UNIX
Developer’s tools

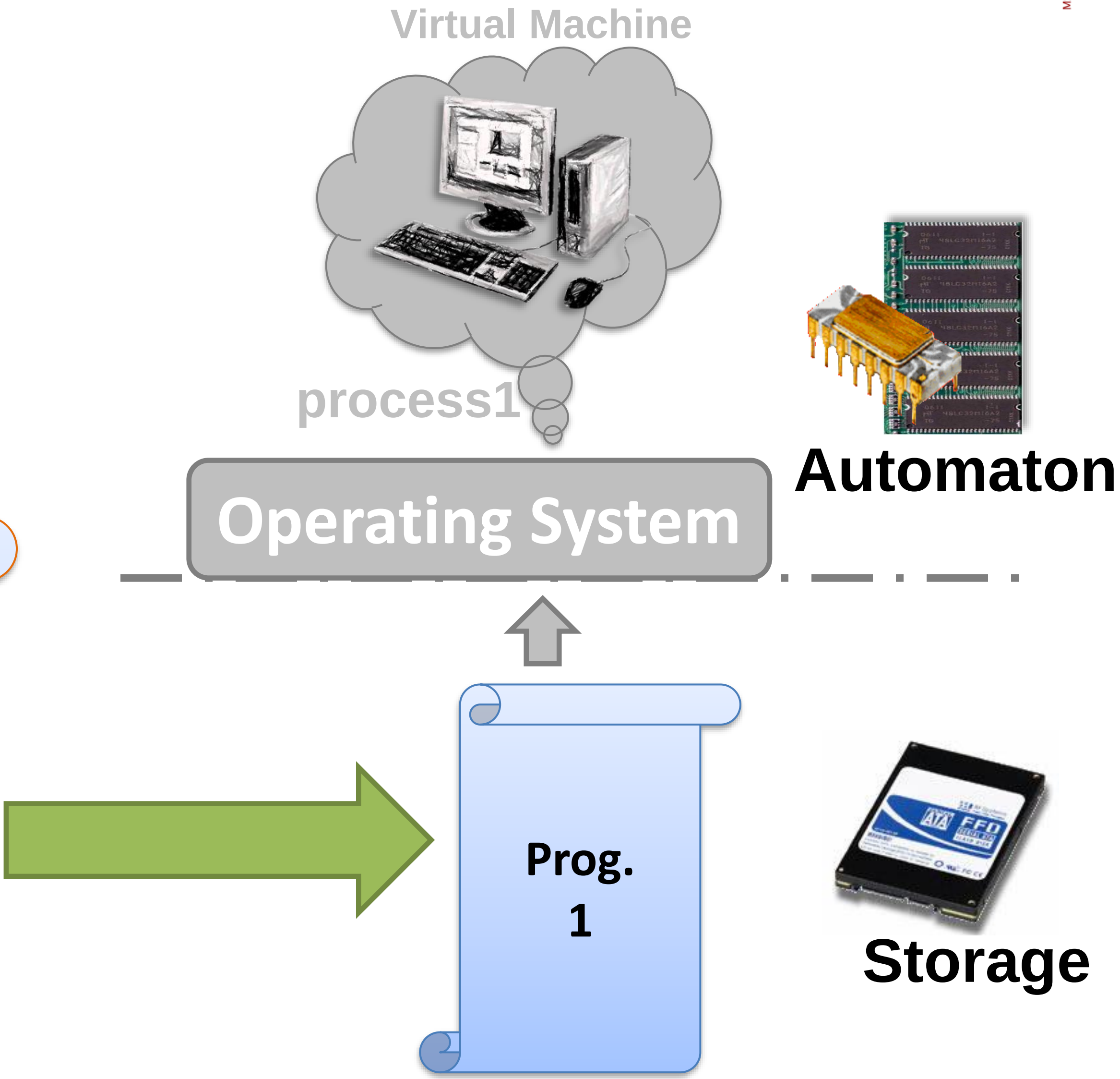
Content

- compile time tools

- run time tools

Produce Prog. files

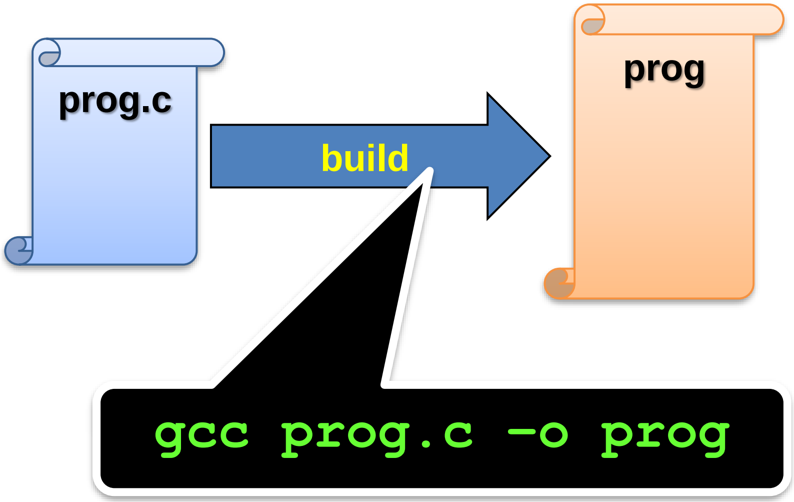
```
int a = 33;

int main()
{
   int b = 42;
   printf("b=%d\n" ,b);
   return 0;
}
```



Compile-time

ONE build command

- `gcc prog.c` → Create executable file named «a.out»
