@DESKTOP-BGRVDL1: ~

```
0x000055555555562b <+48>: mov    $0x0,%eax
0x0000555555555630 <+53>: call   0x5555555555f3 <test5>
0x0000555555555635 <+58>: mov    $0x0,%eax
0x000055555555563a <+63>: pop     %rbp
0x000055555555563b <+64>: ret
```

End of assembler dump.

(gdb) disassemble test1

Dump of assembler code for function test1:

```
0x00005555555551e9 <+0>:   endbr64
0x00005555555551ed <+4>:   push   %rbp
0x00005555555551ee <+5>:   mov     %rsp,%rbp
0x00005555555551f1 <+8>:   sub     $0x10,%rsp
0x00005555555551f5 <+12>:  mov     $0x190,%edi
0x00005555555551fa <+17>:  call    0x5555555550d0 <malloc@plt>
0x00005555555551ff <+22>:  mov     %rax,-0x8(%rbp)
0x0000555555555203 <+26>:  cmpq    $0x0,-0x8(%rbp)
0x0000555555555208 <+31>:  jne     0x55555555521b <test1+50>
0x000055555555520a <+33>:  lea     0xdf3(%rip),%rax      # 0x5555555556004
0x0000555555555211 <+40>:  mov     %rax,%rdi
0x0000555555555214 <+43>:  call    0x5555555550e0 <perror@plt>
0x0000555555555219 <+48>:  jmp     0x555555555273 <test1+138>
0x000055555555521b <+50>:  movl    $0x0,-0xc(%rbp)
0x0000555555555222 <+57>:  jmp     0x555555555241 <test1+88>
0x0000555555555224 <+59>:  mov     -0xc(%rbp),%eax
0x0000555555555227 <+62>:  cltq
0x0000555555555229 <+64>:  lea     0x0(%rax,4),%rdx
0x0000555555555231 <+72>:  mov     -0x8(%rbp),%rax
0x0000555555555235 <+76>:  add     %rax,%rdx
0x0000555555555238 <+79>:  mov     -0xc(%rbp),%eax
0x000055555555523b <+82>:  mov     %eax,(%rdx)
0x000055555555523d <+84>:  addl    $0x1,-0xc(%rbp)
0x0000555555555241 <+88>:  cmpl    $0x63,-0xc(%rbp)
0x0000555555555245 <+92>:  jle     0x555555555224 <test1+50>
0x0000555555555247 <+94>:  mov     -0x8(%rbp),%rax
0x000055555555524b <+98>:  add     $0x28,%rax
0x000055555555524f <+102>: mov     (%rax),%eax
0x0000555555555251 <+104>: mov     %eax,%esi
0x0000555555555253 <+106>: lea     0xdl4(%rip),%rax      # 0x555555555601e
0x000055555555525a <+113>: mov     %rax,%rdi
0x000055555555525d <+116>: mov     $0x0,%eax
0x0000555555555262 <+121>: call    0x5555555550b0 <printf@plt>
0x0000555555555267 <+126>: mov     -0x8(%rbp),%rax
0x000055555555526b <+130>: mov     %rax,%rdi
0x000055555555526e <+133>: call    0x5555555550a0 <free@plt>
0x0000555555555273 <+138>: leave
0x0000555555555274 <+139>: ret
```

End of assembler dump.

(gdb) info registers

```
rax      0x5555555555fb      93824992237051
rbx      0x0
```

