

Esercitazione II - GDB

```
sh-3.1$ make
```

```
gcc -Wall -g3 -c -o test2.o test2.c
```

```
gcc -o test2 test2.o
```

```
sh-3.1$ gdb test2
```

```
GNU gdb 6.4.90-debian
```

```
Copyright (C) 2006 Free Software Foundation, Inc.
```

```
GDB is free software, covered by the GNU General Public License, and you are  
welcome to change it and/or distribute copies of it under certain conditions.  
Type "show copying" to see the conditions.
```

```
There is absolutely no warranty for GDB. Type "show warranty" for details.
```

```
This GDB was configured as "i486-linux-gnu"...Using host libthread_db library  
"/lib/tls/i686/cmov/libthread_db.so.1".
```

```
(gdb) run
```

```
Starting program: test2
```

```
10
```

```
9
```

```
8
```

```
7
```

```
6
```

```
5
```

```
4
```

```
3
```

```
2
```

```
1
```

```
*** glibc detected *** free(): invalid pointer: 0x0804a030 ***
```

```
Program received signal SIGABRT, Aborted.
```

```
0xfffffe410 in ?? ()
```

```
(gdb) backtrace
```

```
#0 0xfffffe410 in ?? ()
```

```
#1 0xbfef9e8c in ?? ()
```

```
#2 0x00000006 in ?? ()
```

```
#3 0x00000c9d in ?? ()
```

```
#4 0xb7e1b811 in raise () from /lib/tls/i686/cmov/libc.so.6
```

```
#5 0xb7e1cfb9 in abort () from /lib/tls/i686/cmov/libc.so.6
```

```
#6 0xb7e50e0a in __fsetlocking () from /lib/tls/i686/cmov/libc.so.6
```

```
#7 0xb7e5869f in malloc () from /lib/tls/i686/cmov/libc.so.6
```

```
#8 0xb7e58742 in free () from /lib/tls/i686/cmov/libc.so.6
```

```
#9 0x0804843e in f1 (n=-1) at test2.c:22
```

```
#10 0x080484d2 in main () at test2.c:41
```

```
(gdb) break f1
```

```
Breakpoint 1 at 0x80483e6: file test2.c, line 15.
```

```
(gdb) break 22
```

```
Breakpoint 2 at 0x8048433: file test2.c, line 22.
```

```
(gdb) run
```

```
The program being debugged has been started already.
```

```
Start it from the beginning? (y or n) y
```

```
Starting program: test2
```

```
Breakpoint 1, f1 (n=10) at test2.c:15
```

```
15 if (!(a = malloc(sizeof(int) * n))) {
```

```
(gdb) list
```

```
10 }
```

```
11 }
```

```
12
```

```
13 void f1(int n) {
```

```
14 int *a;
```

```
15 if (!(a = malloc(sizeof(int) * n))) {
```

```
16 return;
```

```

17     }
18     initialize_array(a, n);
19     while (n--) {
(gdb) next
18     initialize_array(a, n);
(gdb) info args
n = 10
(gdb) x/10d a
0x804a008:  0      0      0      0
0x804a018:  0      0      0      0
0x804a028:  0      0
(gdb) print a
$1 = (int *) 0x804a008
(gdb) list initialize_array
2     #include <stdlib.h>
3     #include <string.h>
4
5     static void initialize_array(int *a, int n);
6
7     static void initialize_array(int *a, int n) {
8         while (n) {
9             *(a++) = n--;
10        }
11    }
(gdb) next
19    while (n--) {
(gdb) x/10d a
0x804a008:  10      9      8      7
0x804a018:  6       5      4      3
0x804a028:  2       1
(gdb) next
20        printf("%d\n", *(a++));
(gdb) print a
$2 = (int *) 0x804a008
(gdb) print *a
$3 = 10
(gdb) next
10
19    while (n--) {
(gdb) print n
$4 = 9
(gdb) next
20        printf("%d\n", *(a++));
(gdb) print n
$5 = 8
(gdb) print a
$6 = (int *) 0x804a00c
(gdb) print *a
$7 = 9
(gdb) cont
Continuing.
9
8
7
6
5
4
3
2
1

