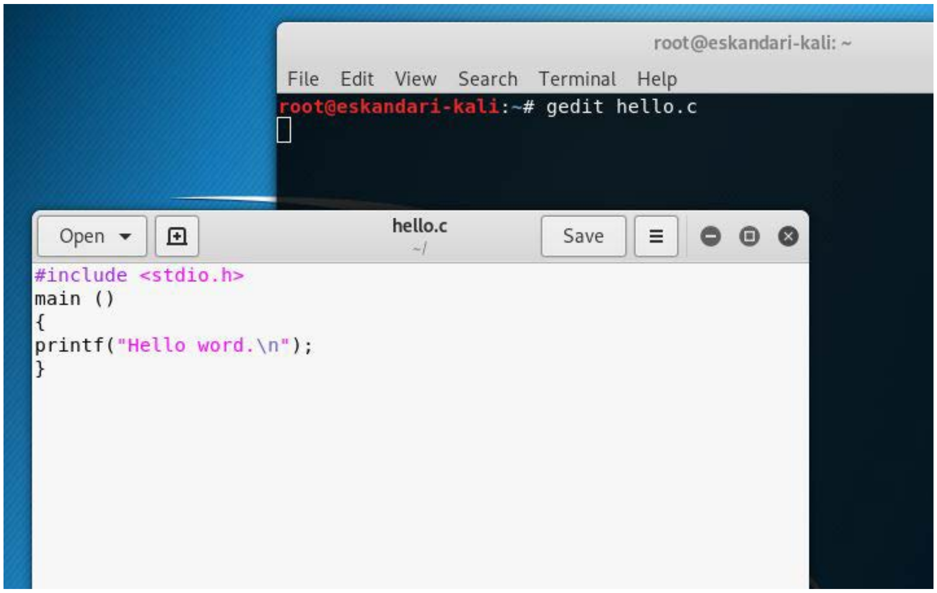
Question:

Write a hello-world C program and explain how we can dump its binary code with radare2.

Answer:

Write a hello world c program.

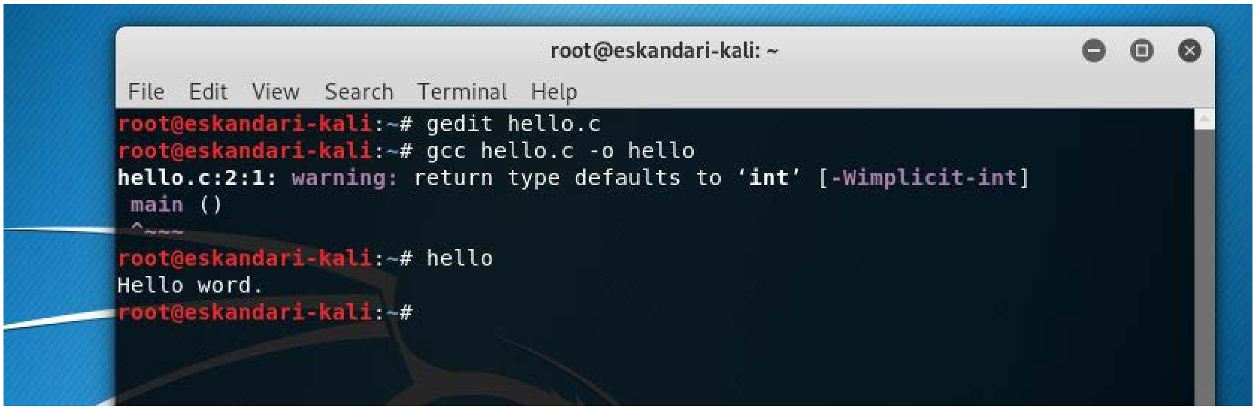


```
root@eskandari-kali:~# gedit hello.c
```

```
#include <stdio.h>
main ()
{
printf("Hello word.\n");
}
```

The return value of main function are missed!!, but it is not imortant, since gcc will add it.

Compile the program

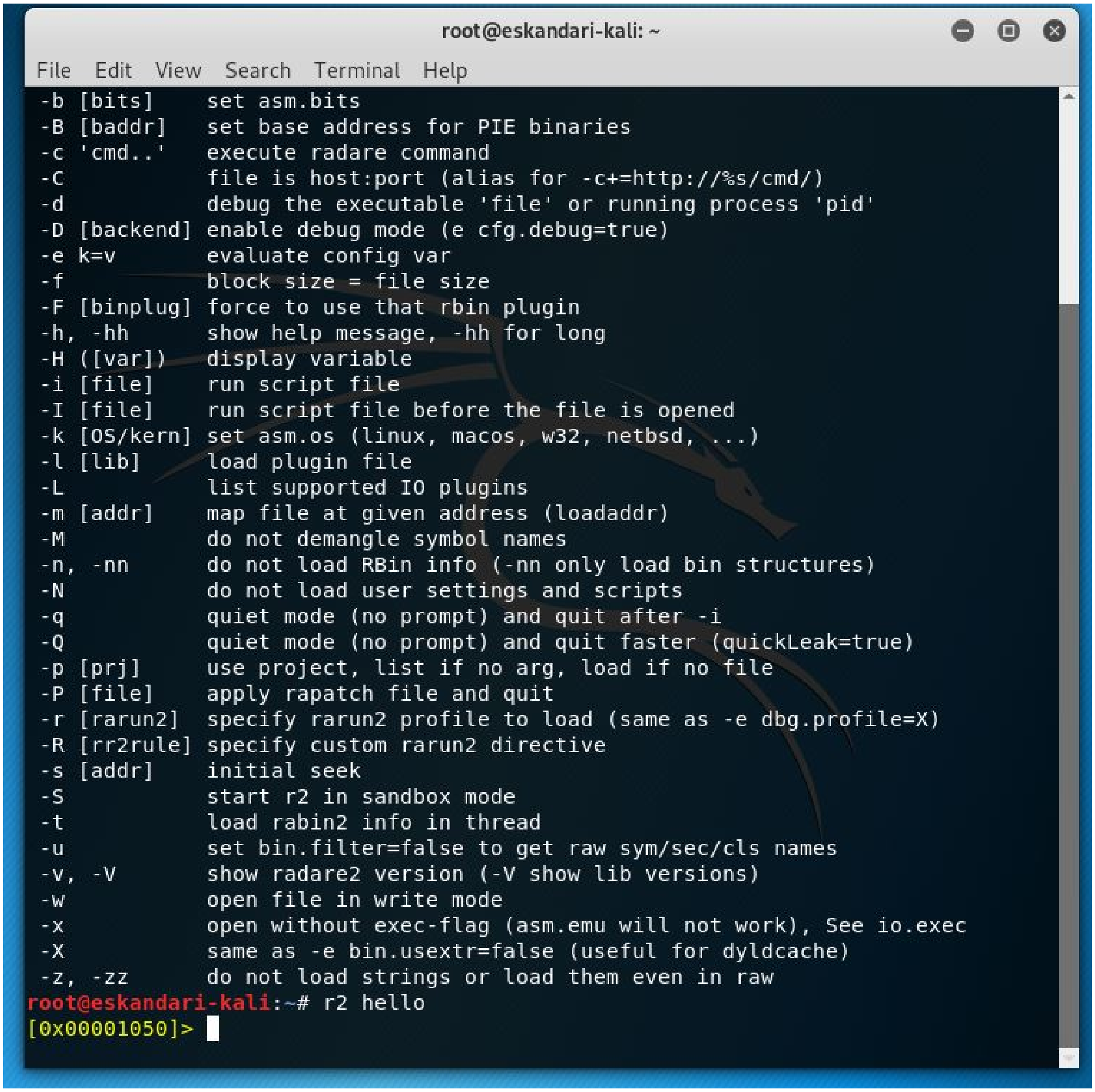


```
root@eskandari-kali:~# gedit hello.c
root@eskandari-kali:~# gcc hello.c -o hello
hello.c:2:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 main ()
 ^~~~
root@eskandari-kali:~# hello
Hello word.
root@eskandari-kali:~#
```

Execute radar 2 from the reverse engineering tools in Kali.

Radare2 is an open source set of tools for reverse-engineering and analysis of binary files (among other things, for example debugging).