jithu@DESKTOP-BGRVDL1: ~

```
0x000055555555224 <+59>: mov    -0xc(%rbp),%eax
0x000055555555227 <+62>: cllq
0x000055555555229 <+64>: lea    0x0(,%rax,4),%rdx
0x000055555555231 <+72>: mov    -0x8(%rbp),%rax
0x000055555555235 <+76>: add    %rax,%rdx
0x000055555555238 <+79>: mov    -0xc(%rbp),%eax
0x00005555555523b <+82>: mov    %eax,%rdx
0x00005555555523d <+84>: addl   $0x1,-0xc(%rbp)
0x000055555555241 <+88>: cmpl   $0x63,-0xc(%rbp)
0x000055555555245 <+92>: jle     0x55555555224 <test1+59>
0x000055555555247 <+94>: mov    -0x8(%rbp),%rax
0x00005555555524b <+98>: add    $0x28,%rax
0x00005555555524f <+102>: mov    (%rax),%eax
0x000055555555251 <+104>: mov    %eax,%esi
0x000055555555253 <+106>: lea    0x4(%rip),%rax    # 0x55555555601e
0x00005555555525a <+113>: mov    %rax,%rdi
0x00005555555525d <+116>: mov    $0x0,%eax
0x000055555555262 <+121>: call   0x555555550b0 <printf@plt>
0x000055555555267 <+126>: mov    -0x8(%rbp),%rax
0x00005555555526b <+130>: mov    %rax,%rdi
0x00005555555526e <+133>: call   0x555555550a0 <free@plt>
0x000055555555273 <+138>: leave
0x000055555555274 <+139>: ret
```

End of assembler dump.

(gdb) info registers

rax	0x55555555fb	93824992237051
rbx	0x0	0
rcx	0x55555557d98	93824992247192
rdx	0x7fffffffefc8	140737488347336
rsi	0x7fffffffefb8	140737488347320
rdi	0x1	1
rbp	0x7fffffffdfa0	0x7fffffffdfa0
rsp	0x7fffffffdfa0	0x7fffffffdfa0
r8	0x7ffff7fa3f10	140737353760528
r9	0x7ffff7fc9040	140737353912384
r10	0x7ffff7fc3908	140737353890056
r11	0x7ffff7fde660	140737353999968
r12	0x7fffffffefb8	140737488347320
r13	0x55555555fb	93824992237051
r14	0x55555557d98	93824992247192
r15	0x7ffff7ffd040	140737354125376
rip	0x55555555603	0x55555555603 <main+8>
eflags	0x246	[ PF ZF IF ]
cs	0x33	51
ss	0x2b	43
ds	0x0	0
es	0x0	0
fs	0x0	0
gs	0x0	0

(gdb)

jithu@DESKTOP-BGRVDL1: ~

```
0x000055555555621 <+38>: mov    $0x0,%eax
0x000055555555626 <+43>: call   0x555555555360 <test4>
0x00005555555562b <+48>: mov    $0x0,%eax
0x000055555555630 <+53>: call   0x5555555553f5 <test5>
0x000055555555635 <+58>: mov    $0x0,%eax
0x00005555555563a <+63>: pop     %rbp
0x00005555555563b <+64>: ret
```

End of assembler dump.

(gdb) disassemble test1

Dump of assembler code for function test1:

```
0x00005555555551e9 <+0>: endbr64
0x00005555555551ed <+4>: push   %rbp
0x00005555555551ee <+5>: mov    %rsp,%rbp
0x00005555555551f1 <+8>: sub    $0x10,%rsp
0x00005555555551f5 <+12>: mov    $0x190,%edi
0x00005555555551fa <+17>: call   0x5555555550d0 <malloc@plt>
0x00005555555551ff <+22>: mov    %rax,-0x8(%rbp)
0x0000555555555203 <+26>: cmpq   $0x0,-0x8(%rbp)
0x0000555555555208 <+31>: jne     0x55555555521b <test1+50>
0x000055555555520a <+33>: lea     0xdf3(%rip),%rax      # 0x555555556004
0x0000555555555211 <+40>: mov     %rax,%rdi
0x0000555555555214 <+43>: call   0x5555555550e0 <permon@plt>
0x0000555555555219 <+48>: jmp     0x555555555273 <test1+138>
0x000055555555521b <+50>: movl    $0x0,-0xc(%rbp)
0x0000555555555222 <+57>: jmp     0x555555555241 <test1+88>
0x0000555555555224 <+59>: mov     -0xc(%rbp),%eax
0x0000555555555227 <+62>: cltq
0x0000555555555229 <+64>: lea     0x0(,%rax,4),%rdx
0x0000555555555231 <+72>: mov     -0x8(%rbp),%rax
0x0000555555555235 <+76>: add     %rax,%rdx
0x0000555555555238 <+79>: mov     -0xc(%rbp),%eax
0x000055555555523b <+82>: mov     %eax,(%rdx)
0x000055555555523d <+84>: addl    $0x1,-0xc(%rbp)
0x0000555555555241 <+88>: cmpl    $0x63,-0xc(%rbp)
0x0000555555555245 <+92>: jle     0x555555555224 <test1+59>
0x0000555555555247 <+94>: mov     -0x8(%rbp),%rax
0x000055555555524b <+98>: add     $0x28,%rax
0x000055555555524f <+102>: mov     (%rax),%eax
0x0000555555555251 <+104>: mov     %eax,%esi
0x0000555555555253 <+106>: lea     0xdc4(%rip),%rax      # 0x55555555601e
0x000055555555525a <+113>: mov     %rax,%rdi
0x000055555555525d <+116>: mov     $0x0,%eax
0x0000555555555262 <+121>: call    0x5555555550b0 <printf@plt>
0x0000555555555267 <+126>: mov     -0x8(%rbp),%rax
0x000055555555526b <+130>: mov     %rax,%rdi
0x000055555555526e <+133>: call    0x5555555550a0 <free@plt>
0x0000555555555273 <+138>: leave
0x0000555555555274 <+139>: ret
```