- `gcc prog.c -o prog` → Create executable file named «prog»

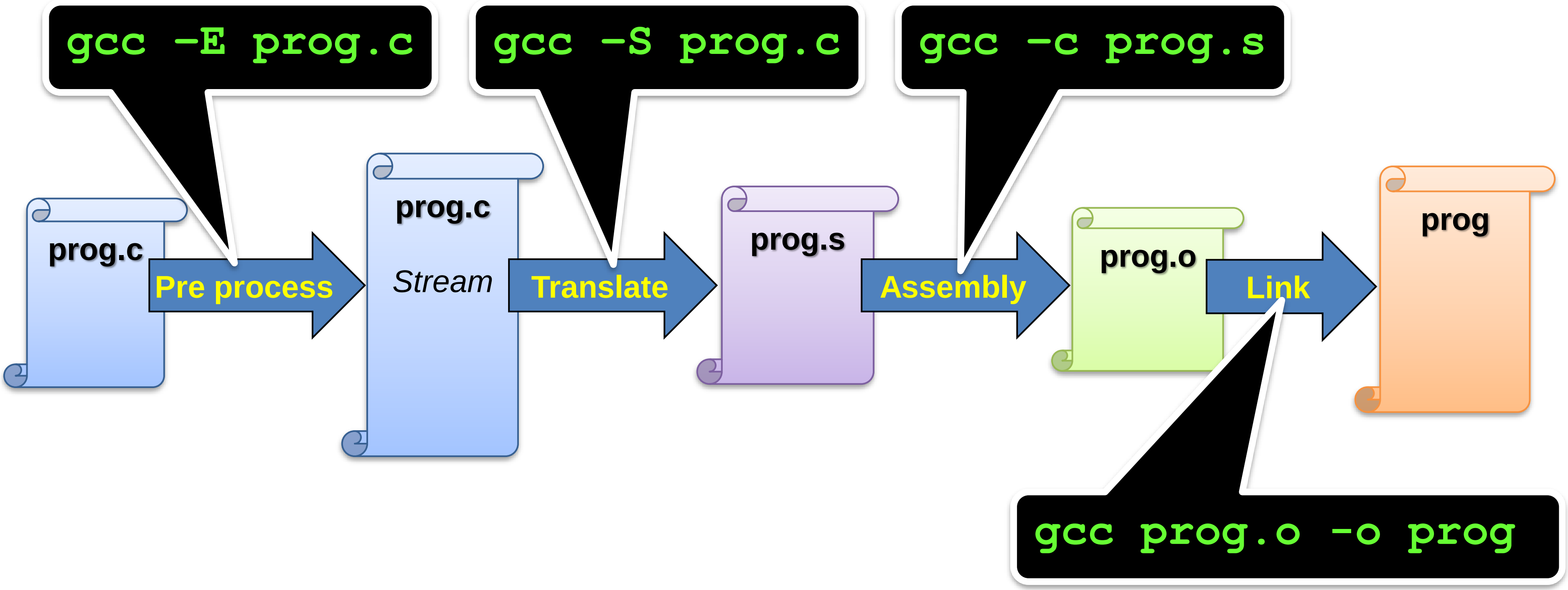


```
gcc prog.c -o prog
```

Compile-time

FOUR steps build

- `gcc -E prog.c` → transform # directives (stream)
- `gcc -S prog.c` → translate C text (.c) to assembly text (.s)
- `gcc -c prog.s` → translate ASM text (.s) to binary (.o)
- `gcc prog.o` → Create binary executable (a.out)