```

Breakpoint 2, f1 (n=-1) at test2.c:22

```

22     free(a);
(gdb) print a
$8 = (int *) 0x804a030
(gdb) print *a
$9 = 0
(gdb) print $1
$10 = (int *) 0x804a008
(gdb) print *$1
$11 = 10
(gdb) x/10d $1
0x804a008: 10    9    8    7
0x804a018: 6    5    4    3
0x804a028: 2    1
(gdb) print a - $1
$12 = 10
(gdb) set variable a = $1
(gdb) print a
$13 = (int *) 0x804a008
(gdb) list 22
17     }
18     initialize_array(a, n);
19     while (n--) {
20         printf("%d\n", *(a++));
21     }
22     free(a);
23 }
24
25 void f2(void) {
26     char *a, *b, *c;
(gdb) next
23 }
(gdb) next
main () at test2.c:42
42     f2();
(gdb) where
#0  main () at test2.c:42

```

Primo errore: il puntatore "a" era stato incrementato di 10 posizioni

```

(gdb) cont
Continuing.
a = 'proval'
b = 'ciao'

```

Program received signal SIGSEGV, Segmentation fault.
0xb7e15766 in free () from /lib/tls/i686/cmov/libc.so.6

```

(gdb) backtrace
#0  0xb7e15766 in free () from /lib/tls/i686/cmov/libc.so.6
#1  0x080484b3 in f2 () at test2.c:37
#2  0x080484d7 in main () at test2.c:42
(gdb) info breakpoints
Num Type           Disp Enb Address      What
1  breakpoint      keep y   0x080483e6 in f1 at test2.c:15
    breakpoint already hit 1 time
2  breakpoint      keep y   0x08048433 in f1 at test2.c:22
    breakpoint already hit 1 time
(gdb) delete 1
(gdb) break f2
Breakpoint 3 at 0x8048446: file test2.c, line 27.
(gdb) break 37
Breakpoint 4 at 0x80484a8: file test2.c, line 37.
(gdb) run

```

The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: test2

10
9
8
7
6
5
4
3
2
1

Breakpoint 2, f1 (n=-1) at test2.c:22

22 free(a);

(gdb) print a

\$14 = (int *) 0x804a030

(gdb) set variable a = a-10

(gdb) x/10d a

0x804a008: 10 9 8 7
0x804a018: 6 5 4 3
0x804a028: 2 1

(gdb) cont

Continuing.

Breakpoint 3, f2 () at test2.c:27

27 a = "ciao";

(gdb) list

22 free(a);

23 }

24

25 void f2(void) {

26 char *a, *b, *c;

27 a = "ciao";

28 if (!(b = malloc(sizeof(char) * 10))) {

29 return;

30 }

31 strcpy(b, "prova1");

(gdb) next

28 if (!(b = malloc(sizeof(char) * 10))) {

(gdb) next

31 strcpy(b, "prova1");

(gdb) print a

\$15 = 0x80485fc "ciao"

(gdb) print b

\$16 = 0x804a038 ""

(gdb) x/10c b

0x804a038: 0 '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0' '\0'
0x804a040: 0 '\0' '\0'

(gdb) next

32 c = b;

(gdb) x/10c b

0x804a038: 112 'p' 114 'r' 111 'o' 118 'v' 97 'a' 49 '1' '\0' '\0'
'\0'

0x804a040: 0 '\0' '\0'

(gdb) next

33 b = a;

(gdb) next

34 a = c;

(gdb) next

35 c = NULL;

```
(gdb) next
36     printf("a = '%s'\nb = '%s'\n", a, b);
(gdb) info locals
a = 0x804a038 "prova1"
b = 0x80485fc "ciao"
c = 0x0
(gdb) next
a = 'prova1'
b = 'ciao'
```

Breakpoint 4, f2 () at test2.c:37

```
37     free(b);
(gdb) print b
$17 = 0x80485fc "ciao"
(gdb) print $15
$18 = 0x804a038 "prova1"
(gdb) set variable b = $16
(gdb) print b
$19 = 0x804a038 "prova1"
(gdb) next
38     }
(gdb) cont
Continuing.
```

Program exited normally.

```
(gdb) quit
```

Secondo errore: chiamata free() su un indirizzo statico