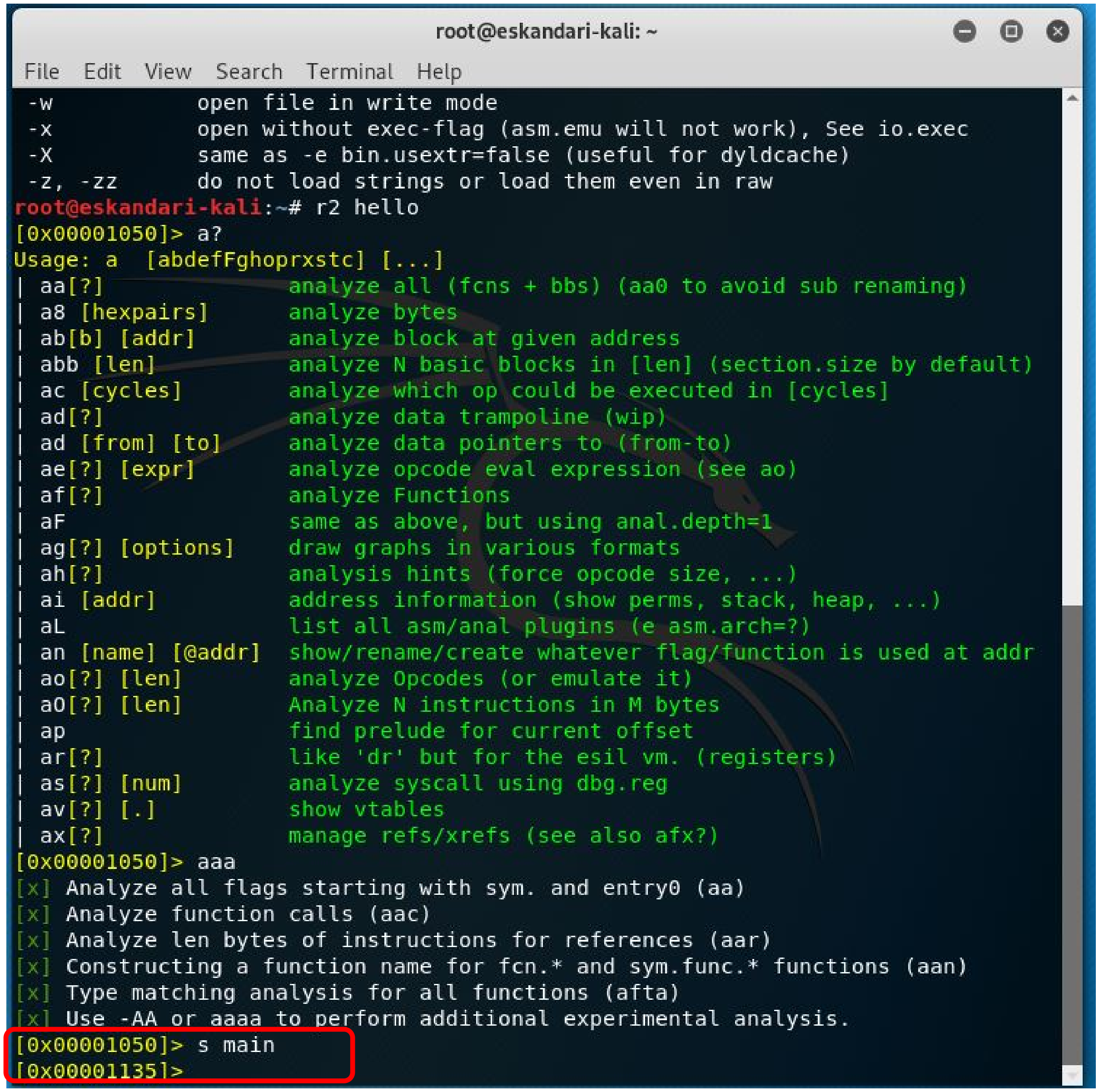
let's disassemble the hello program by running the following command:



```
 -b [bits]     set asm.bits
 -B [baddr]    set base address for PIE binaries
 -c 'cmd..'    execute radare command
 -C            file is host:port (alias for -c+=http://%s/cmd/)
 -d            debug the executable 'file' or running process 'pid'
 -D [backend] enable debug mode (e cfg.debug=true)
 -e k=v        evaluate config var
 -f            block size = file size
 -F [binplug] force to use that rbin plugin
 -h, -hh       show help message, -hh for long
 -H ([var])    display variable
 -i [file]     run script file
 -I [file]     run script file before the file is opened
 -k [OS/kern] set asm.os (linux, macos, w32, netbsd, ...)
 -l [lib]      load plugin file
 -L            list supported IO plugins
 -m [addr]     map file at given address (loadaddr)
 -M            do not demangle symbol names
 -n, -nn       do not load RBin info (-nn only load bin structures)
 -N            do not load user settings and scripts
 -q            quiet mode (no prompt) and quit after -i
 -Q            quiet mode (no prompt) and quit faster (quickLeak=true)
 -p [prj]      use project, list if no arg, load if no file
 -P [file]     apply rapatch file and quit
 -r [rarun2]   specify rarun2 profile to load (same as -e dbg.profile=X)
 -R [rr2rule] specify custom rarun2 directive
 -s [addr]     initial seek
 -S            start r2 in sandbox mode
 -t            load rabin2 info in thread
 -u            set bin.filter=false to get raw sym/sec/cls names
 -v, -V        show radare2 version (-V show lib versions)
 -w            open file in write mode
 -x            open without exec-flag (asm.emu will not work), See io.exec
 -X            same as -e bin.usextr=false (useful for dyldcache)
 -z, -zz       do not load strings or load them even in raw
root@eskandari-kali:~# r2 hello
[0x00001050]>
```

At this point, analyze the whole code: aa (Analyze All)

Analyze all with aaa command then seek to main function

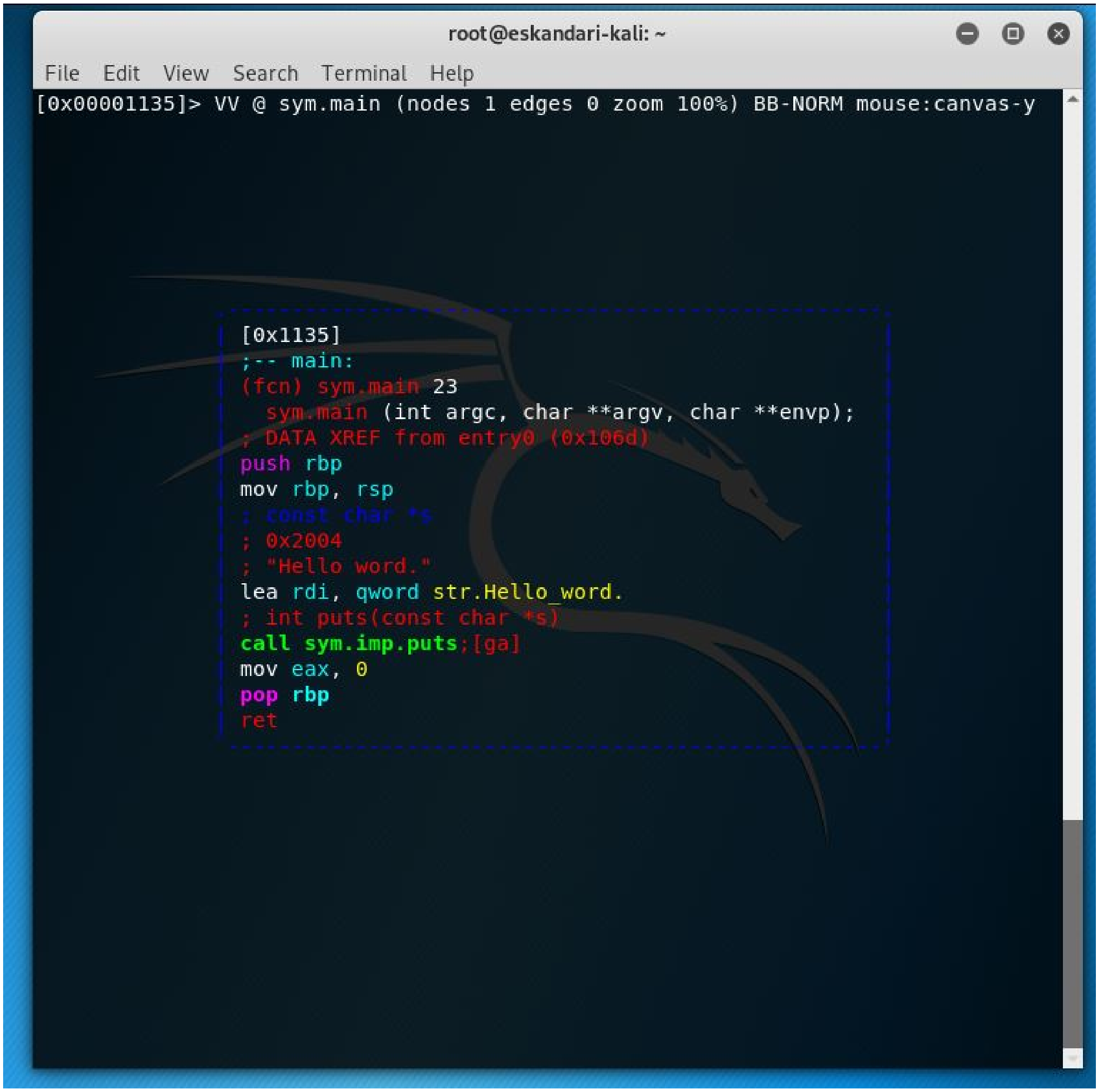


```
 -w            open file in write mode
 -x            open without exec-flag (asm.emu will not work), See io.exec
 -X            same as -e bin.usextr=false (useful for dyldcache)
 -z, -zz       do not load strings or load them even in raw
root@eskandari-kali:~# r2 hello
[0x00001050]> a?
Usage: a  [abdefFghoprxstc] [...]
| aa[?]                   analyze all (fcns + bbs) (aa0 to avoid sub renaming)
| a8 [hexpairs]           analyze bytes
| ab[b] [addr]            analyze block at given address
| abb [len]               analyze N basic blocks in [len] (section.size by default)
| ac [cycles]             analyze which op could be executed in [cycles]
| ad[?]                   analyze data trampoline (wip)
| ad [from] [to]          analyze data pointers to (from-to)
| ae[?] [expr]            analyze opcode eval expression (see ao)
| af[?]                   analyze Functions
| aF                      same as above, but using anal.depth=1
| ag[?] [options]         draw graphs in various formats
| ah[?]                   analysis hints (force opcode size, ...)
| ai [addr]               address information (show perms, stack, heap, ...)
| aL                      list all asm/anal plugins (e asm.arch=?)
| an [name] [@addr]       show/rename/create whatever flag/function is used at addr
| ao[?] [len]             analyze Opcodes (or emulate it)
| aO[?] [len]             Analyze N instructions in M bytes
| ap                      find prelude for current offset
| ar[?]                   like 'dr' but for the esil vm. (registers)
| as[?] [num]             analyze syscall using dbg.reg
| av[?] [.]               show vtables
| ax[?]                   manage refs/xrefs (see also afx?)
[0x00001050]> aaa
[x] Analyze all flags starting with sym. and entry0 (aa)
[x] Analyze function calls (aac)
[x] Analyze len bytes of instructions for references (aar)
[x] Constructing a function name for fcn.* and sym.func.* functions (aan)
[x] Type matching analysis for all functions (afta)
[x] Use -AA or aaaa to perform additional experimental analysis.
[0x00001050]> s main
[0x00001135]>
```