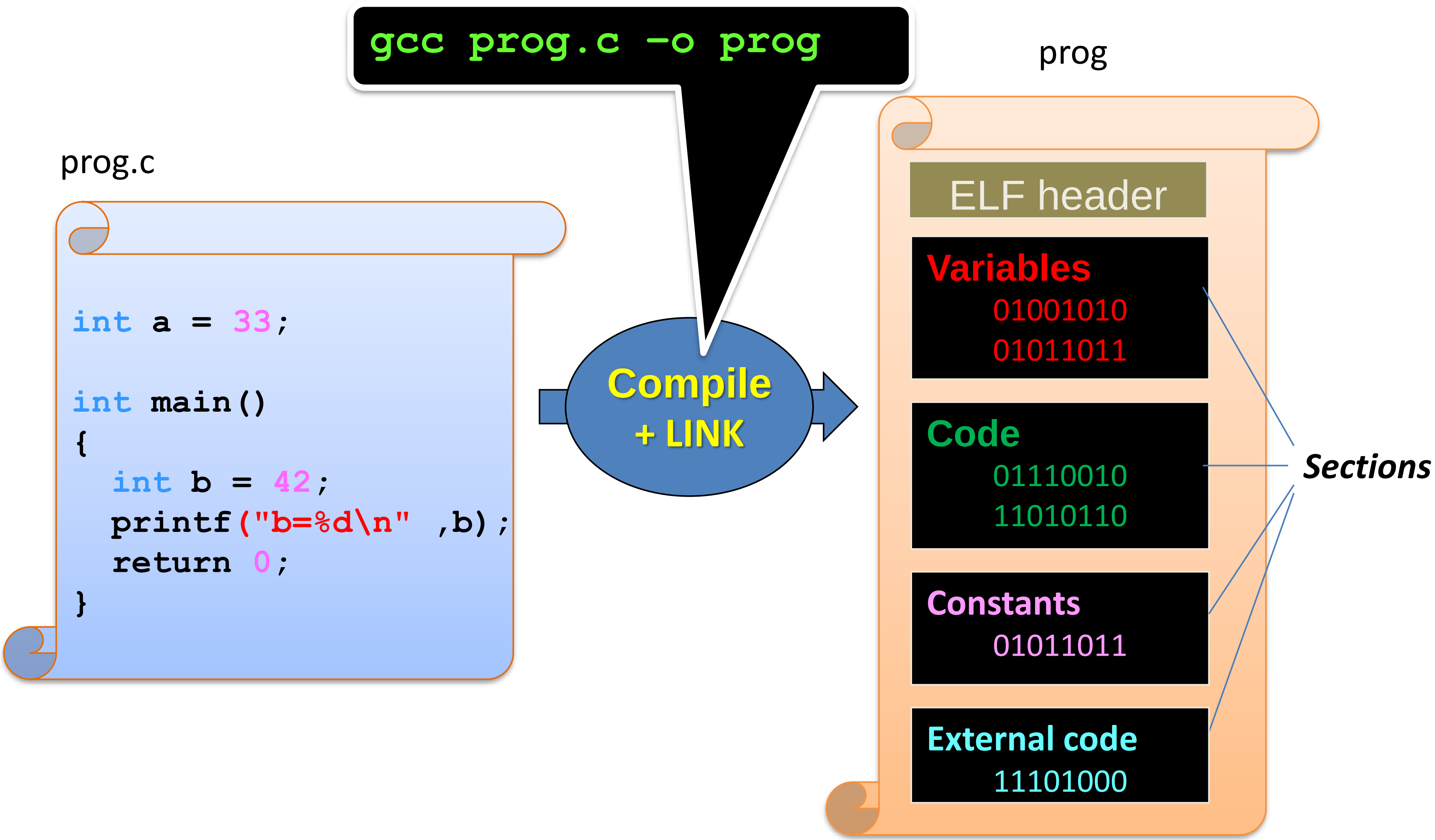
```
gcc -E prog.c
```

```
gcc -S prog.c
```

```
gcc -c prog.s
```

```
gcc prog.o -o prog
```