End of assembler dump.

(gdb)



```

jithu@DESKTOP-BGRVDL1: ~
(gdb)
77     for(int i=0;i<NUM_STRUCTS;i++){
(gdb)
78         free(data[i].name);
(gdb)
79         free(data[i].values);
(gdb)
77     for(int i=0;i<NUM_STRUCTS;i++){
(gdb)
78         free(data[i].name);
(gdb)
79         free(data[i].values);
(gdb)
77     for(int i=0;i<NUM_STRUCTS;i++){
(gdb)
78         free(data[i].name);
(gdb)
79         free(data[i].values);
(gdb)
77     for(int i=0;i<NUM_STRUCTS;i++){
(gdb)
78         free(data[i].name);
(gdb)
79         free(data[i].values);
(gdb)
77     for(int i=0;i<NUM_STRUCTS;i++){
(gdb)
81         free(data);
(gdb)
82     }
(gdb)
main () at grind.c:91
91     return 0;
(gdb)
92 }
(gdb)
__libc_start_call_main (main=main@entry=0x5555555555fb <main>, argc=argc@entry=1, argv=argv@entry=0x7fff
ffffe0b8) at ../sysdeps/nptl/libc_start_call_main.h:74
74     ../sysdeps/nptl/libc_start_call_main.h: No such file or directory.
(gdb)
[Inferior 1 (process 18888) exited normally]
(gdb)
The program is not being run.
(gdb)
The program is not being run.
(gdb)
The program is not being run.
(gdb)
The program is not being run.
(gdb)

```

```

grid - Notepad
File Edit Format View Help
    ptr[i] = i * 2;
}
free(ptr);
}

void test4() {
    int *ptr = malloc(sizeof(int) * 10);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 10; i++) {
        ptr[i] = i * 3;
    }
    free(ptr);
    ptr = NULL;
    printf("Value of *ptr: %d\n", *ptr);
}

void test5() {
    Datastruct *data = malloc(NUM_STRUCTS * sizeof(Datastruct));
    for (int i = 0; i < NUM_STRUCTS; i++) {
        data[i].name = malloc(50 * sizeof(char));
        strcpy(data[i].name, "Example Name");
        data[i].id = rand() % 1000;
        data[i].values = malloc(10 * sizeof(int));
        for (int j = 0; j < 10; j++) {
            data[i].values[j] = rand() % 100;
        }
    }
}

int main() {
    test1();
    test2();
    test3();
    test4();
    test5();

    return 0;
}

```

Ln 45, Col 15

100%

Windows (CRLF)

UTF-8



jithu@DESKTOP-BGRVDL1: ~

```
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb) p ptr[i]
$2 = 0
(gdb) next
55 for (int i = 0; i < 10; i++) {
(gdb)
56 ptr[i] = i * 3;
(gdb)
55 for (int i = 0; i < 10; i++) {
(gdb)
58 printf("Value of *ptr: %d\n", *ptr);
(gdb)
Value of *ptr: 0
59 free(ptr);
(gdb)
60 }
(gdb)
main () at grind.c:89
89 test5();
(gdb)
```