The memory address changed to 0x00001135 (start of main function)

Now, we show the execution of the program step by step. let's see the main function:

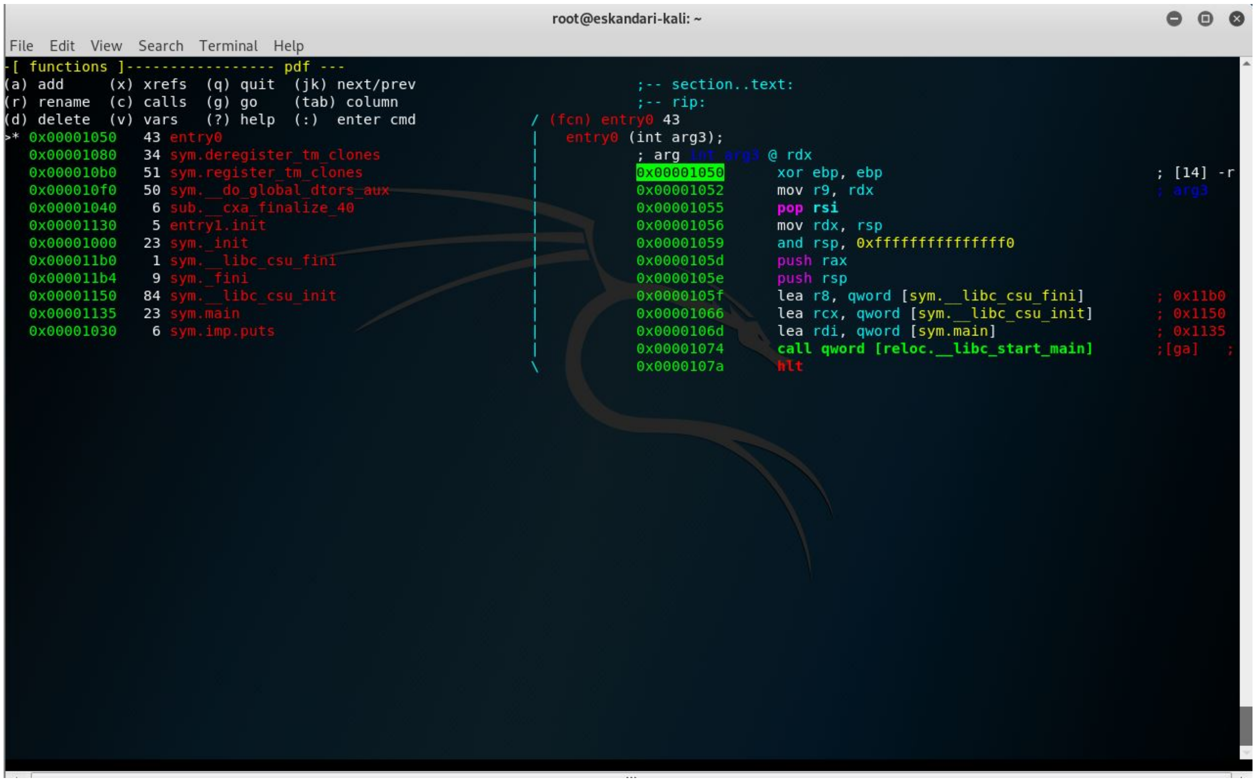
pdf @ sym.main (Print Disassemble Function)



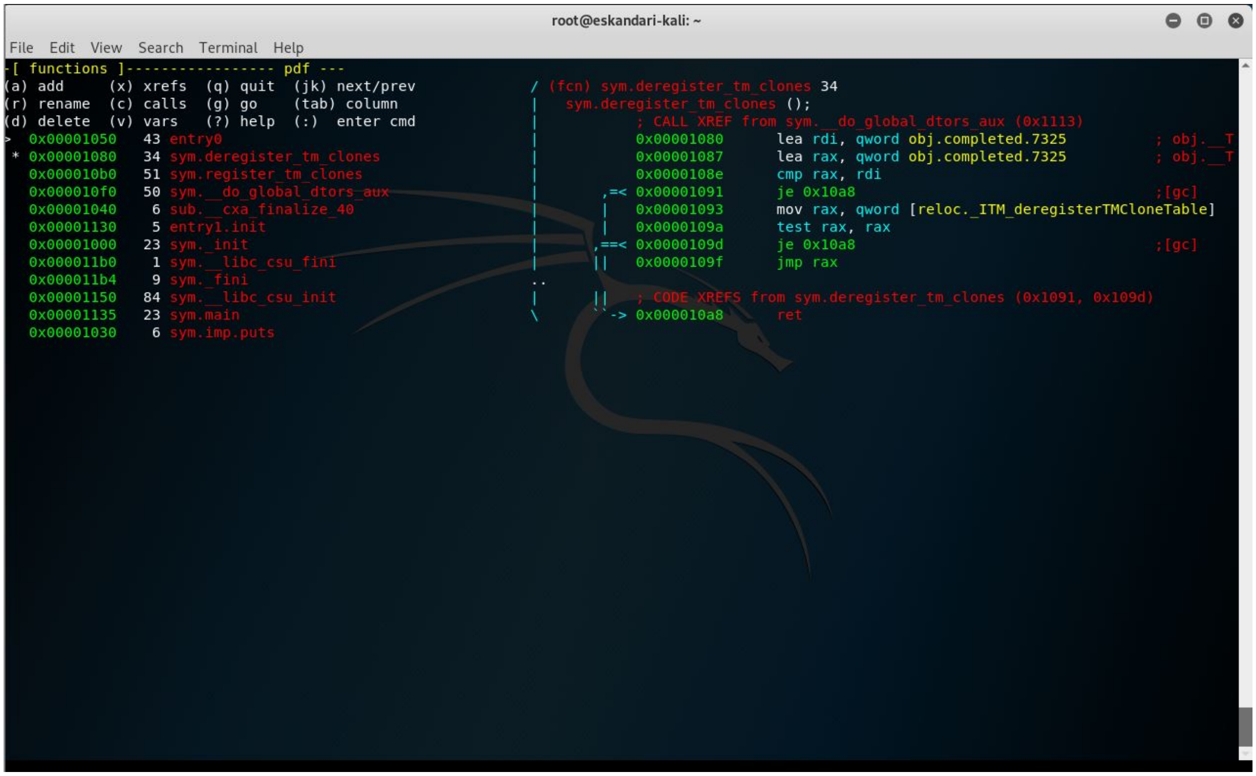
```
[0x00001135]> VV @ sym.main (nodes 1 edges 0 zoom 100%) BB-NORM mouse:canvas-y

              [0x1135]
              ;-- main:
              (fcn) sym.main 23
                sym.main (int argc, char **argv, char **envp);
              ; DATA XREF from entry0 (0x106d)
              push rbp
              mov rbp, rsp
              ; const char *s
              ; 0x2004
              ; "Hello word."
              lea rdi, qword str.Hello_word.
              ; int puts(const char *s)
              call sym.imp.puts;[ga]
              mov eax, 0
              pop rbp
              ret
```