Content of an executable



```
gcc prog.c -o prog
```

prog.c

```
int a = 33;

int main()
{
   int b = 42;
   printf("b=%d\n" ,b);
   return 0;
}
```

prog

ELF header

Variables
01001010
01011011

Code
01110010
11010110

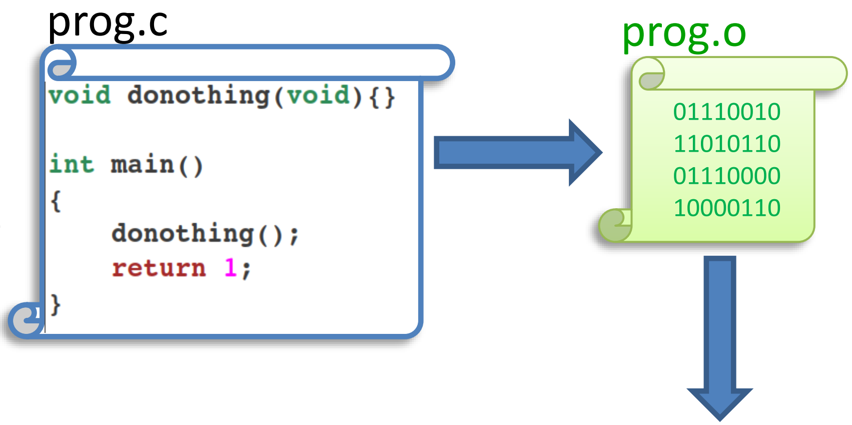
Constants
01011011

External code
11101000

Sections

Compile time ELF dump

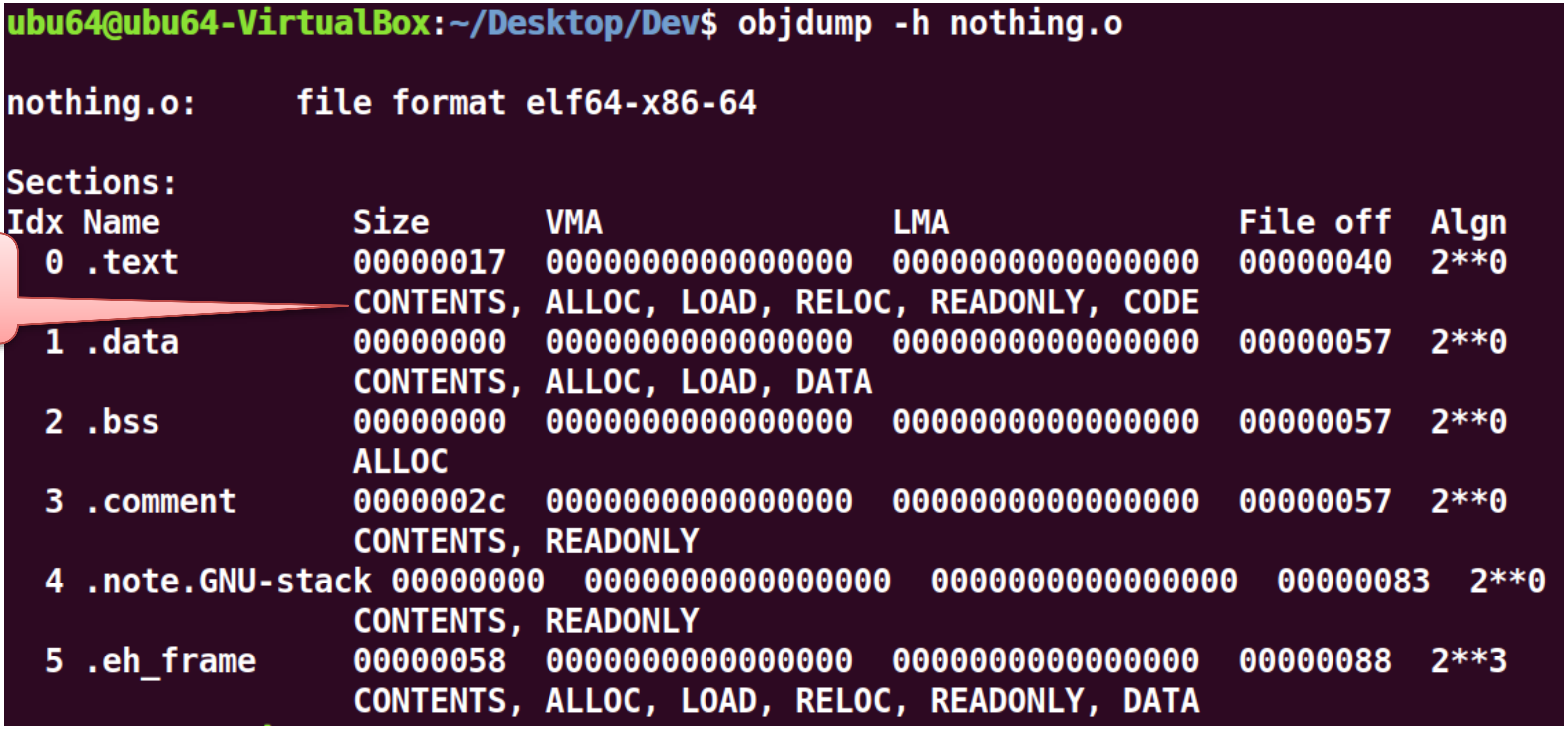
- **objdump -h**
  - Many options
  - Works on .O and executable