Breakpoint 2, test5 () at grind.c:63

```
63 DataStruct *data = malloc(NUM_STRUCTS * sizeof(DataStruct));
(gdb)
```

grid - Notepad

```
File Edit Format View Help
for (int i = 0; i < 100; i++) {
    ptr[i] = i;
}
free(ptr);
printf("Value of *ptr: %d\n", ptr[10]);
}

void test2() {
    char *str = malloc(100 * sizeof(char));
    if (str == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    strcpy(str, "Good day to you!");
    printf("String: %s\n", str);
}

void test3() {
    int *ptr = malloc(sizeof(int) * 50);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 2; i++) {
        ptr[i] = i * 2;
    }
    free(ptr);
}

void test4() {
    int *ptr = malloc(sizeof(int) * 10);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 10; i++) {
        ptr[i] = i * 3;
    }
    free(ptr);
    ptr = NULL;
    printf("Value of *ptr: %d\n", *ptr);
}
```

Ln 45, Col 15

100%

Windows (CRLF)

UTF-8

Type here to search



33°C Partly sunny



18:41  
14-06-2024

jithu@DESKTOP-BGRVDL1: ~

```
(gdb)
main () at grid.c:77
77     test2();
(gdb)
```

```
Breakpoint 2, test2 () at grid.c:27
27     char *str = malloc(100 * sizeof(char));
(gdb)
28     if (str == NULL) {
(gdb)
32     strcpy(str, "Good day to you!");
(gdb)
33     printf("String: %s\n", str);
(gdb) p str
$29 = 0x555555559850 "Good day to you!"
(gdb)
$30 = 0x555555559850 "Good day to you!"
(gdb) next
String: Good day to you!
34 }
```

```
main () at grid.c:78
78     test3();
(gdb)
```

```
Breakpoint 3, test3 () at grid.c:37
37     int *ptr = malloc(sizeof(int) * 50);
(gdb)
38     if (ptr == NULL) {
(gdb)
42     for (int i = 0; i < 2; i++) {
(gdb)
43         ptr[i] = i * 2;
(gdb) p ptr[i]
$31 = 0
(gdb) next
44     free(ptr);
(gdb)
42     for (int i = 0; i < 2; i++) {
(gdb)
43         ptr[i] = i * 2;
(gdb)
44         free(ptr);
(gdb)
```

free(): double free detected in tcache 2

Program received signal SIGABRT, Aborted.

\_\_pthread\_kill\_implementation (no\_tid=0, signo=6, threadid=140737351538496) at ./nptl/pthread\_kill.c:44  
44 ./nptl/pthread\_kill.c: No such file or directory.

```
(gdb)
```

grid - Notepad

```
File Edit Format View Help
for (int i = 0; i < 100; i++) {
    ptr[i] = i;
}
free(ptr);
printf("Value of *ptr: %d\n", ptr[10]);
}

void test2() {
    char *str = malloc(100 * sizeof(char));
    if (str == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    strcpy(str, "Good day to you!");
    printf("String: %s\n", str);
}

void test3() {
    int *ptr = malloc(sizeof(int) * 50);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 2; i++) {
        ptr[i] = i * 2;
        free(ptr);
    }
}

void test4() {
    int *ptr = malloc(sizeof(int) * 10);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 10; i++) {
        ptr[i] = i * 3;
    }
    free(ptr);
    ptr = NULL;
    printf("Value of *ptr: %d\n", *ptr);
}
```

Ln 1, Col 1

100%

Windows (CRLF)

UTF-8

Type here to search



33°C Partly sunny



18:33  
14-06-2024



jithu@DESKTOP-BGRVDL1: -

```
(gdb) ptr[i] = i;
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   for (int i = 0; i < 100; i++) {
(gdb)
20       ptr[i] = i;
(gdb)
19   free(ptr);
22
(gdb) printf("Value of *ptr: %d\n", ptr[10]);
23 p ptr[10]
$27 = 10
(gdb)
$28 = 10
(gdb) next
Value of *ptr: 10
24 }
(gdb)
main () at grid.c:77
77 test2();
(gdb)
```