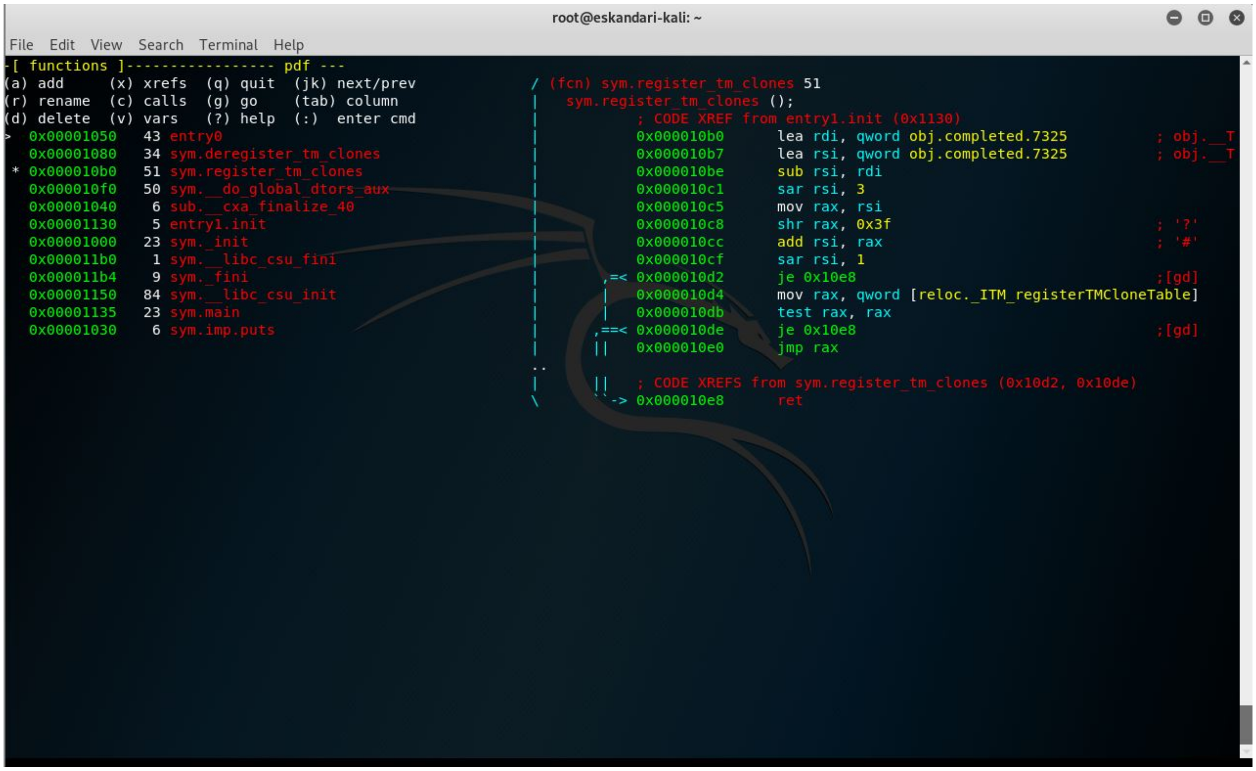
As you see, it shows the hexadecimal code and the assembly code.



```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev                ;-- section..text:
(r) rename  (c) calls  (g) go    (tab) column                  ;-- rip:
(d) delete  (v) vars   (?) help  (:)  enter cmd          / (fcn) entry0 43
>* 0x00001050   43 entry0                                |   entry0 (int arg3);
   0x00001080   34 sym.deregister_tm_clones              |           ; arg int arg3 @ rdx
   0x000010b0   51 sym.register_tm_clones                |           0x00001050      xor ebp, ebp                              ; [14] -r
   0x000010f0   50 sym.__do_global_dtors_aux             |           0x00001052      mov r9, rdx                               ; arg3
   0x00001040    6 sub.__cxa_finalize_40                 |           0x00001055      pop rsi
   0x00001130    5 entry1.init                           |           0x00001056      mov rdx, rsp
   0x00001000   23 sym._init                             |           0x00001059      and rsp, 0xfffffffffffffff0
   0x000011b0    1 sym.__libc_csu_fini                   |           0x0000105d      push rax
   0x000011b4    9 sym._fini                             |           0x0000105e      push rsp
   0x00001150   84 sym.__libc_csu_init                   |           0x0000105f      lea r8, qword [sym.__libc_csu_fini]       ; 0x11b0
   0x00001135   23 sym.main                              |           0x00001066      lea rcx, qword [sym.__libc_csu_init]      ; 0x1150
   0x00001030    6 sym.imp.puts                          |           0x0000106d      lea rdi, qword [sym.main]                 ; 0x1135
                                                         |           0x00001074      call qword [reloc.__libc_start_main]      ;[ga]   ;
                                                         \           0x0000107a      hlt
```

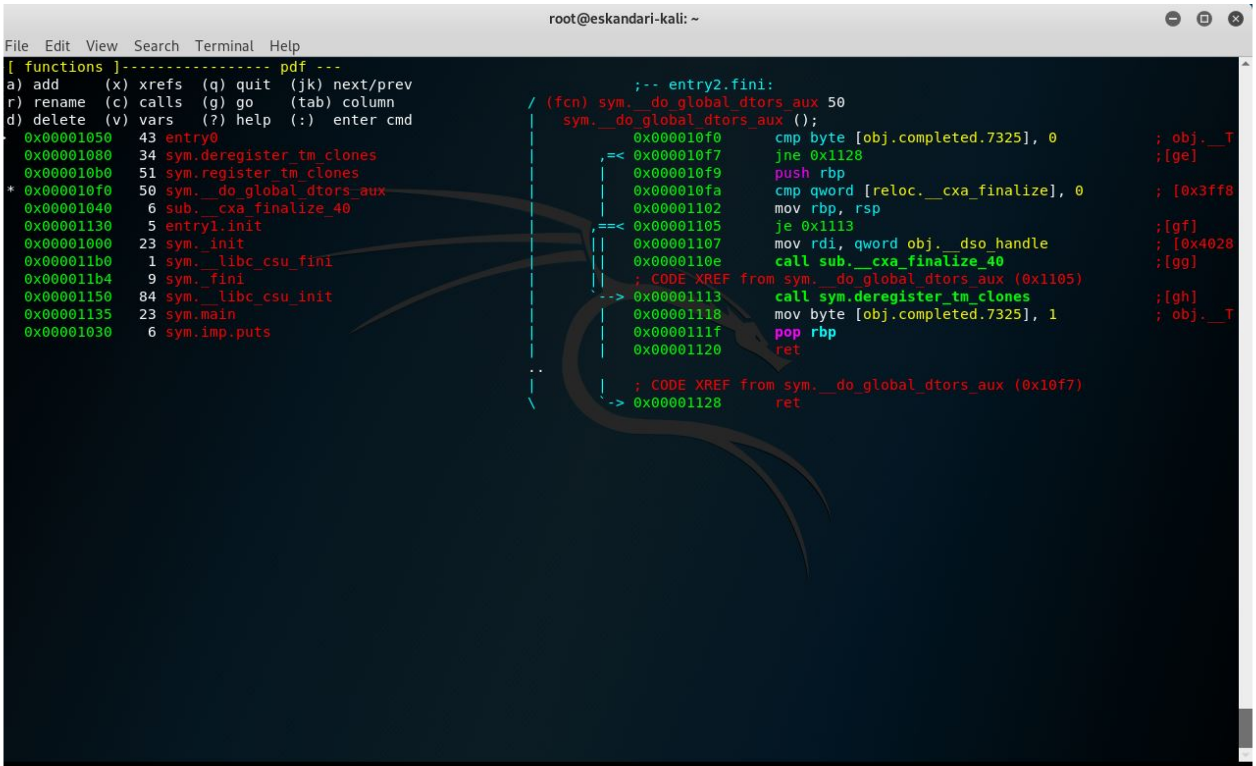


```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev          / (fcn) sym.deregister_tm_clones 34
(r) rename  (c) calls  (g) go    (tab) column            |   sym.deregister_tm_clones ();
(d) delete  (v) vars   (?) help  (:)  enter cmd          |           ; CALL XREF from sym.__do_global_dtors_aux (0x1113)
>  0x00001050   43 entry0                                |           0x00001080      lea rdi, qword obj.completed.7325         ; obj.__T
 * 0x00001080   34 sym.deregister_tm_clones              |           0x00001087      lea rax, qword obj.completed.7325         ; obj.__T
   0x000010b0   51 sym.register_tm_clones                |           0x0000108e      cmp rax, rdi
   0x000010f0   50 sym.__do_global_dtors_aux             |       ,=< 0x00001091      je 0x10a8                                 ;[gc]
   0x00001040    6 sub.__cxa_finalize_40                 |       |   0x00001093      mov rax, qword [reloc._ITM_deregisterTMCloneTable]
   0x00001130    5 entry1.init                           |       |   0x0000109a      test rax, rax
   0x00001000   23 sym._init                             |      ,==< 0x0000109d      je 0x10a8                                 ;[gc]
   0x000011b0    1 sym.__libc_csu_fini                   |      ||   0x0000109f      jmp rax
   0x000011b4    9 sym._fini                             ..
   0x00001150   84 sym.__libc_csu_init                   |      ||   ; CODE XREFS from sym.deregister_tm_clones (0x1091, 0x109d)
   0x00001135   23 sym.main                              \      ``-> 0x000010a8      ret
   0x00001030    6 sym.imp.puts
```

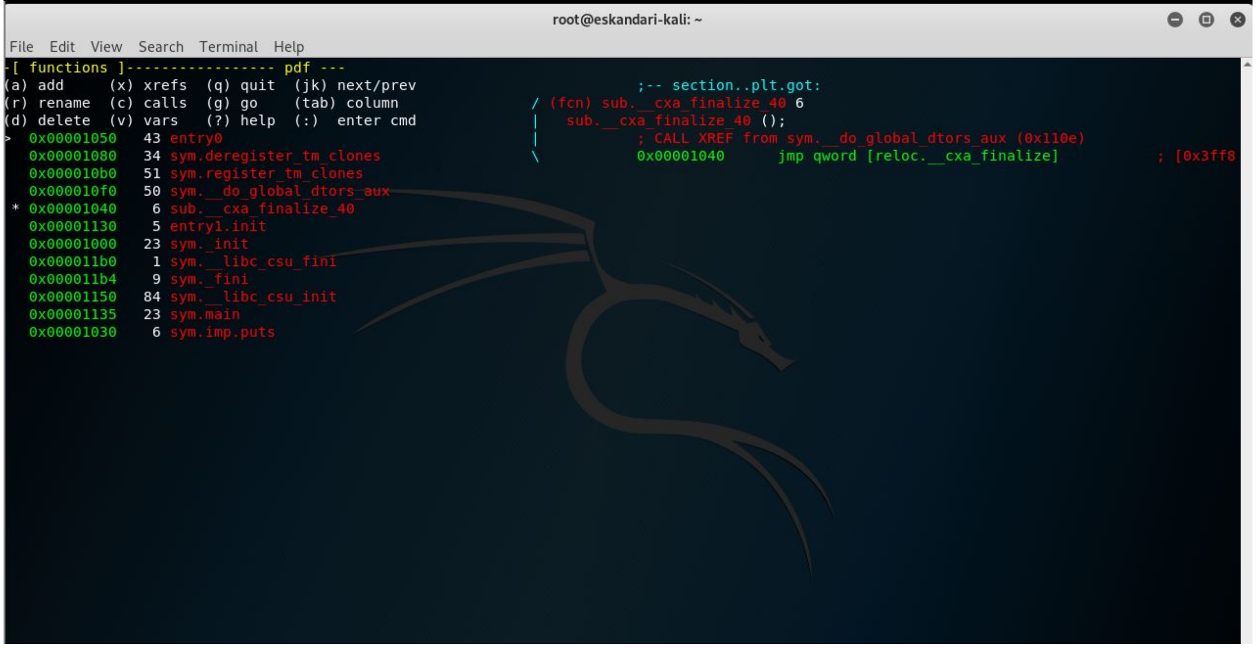


```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev
(r) rename  (c) calls  (g) go    (tab) column
(d) delete  (v) vars   (?) help  (:)  enter cmd
>  0x00001050   43 entry0
   0x00001080   34 sym.deregister_tm_clones
 * 0x000010b0   51 sym.register_tm_clones
   0x000010f0   50 sym.__do_global_dtors_aux
   0x00001040    6 sub.__cxa_finalize_40
   0x00001130    5 entry1.init
   0x00001000   23 sym._init
   0x000011b0    1 sym.__libc_csu_fini
   0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
   0x00001030    6 sym.imp.puts
```