prog.c

```
void donothing(void){}

int main()
{
    donothing();
    return 1;
}
```

prog.o

```
01110010
11010110
01110000
10000110
```

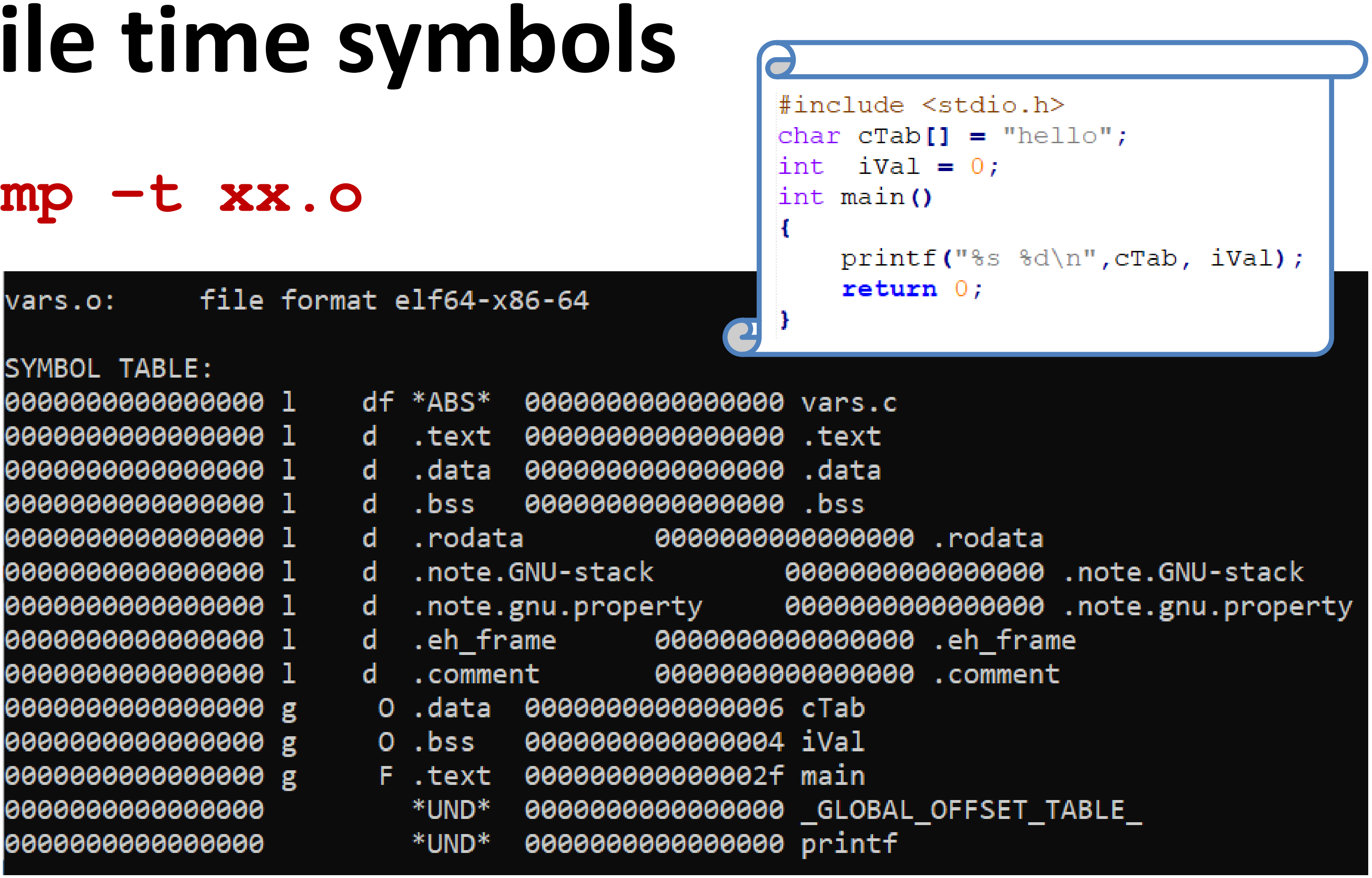


```
ubu64@ubu64-VirtualBox:~/Desktop/Dev$ objdump -h nothing.o

nothing.o:     file format elf64-x86-64

Sections:
Idx Name          Size      VMA               LMA               File off  Algn
  0 .text         00000017  0000000000000000  0000000000000000  00000040  2**0
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  0000000000000000  0000000000000000  00000057  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  0000000000000000  0000000000000000  00000057  2**0
                  ALLOC
  3 .comment      0000002c  0000000000000000  0000000000000000  00000057  2**0
                  CONTENTS, READONLY
  4 .note.GNU-stack 00000000  0000000000000000  0000000000000000  00000083  2**0
                  CONTENTS, READONLY
  5 .eh_frame     00000058  0000000000000000  0000000000000000  00000088  2**3
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, DATA
```