Breakpoint 2, test2 () at grid.c:27

```
27 char *str = malloc(100 * sizeof(char));
(gdb)
```

grid - Notepad

```
File Edit Format View Help
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define NUM_STRUCTS 50

typedef struct {
    char *name;
    int id;
    int *values;
} Datastruct;

void test1() {
    int *ptr = malloc(sizeof(int) * 100);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 100; i++) {
        ptr[i] = i;
    }
    free(ptr);
    printf("Value of *ptr: %d\n", ptr[10]);
}

void test2() {
    char *str = malloc(100 * sizeof(char));
    if (str == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    strcpy(str, "Good day to you!");
    printf("String: %s\n", str);
}


void test3() {
    int *ptr = malloc(sizeof(int) * 50);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 2; i++) {
```

Ln 1, Col 1

100%

Windows (CRLF)

UTF-8

 33°C Partly sunny

18:31  
14-06-2024

jithu@DESKTOP-BGRVDL1: -

```
(gdb) ptr[i] = i;
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   for (int i = 0; i < 100; i++) {
(gdb) 
20       ptr[i] = i;
(gdb) 
19   free(ptr);
22   printf("Value of *ptr: %d\n", ptr[10]);
(gdb) p ptr[10]
$27 = 10
(gdb)
```

grid - Notepad

```
File Edit Format View Help
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define NUM_STRUCTS 50

typedef struct {
    char *name;
    int id;
    int *values;
} Datastruct;

void test1() {
    int *ptr = malloc(sizeof(int) * 100);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 100; i++) {
        ptr[i] = i;
    }
    free(ptr);
    printf("Value of *ptr: %d\n", ptr[10]);
}

void test2() {
    char *str = malloc(100 * sizeof(char));
    if (str == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    strcpy(str, "Good day to you!");
    printf("String: %s\n", str);
}

void test3() {
    int *ptr = malloc(sizeof(int) * 50);
    if (ptr == NULL) {
        perror("Failed to allocate memory");
        return;
    }
    for (int i = 0; i < 50; i++) {
```


Ln 1, Col 1

100%

Windows (CRLF)

UTF-8

Windows Search bar: Type here to search

 33°C Partly sunny





 ENG

18:31  
14-06-2024



jithu@DESKTOP-BGRVDL1: ~

```
0x00005555555525d <+116>: mov    $0x0,%eax
0x000055555555262 <+121>: call   0x5555555550b0 <printf@plt>
0x000055555555267 <+126>: mov    -0x0(%rbp),%rax
0x00005555555526b <+130>: mov    %rax,%rdi
0x00005555555526e <+133>: call   0x5555555550a0 <free@plt>
0x000055555555273 <+138>: leave
0x000055555555274 <+139>: ret
```

End of assembler dump.

(gdb) info registers

rax	0x555555555fb	93824992237051
rbx	0x0	0
rcx	0x555555557d98	93824992247192
rdx	0x7fffffffefc8	140737488347336
rsi	0x7fffffffefb8	140737488347320
rdi	0x1	1
rbp	0x7fffffffdfa0	0x7fffffffdfa0
rsp	0x7fffffffdfa0	0x7fffffffdfa0
r8	0x7ffff7fa3f10	140737353760528
r9	0x7ffff7fc9040	140737353912384
r10	0x7ffff7fc3908	140737353890056
r11	0x7ffff7fde660	140737353999968
r12	0x7fffffffefb8	140737488347320
r13	0x555555555fb	93824992237051
r14	0x555555557d98	93824992247192
r15	0x7ffff7ffd040	140737354125376
rip	0x555555555603	0x555555555603 <main+8>
eflags	0x246	[ PF ZF IF ]
cs	0x33	51
ss	0x2b	43
ds	0x0	0
es	0x0	0
fs	0x0	0
gs	0x0	0

(gdb) disable 1

(gdb) enable

(gdb) enable 1

(gdb) info breakpoints

Num	Type	Disp	Enb	Address	What
1	breakpoint	keep y		0x0000555555555603	in main at grind.c:85

breakpoint already hit 1 time

(gdb)



Type here to search



33°C Mostly cloudy



18:57  
14-06-2024