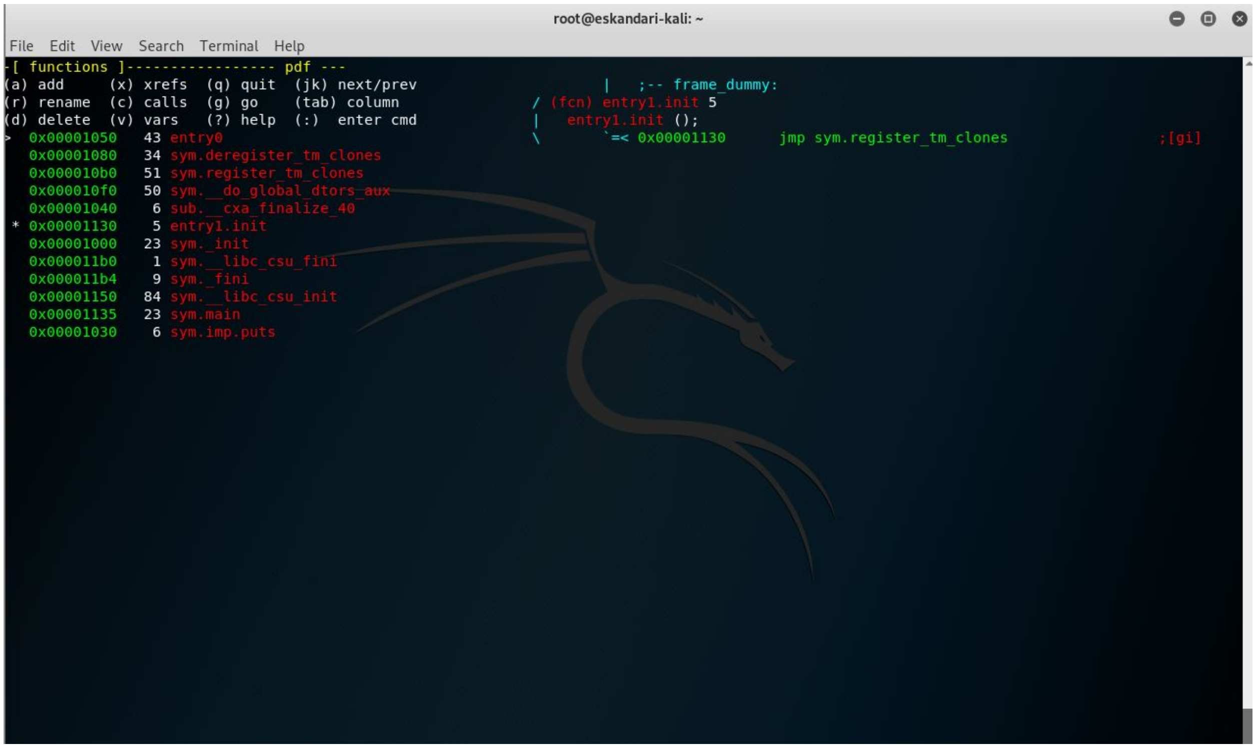
```
/ (fcn) sym.register_tm_clones 51
|   sym.register_tm_clones ();
|           ; CODE XREF from entry1.init (0x1130)
|           0x000010b0      lea rdi, qword obj.completed.7325          ; obj.__T
|           0x000010b7      lea rsi, qword obj.completed.7325          ; obj.__T
|           0x000010be      sub rsi, rdi
|           0x000010c1      sar rsi, 3
|           0x000010c5      mov rax, rsi
|           0x000010c8      shr rax, 0x3f                              ; '?'
|           0x000010cc      add rsi, rax                               ; '#'
|           0x000010cf      sar rsi, 1
|       ,=< 0x000010d2      je 0x10e8                                  ;[gd]
|       |   0x000010d4      mov rax, qword [reloc._ITM_registerTMCloneTable]
|       |   0x000010db      test rax, rax
|      ,==< 0x000010de      je 0x10e8                                  ;[gd]
|      ||   0x000010e0      jmp rax
..
|      ||   ; CODE XREFS from sym.register_tm_clones (0x10d2, 0x10de)
\      ``-> 0x000010e8      ret
```


root@eskandari-kali: ~

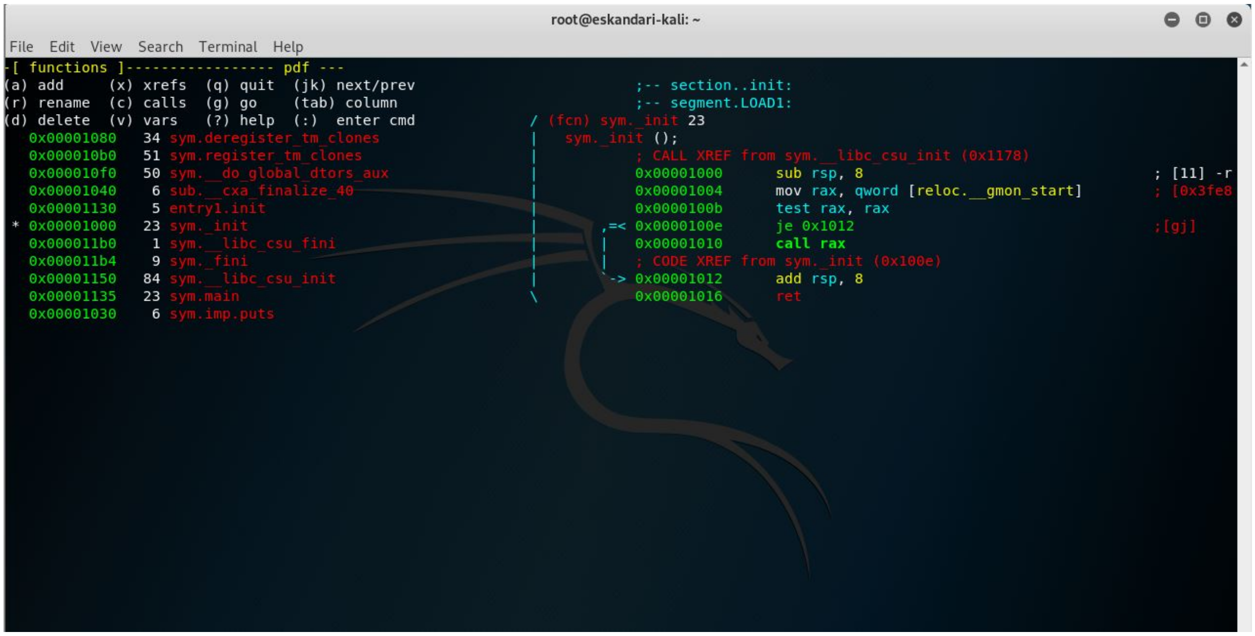
```
[ functions ]----------------- pdf ---
a) add     (x) xrefs  (q) quit  (jk) next/prev                         ;-- entry2.fini:
r) rename  (c) calls  (g) go    (tab) column               / (fcn) sym.__do_global_dtors_aux 50
d) delete  (v) vars   (?) help  (:)  enter cmd             |   sym.__do_global_dtors_aux ();
  0x00001050   43 entry0                                   |           0x000010f0      cmp byte [obj.completed.7325], 0           ; obj.__T
  0x00001080   34 sym.deregister_tm_clones                 |       ,=< 0x000010f7      jne 0x1128                                 ;[ge]
  0x000010b0   51 sym.register_tm_clones                   |       |   0x000010f9      push rbp
* 0x000010f0   50 sym.__do_global_dtors_aux                |       |   0x000010fa      cmp qword [reloc.__cxa_finalize], 0        ; [0x3ff8
  0x00001040    6 sub.__cxa_finalize_40                    |       |   0x00001102      mov rbp, rsp
  0x00001130    5 entry1.init                              |      ,==< 0x00001105      je 0x1113                                  ;[gf]
  0x00001000   23 sym._init                                |      ||   0x00001107      mov rdi, qword obj.__dso_handle            ; [0x4028
  0x000011b0    1 sym.__libc_csu_fini                      |      ||   0x0000110e      call sub.__cxa_finalize_40                 ;[gg]
  0x000011b4    9 sym._fini                                |      ||   ; CODE XREF from sym.__do_global_dtors_aux (0x1105)
  0x00001150   84 sym.__libc_csu_init                      |      `--> 0x00001113      call sym.deregister_tm_clones              ;[gh]
  0x00001135   23 sym.main                                 |       |   0x00001118      mov byte [obj.completed.7325], 1           ; obj.__T
  0x00001030    6 sym.imp.puts                             |       |   0x0000111f      pop rbp
                                                           |       |   0x00001120      ret
                                                           ..
                                                           |       |   ; CODE XREF from sym.__do_global_dtors_aux (0x10f7)
                                                           \       `-> 0x00001128      ret
```