Runtime information

Compile time symbols

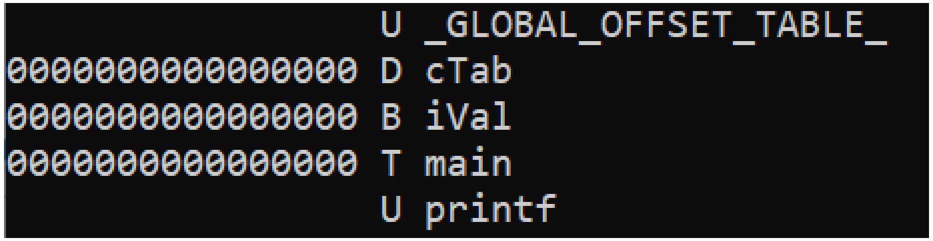
- **objdump –t xx.o**



```c
#include <stdio.h>
char cTab[] = "hello";
int  iVal = 0;
int main()
{
    printf("%s %d\n",cTab, iVal);
    return 0;
}
```

```
vars.o:     file format elf64-x86-64

SYMBOL TABLE:
0000000000000000 l    df *ABS*  0000000000000000 vars.c
0000000000000000 l    d  .text  0000000000000000 .text
0000000000000000 l    d  .data  0000000000000000 .data
0000000000000000 l    d  .bss   0000000000000000 .bss
0000000000000000 l    d  .rodata        0000000000000000 .rodata
0000000000000000 l    d  .note.GNU-stack        0000000000000000 .note.GNU-stack
0000000000000000 l    d  .note.gnu.property     0000000000000000 .note.gnu.property
0000000000000000 l    d  .eh_frame      0000000000000000 .eh_frame
0000000000000000 l    d  .comment       0000000000000000 .comment
0000000000000000 g     O .data  0000000000000006 cTab
0000000000000000 g     O .bss   0000000000000004 iVal
0000000000000000 g     F .text  000000000000002f main
0000000000000000         *UND*  0000000000000000 _GLOBAL_OFFSET_TABLE_
0000000000000000         *UND*  0000000000000000 printf
```

