```
 1: // $Id: segfault.c,v 1.24 2014-04-15 18:40:30-07 - - $
 2:
 3: // Illustrate a segfault.
 4:
 5: #include <stdio.h>
 6:
 7: int main (int argc, char **argv) {
 8:    (void) argc; // warning: unused parameter 'argc' [-Wunused-parameter]
 9:    for (int i = 0;; ++i) {
10:       printf ("argv[%d]=\"%s\"\n", i, argv[i]);
11:       fflush (NULL);
12:    }
13: }
14:
15: /*
16: //TEST// env -i FOO=value1 BAR=value2 \
17: //TEST//    valgrind --leak-check=full --show-reachable=yes \
18: //TEST//       ./segfault arg1 arg2 >segfault.out 2>segfault.err
19: //TEST// for file in segfault.out segfault.err; do
20: //TEST//    more $file </dev/null; echo ""
21: //TEST// done >segfault.lis
22: //TEST// rm segfault.out segfault.err
23: //TEST// mkpspdf segfault.ps segfault.c* segfault.lis
24: */
25:
```

```
1: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: starting segfault.c
2: segfault.c:
3:      $Id: segfault.c,v 1.24 2014-04-15 18:40:30-07 - - $
4: gcc -g -O0 -Wall -Wextra -rdynamic -std=gnu11 segfault.c
5:          -o segfault -lglut -lGLU -lGL -lX11 -lrt -lm
6: rm -f segfault.o
7: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ mkc: finished segfault.c
```

```
    1: ::::::::::::::
    2: segfault.out
    3: ::::::::::::::
    4: argv[0]="./segfault"
    5: argv[1]="arg1"
    6: argv[2]="arg2"
    7: argv[3]="(null)"
    8: argv[4]="FOO=value1"
    9: argv[5]="BAR=value2"
   10: argv[6]="LD_PRELOAD=/usr/lib64/valgrind/vgpreload_core-amd64-linux.so:/u
sr/lib64/valgrind/vgpreload_memcheck-amd64-linux.so"
   11: argv[7]="(null)"
   12: argv[8]="
   13: ::::::::::::::
   14: segfault.err
   15: ::::::::::::::
   16: ==20152== Memcheck, a memory error detector
   17: ==20152== Copyright (C) 2002-2013, and GNU GPL'd, by Julian Seward et al
.
   18: ==20152== Using Valgrind-3.10.0 and LibVEX; rerun with -h for copyright
info
   19: ==20152== Command: ./segfault arg1 arg2
   20: ==20152==
   21: ==20152== Invalid read of size 1
   22: ==20152==    at 0x5E27AB4: vfprintf (in /usr/lib64/libc-2.17.so)
   23: ==20152==    by 0x5E30C98: printf (in /usr/lib64/libc-2.17.so)
   24: ==20152==    by 0x400876: main (segfault.c:10)
   25: ==20152==  Address 0x1 is not stack'd, malloc'd or (recently) free'd
   26: ==20152==
   27: ==20152==
   28: ==20152== Process terminating with default action of signal 11 (SIGSEGV)
   29: ==20152==  Access not within mapped region at address 0x1
   30: ==20152==    at 0x5E27AB4: vfprintf (in /usr/lib64/libc-2.17.so)
   31: ==20152==    by 0x5E30C98: printf (in /usr/lib64/libc-2.17.so)
   32: ==20152==    by 0x400876: main (segfault.c:10)
   33: ==20152==  If you believe this happened as a result of a stack
   34: ==20152==  overflow in your program's main thread (unlikely but
   35: ==20152==  possible), you can try to increase the size of the
   36: ==20152==  main thread stack using the --main-stacksize= flag.
   37: ==20152==  The main thread stack size used in this run was 8388608.
   38: ==20152==
   39: ==20152== HEAP SUMMARY:
   40: ==20152==     in use at exit: 16 bytes in 1 blocks
   41: ==20152==   total heap usage: 1 allocs, 0 frees, 16 bytes allocated
   42: ==20152==
   43: ==20152== 16 bytes in 1 blocks are still reachable in loss record 1 of 1
   44: ==20152==    at 0x4C29BFD: malloc (in /usr/lib64/valgrind/vgpreload_memc
heck-amd64-linux.so)
   45: ==20152==    by 0x5E64FE9: strdup (in /usr/lib64/libc-2.17.so)
   46: ==20152==    by 0x7144628: ??? (in /usr/lib64/libselinux.so.1)
   47: ==20152==    by 0x71446B2: ??? (in /usr/lib64/libselinux.so.1)
   48: ==20152==    by 0x400F3A2: _dl_init (in /usr/lib64/ld-2.17.so)
   49: ==20152==    by 0x4001469: ??? (in /usr/lib64/ld-2.17.so)
   50: ==20152==    by 0x2: ???
   51: ==20152==    by 0xFFF000F3E: ???
   52: ==20152==    by 0xFFF000F49: ???
   53: ==20152==    by 0xFFF000F4E: ???
   54: ==20152==
```

```
55: ==20152== LEAK SUMMARY:
56: ==20152==    definitely lost: 0 bytes in 0 blocks
57: ==20152==    indirectly lost: 0 bytes in 0 blocks
58: ==20152==      possibly lost: 0 bytes in 0 blocks
59: ==20152==    still reachable: 16 bytes in 1 blocks
60: ==20152==         suppressed: 0 bytes in 0 blocks
61: ==20152==
62: ==20152== For counts of detected and suppressed errors, rerun with: -v
63: ==20152== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 1 from 1)
64:
```