root@eskandari-kali: ~

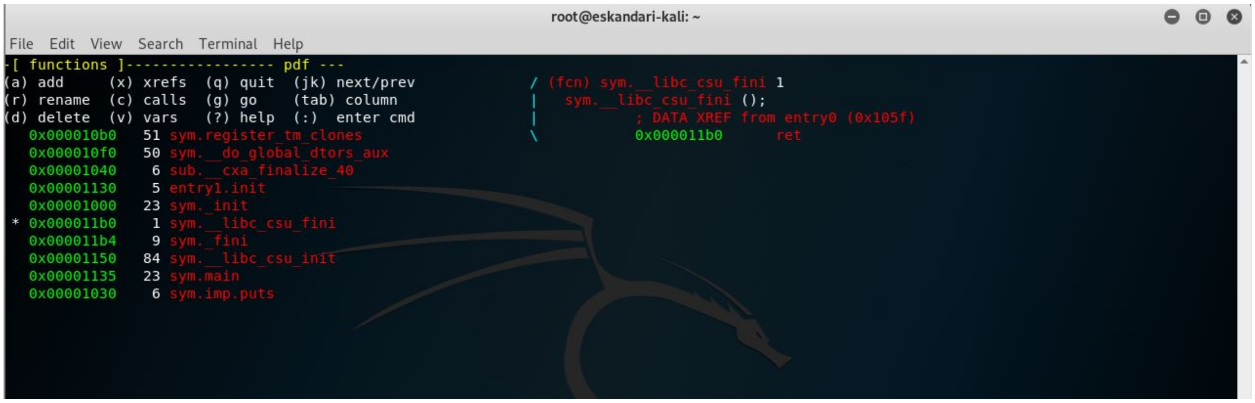
```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev                         ;-- section..plt.got:
(r) rename  (c) calls  (g) go    (tab) column               / (fcn) sub.__cxa_finalize_40 6
(d) delete  (v) vars   (?) help  (:)  enter cmd             |   sub.__cxa_finalize_40 ();
>  0x00001050   43 entry0                                   |           ; CALL XREF from sym.__do_global_dtors_aux (0x110e)
   0x00001080   34 sym.deregister_tm_clones                 \           0x00001040      jmp qword [reloc.__cxa_finalize]          ; [0x3ff8
   0x000010b0   51 sym.register_tm_clones
   0x000010f0   50 sym.__do_global_dtors_aux
 * 0x00001040    6 sub.__cxa_finalize_40
   0x00001130    5 entry1.init
   0x00001000   23 sym._init
   0x000011b0    1 sym.__libc_csu_fini
   0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
   0x00001030    6 sym.imp.puts
```



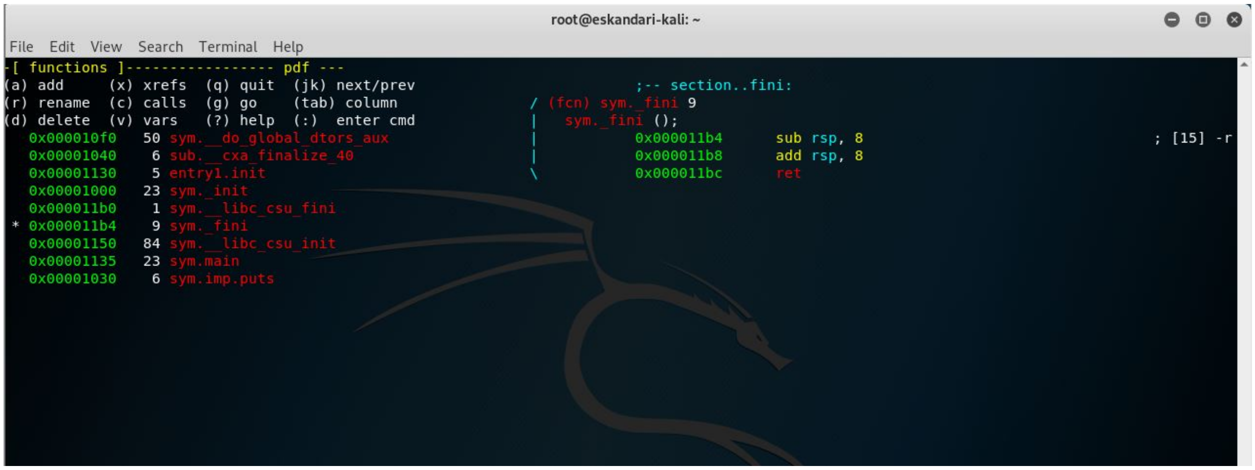
```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev              |   ;-- frame_dummy:
(r) rename  (c) calls  (g) go    (tab) column            / (fcn) entry1.init 5
(d) delete  (v) vars   (?) help  (:)  enter cmd          |   entry1.init ();
>  0x00001050   43 entry0                                \        `=< 0x00001130      jmp sym.register_tm_clones                 ;[gi]
   0x00001080   34 sym.deregister_tm_clones
   0x000010b0   51 sym.register_tm_clones
   0x000010f0   50 sym.__do_global_dtors_aux
   0x00001040    6 sub.__cxa_finalize_40
 * 0x00001130    5 entry1.init
   0x00001000   23 sym._init
   0x000011b0    1 sym.__libc_csu_fini
   0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
   0x00001030    6 sym.imp.puts
```



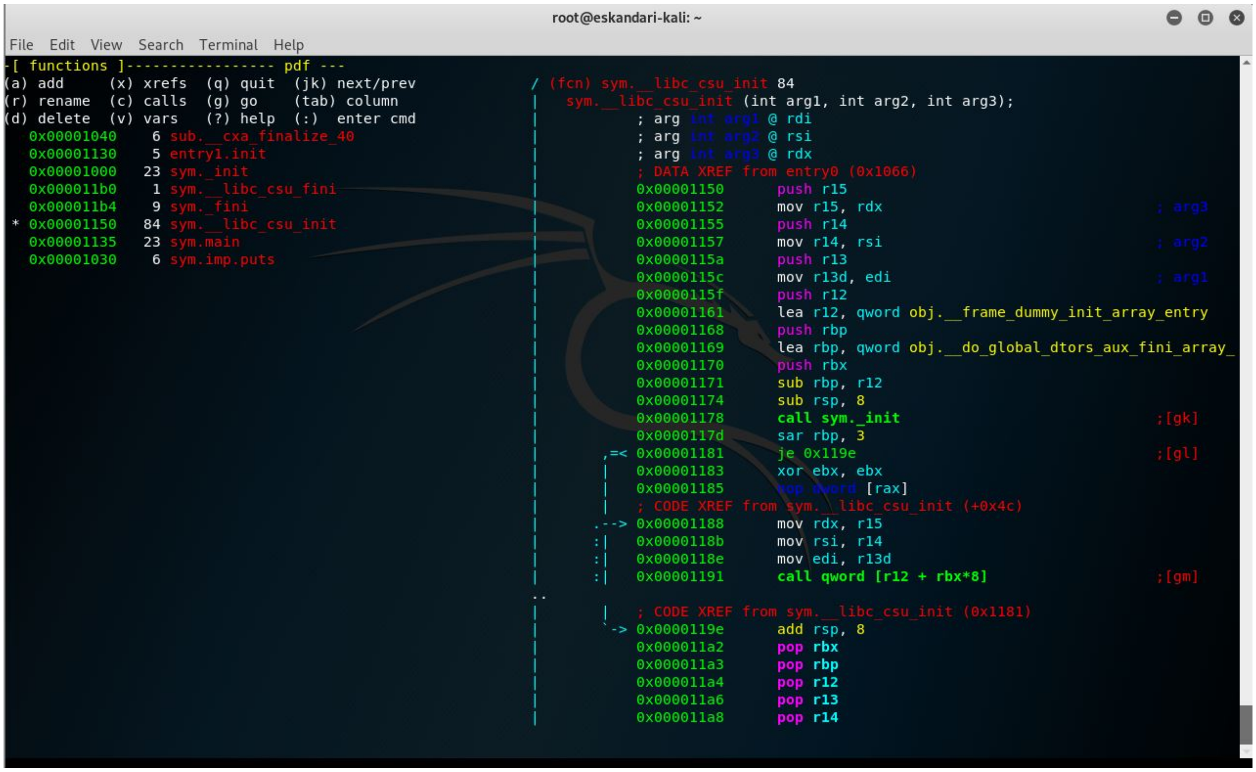
```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev                   ;-- section..init:
(r) rename  (c) calls  (g) go    (tab) column                     ;-- segment.LOAD1:
(d) delete  (v) vars   (?) help  (:)  enter cmd          / (fcn) sym._init 23
   0x00001080   34 sym.deregister_tm_clones              |   sym._init ();
   0x000010b0   51 sym.register_tm_clones                |           ; CALL XREF from sym.__libc_csu_init (0x1178)
   0x000010f0   50 sym.__do_global_dtors_aux             |           0x00001000      sub rsp, 8                            ; [11] -r
   0x00001040    6 sub.__cxa_finalize_40                 |           0x00001004      mov rax, qword [reloc.__gmon_start]   ; [0x3fe8
   0x00001130    5 entry1.init                           |           0x0000100b      test rax, rax
 * 0x00001000   23 sym._init                             |       ,=< 0x0000100e      je 0x1012                             ;[gj]
   0x000011b0    1 sym.__libc_csu_fini                   |       |   0x00001010      call rax
   0x000011b4    9 sym._fini                             |       |   ; CODE XREF from sym._init (0x100e)
   0x00001150   84 sym.__libc_csu_init                   |       `-> 0x00001012      add rsp, 8
   0x00001135   23 sym.main                              \           0x00001016      ret
   0x00001030    6 sym.imp.puts
```