- **nm xx.o**



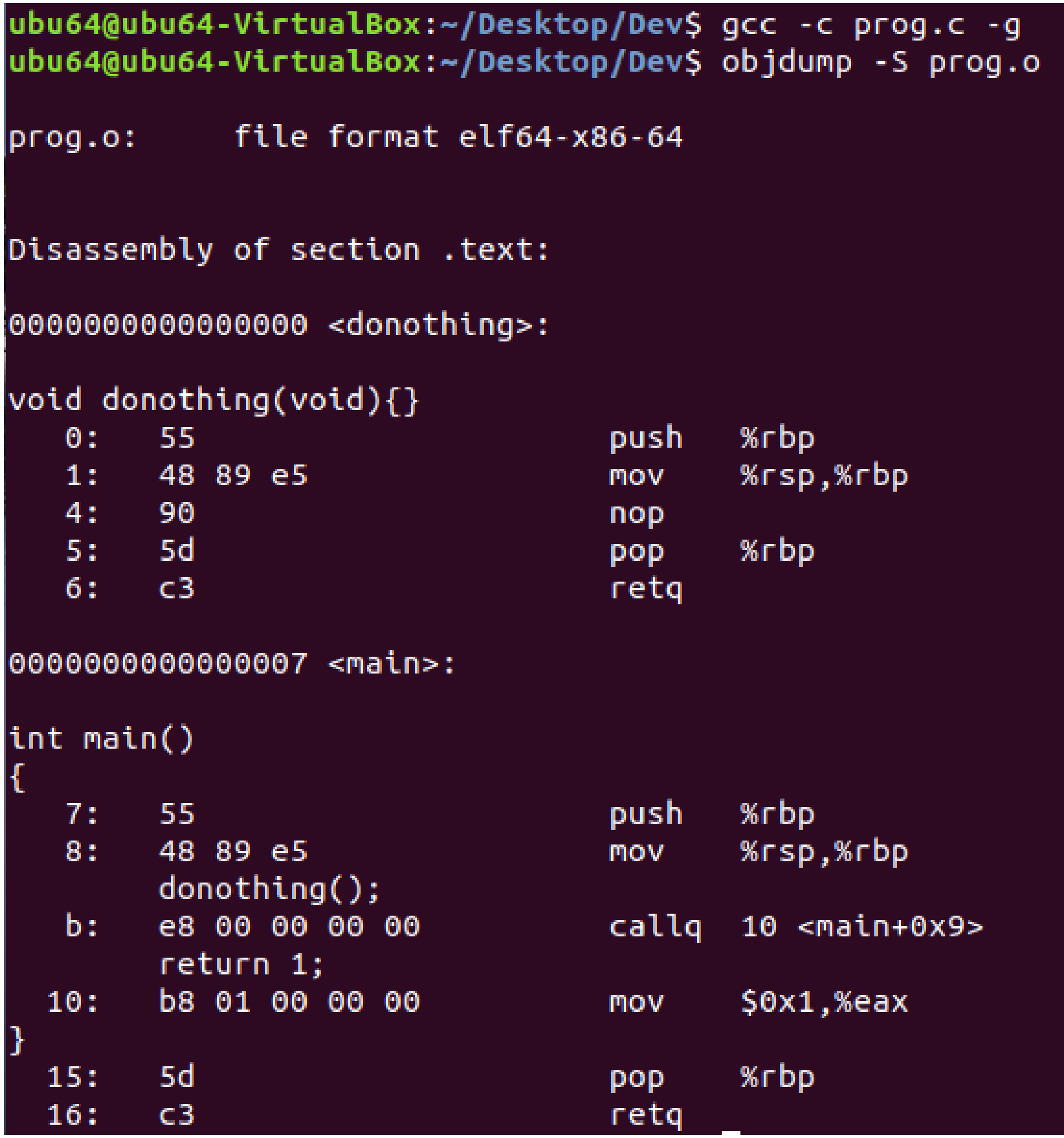
```
                 U _GLOBAL_OFFSET_TABLE_
0000000000000000 D cTab
0000000000000000 B iVal
0000000000000000 T main
                 U printf
```