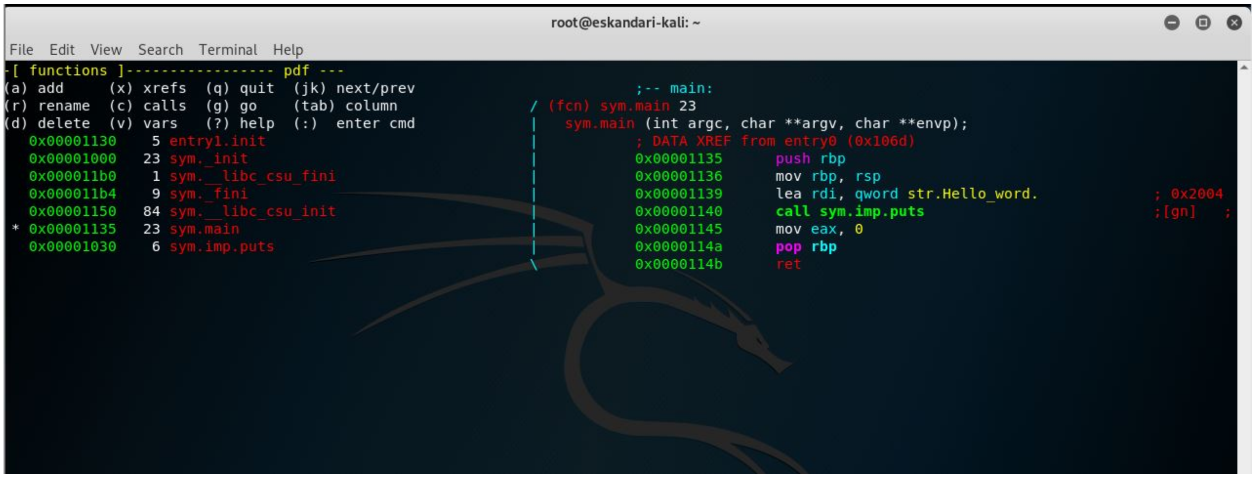
```
-[ functions ]---------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev           / (fcn) sym.__libc_csu_fini 1
(r) rename  (c) calls  (g) go    (tab) column             |   sym.__libc_csu_fini ();
(d) delete  (v) vars   (?) help  (:)  enter cmd           |           ; DATA XREF from entry0 (0x105f)
   0x000010b0   51 sym.register_tm_clones                 \           0x000011b0      ret
   0x000010f0   50 sym.__do_global_dtors_aux
   0x00001040    6 sub.__cxa_finalize_40
   0x00001130    5 entry1.init
   0x00001000   23 sym._init
 * 0x000011b0    1 sym.__libc_csu_fini
   0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
   0x00001030    6 sym.imp.puts
```



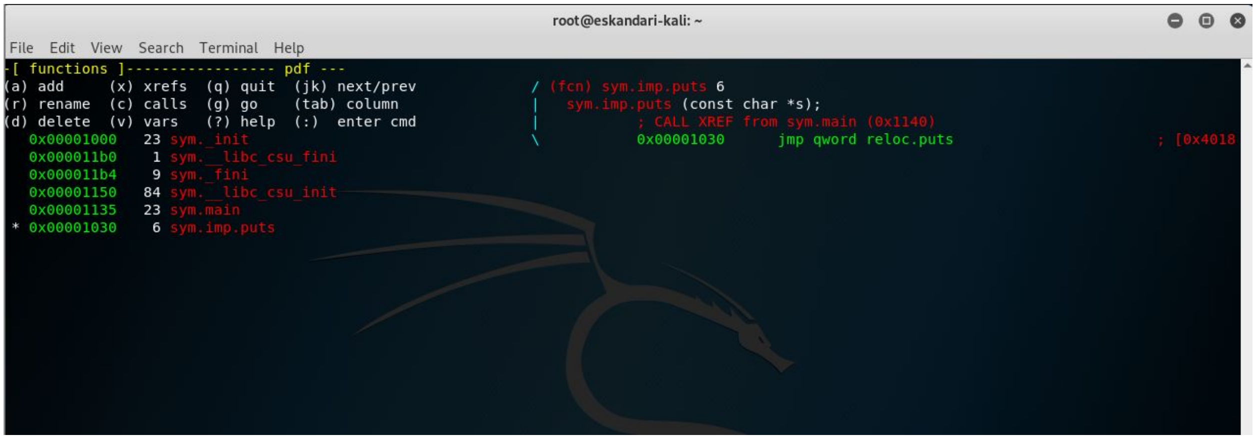
```
-[ functions ]---------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev                       ;-- section..fini:
(r) rename  (c) calls  (g) go    (tab) column             / (fcn) sym._fini 9
(d) delete  (v) vars   (?) help  (:)  enter cmd           |   sym._fini ();
   0x000010f0   50 sym.__do_global_dtors_aux              |           0x000011b4      sub rsp, 8                                 ; [15] -r
   0x00001040    6 sub.__cxa_finalize_40                  |           0x000011b8      add rsp, 8
   0x00001130    5 entry1.init                            \           0x000011bc      ret
   0x00001000   23 sym._init
   0x000011b0    1 sym.__libc_csu_fini
 * 0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
   0x00001030    6 sym.imp.puts
```



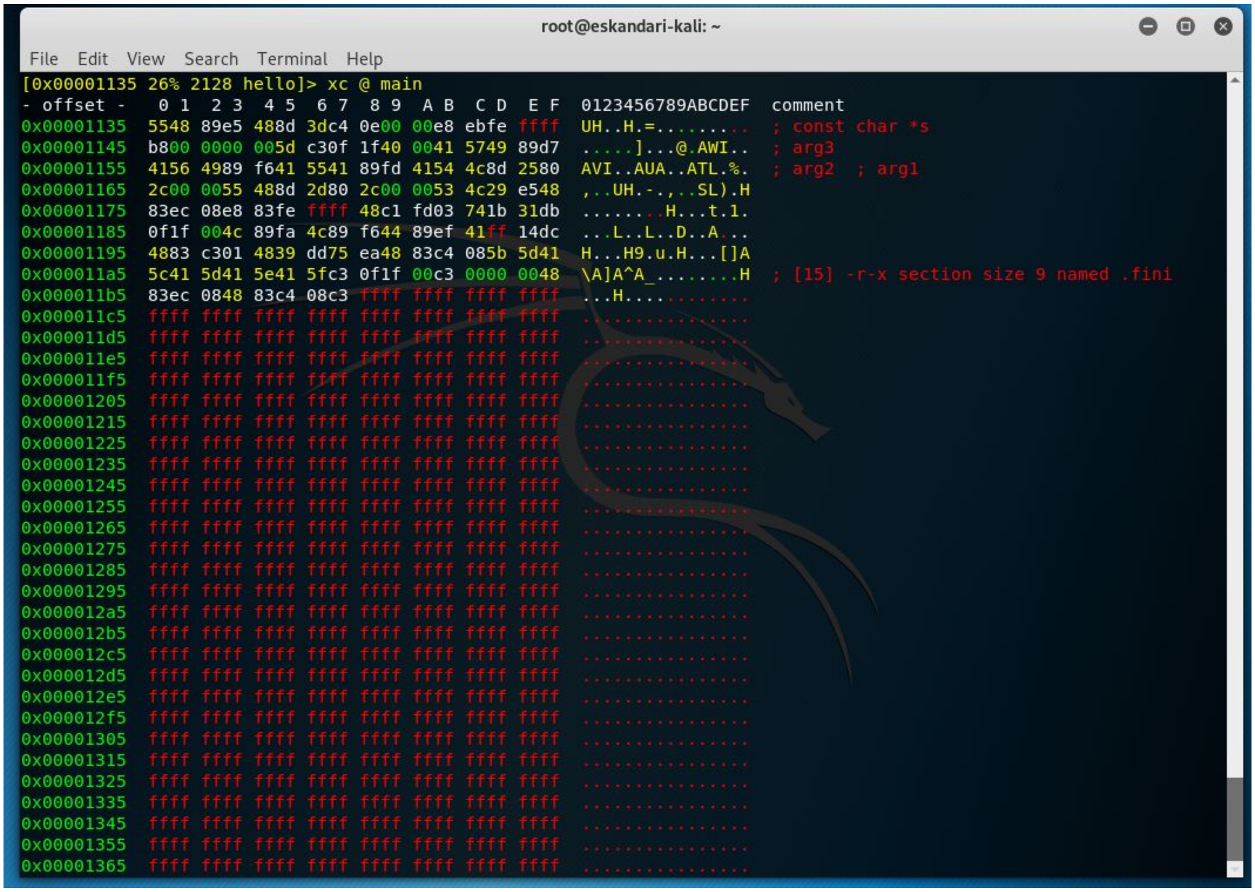
```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev          / (fcn) sym.__libc_csu_init 84
(r) rename  (c) calls  (g) go    (tab) column            |   sym.__libc_csu_init (int arg1, int arg2, int arg3);
(d) delete  (v) vars   (?) help  (:)  enter cmd          |           ; arg int arg1 @ rdi
   0x00001040    6 sub.__cxa_finalize_40                 |           ; arg int arg2 @ rsi
   0x00001130    5 entry1.init                           |           ; arg int arg3 @ rdx
   0x00001000   23 sym._init                             |           ; DATA XREF from entry0 (0x1066)
   0x000011b0    1 sym.__libc_csu_fini                   |           0x00001150      push r15
   0x000011b4    9 sym._fini                             |           0x00001152      mov r15, rdx                         ; arg3
 * 0x00001150   84 sym.__libc_csu_init                   |           0x00001155      push r14
   0x00001135   23 sym.main                              |           0x00001157      mov r14, rsi                         ; arg2
   0x00001030    6 sym.imp.puts                          |           0x0000115a      push r13
                                                         |           0x0000115c      mov r13d, edi                        ; arg1
                                                         |           0x0000115f      push r12
                                                         |           0x00001161      lea r12, qword obj.__frame_dummy_init_array_entry
                                                         |           0x00001168      push rbp
                                                         |           0x00001169      lea rbp, qword obj.__do_global_dtors_aux_fini_array_
                                                         |           0x00001170      push rbx
                                                         |           0x00001171      sub rbp, r12
                                                         |           0x00001174      sub rsp, 8
                                                         |           0x00001178      call sym._init                       ;[gk]
                                                         |           0x0000117d      sar rbp, 3
                                                         |       ,=< 0x00001181      je 0x119e                            ;[gl]
                                                         |       |   0x00001183      xor ebx, ebx
                                                         |       |   0x00001185      nop dword [rax]
                                                         |       |   ; CODE XREF from sym.__libc_csu_init (+0x4c)
                                                         |      .--> 0x00001188      mov rdx, r15
                                                         |      :|   0x0000118b      mov rsi, r14
                                                         |      :|   0x0000118e      mov edi, r13d
                                                         |      :|   0x00001191      call qword [r12 + rbx*8]             ;[gm]
                                                         ..
                                                         |       |   ; CODE XREF from sym.__libc_csu_init (0x1181)
                                                         |       `-> 0x0000119e      add rsp, 8
                                                         |           0x000011a2      pop rbx
                                                         |           0x000011a3      pop rbp
                                                         |           0x000011a4      pop r12
                                                         |           0x000011a6      pop r13
                                                         |           0x000011a8      pop r14
```



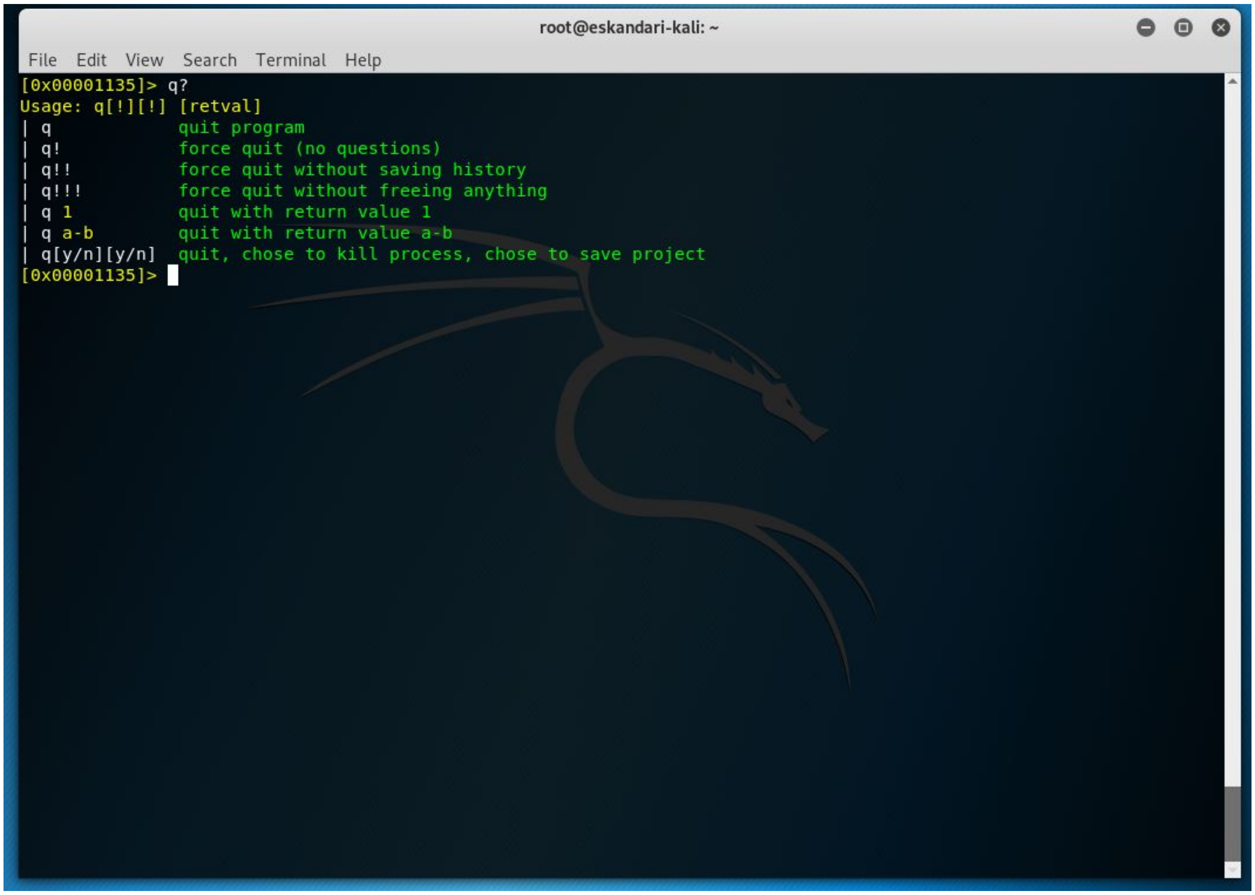
```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev                      ;-- main:
(r) rename  (c) calls  (g) go    (tab) column            / (fcn) sym.main 23
(d) delete  (v) vars   (?) help  (:)  enter cmd          |   sym.main (int argc, char **argv, char **envp);
   0x00001130    5 entry1.init                           |           ; DATA XREF from entry0 (0x106d)
   0x00001000   23 sym._init                             |           0x00001135      push rbp
   0x000011b0    1 sym.__libc_csu_fini                   |           0x00001136      mov rbp, rsp
   0x000011b4    9 sym._fini                             |           0x00001139      lea rdi, qword str.Hello_word.       ; 0x2004
   0x00001150   84 sym.__libc_csu_init                   |           0x00001140      call sym.imp.puts                    ;[gn]   ;
 * 0x00001135   23 sym.main                              |           0x00001145      mov eax, 0
   0x00001030    6 sym.imp.puts                          |           0x0000114a      pop rbp
                                                         \           0x0000114b      ret
```



```
-[ functions ]----------------- pdf ---
(a) add     (x) xrefs  (q) quit  (jk) next/prev           / (fcn) sym.imp.puts 6
(r) rename  (c) calls  (g) go    (tab) column             |   sym.imp.puts (const char *s);
(d) delete  (v) vars   (?) help  (:)  enter cmd           |          ; CALL XREF from sym.main (0x1140)
   0x00001000   23 sym._init                              \          0x00001030      jmp qword reloc.puts                       ; [0x4018
   0x000011b0    1 sym.__libc_csu_fini
   0x000011b4    9 sym._fini
   0x00001150   84 sym.__libc_csu_init
   0x00001135   23 sym.main
 * 0x00001030    6 sym.imp.puts
```



```
[0x00001135 26% 2128 hello]> xc @ main
- offset -   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0123456789ABCDEF  comment
0x00001135  5548 89e5 488d 3dc4 0e00 00e8 ebfe ffff  UH..H.=.........  ; const char *s
0x00001145  b800 0000 005d c30f 1f40 0041 5749 89d7  .....]...@.AWI..  ; arg3
0x00001155  4156 4989 f641 5541 89fd 4154 4c8d 2580  AVI..AUA..ATL.%.  ; arg2  ; arg1
0x00001165  2c00 0055 488d 2d80 2c00 0053 4c29 e548  ,..UH.-.,..SL).H
0x00001175  83ec 08e8 83fe ffff 48c1 fd03 741b 31db  ........H...t.1.
0x00001185  0f1f 004c 89fa 4c89 f644 89ef 41ff 14dc  ...L..L..D..A...
0x00001195  4883 c301 4839 dd75 ea48 83c4 085b 5d41  H...H9.u.H...[]A
0x000011a5  5c41 5d41 5e41 5fc3 0f1f 00c3 0000 0048  \A]A^A_........H  ; [15] -r-x section size 9 named .fini
0x000011b5  83ec 0848 83c4 08c3 ffff ffff ffff ffff  ...H............
0x000011c5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000011d5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000011e5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000011f5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001205  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001215  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001225  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001235  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001245  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001255  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001265  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001275  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001285  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001295  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012a5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012b5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012c5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012d5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012e5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x000012f5  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001305  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001315  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001325  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001335  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001345  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001355  ffff ffff ffff ffff ffff ffff ffff ffff  ................
0x00001365  ffff ffff ffff ffff ffff ffff ffff ffff  ................
```



```
[0x00001135]> q?
Usage: q[!][!] [retval]
| q             quit program
| q!            force quit (no questions)
| q!!           force quit without saving history
| q!!!          force quit without freeing anything
| q 1           quit with return value 1
| q a-b         quit with return value a-b
| q[y/n][y/n]   quit, chose to kill process, chose to save project
[0x00001135]> 
```