Compile-time assembly dump

- **objdump -S**
  - Format
  - Sections
  - Data
  - Assembly code

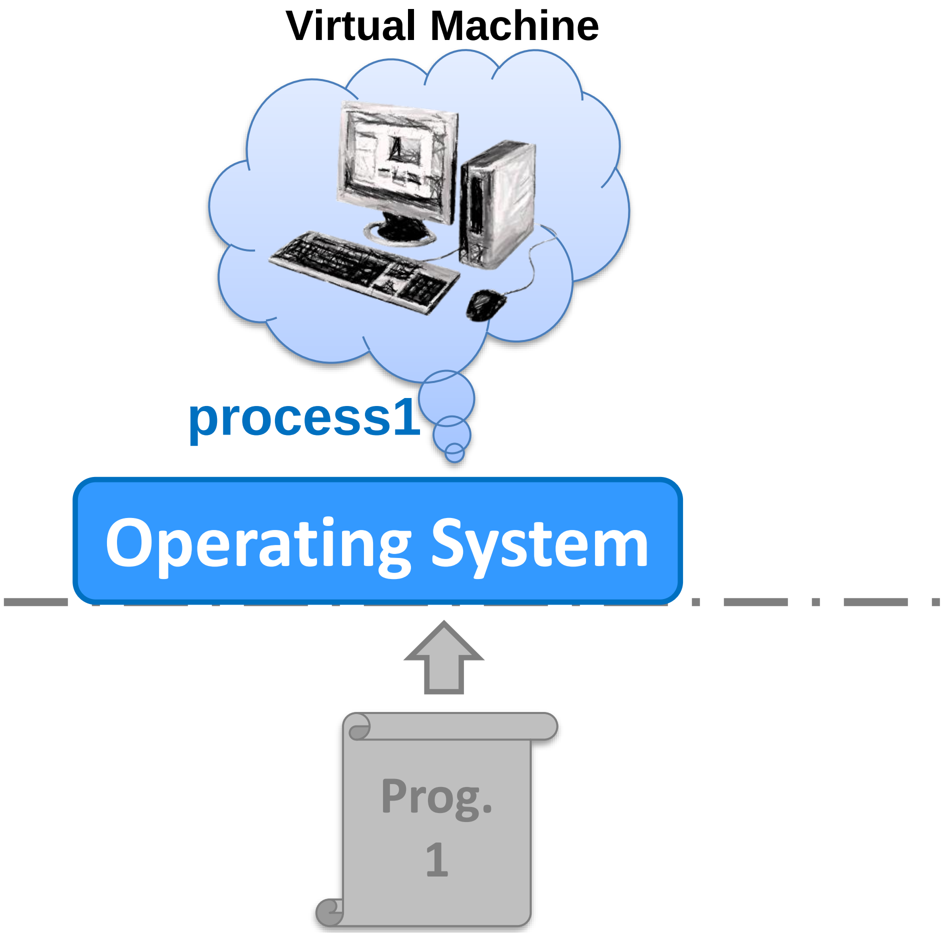
prog.c

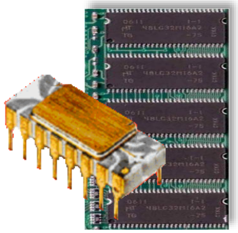
```c
void donothing(void){}

int main()
{
    donothing();
    return 1;
}
```



```
ubu64@ubu64-VirtualBox:~/Desktop/Dev$ gcc -c prog.c -g
ubu64@ubu64-VirtualBox:~/Desktop/Dev$ objdump -S prog.o

prog.o:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <donothing>:

void donothing(void){}
   0:   55                      push   %rbp
   1:   48 89 e5                mov    %rsp,%rbp
   4:   90                      nop
   5:   5d                      pop    %rbp
   6:   c3                      retq

0000000000000007 <main>:

int main()
{
   7:   55                      push   %rbp
   8:   48 89 e5                mov    %rsp,%rbp
        donothing();
   b:   e8 00 00 00 00          callq  10 <main+0x9>
        return 1;
  10:   b8 01 00 00 00          mov    $0x1,%eax
}
  15:   5d                      pop    %rbp
  16:   c3                      retq
```

Execute ELF files

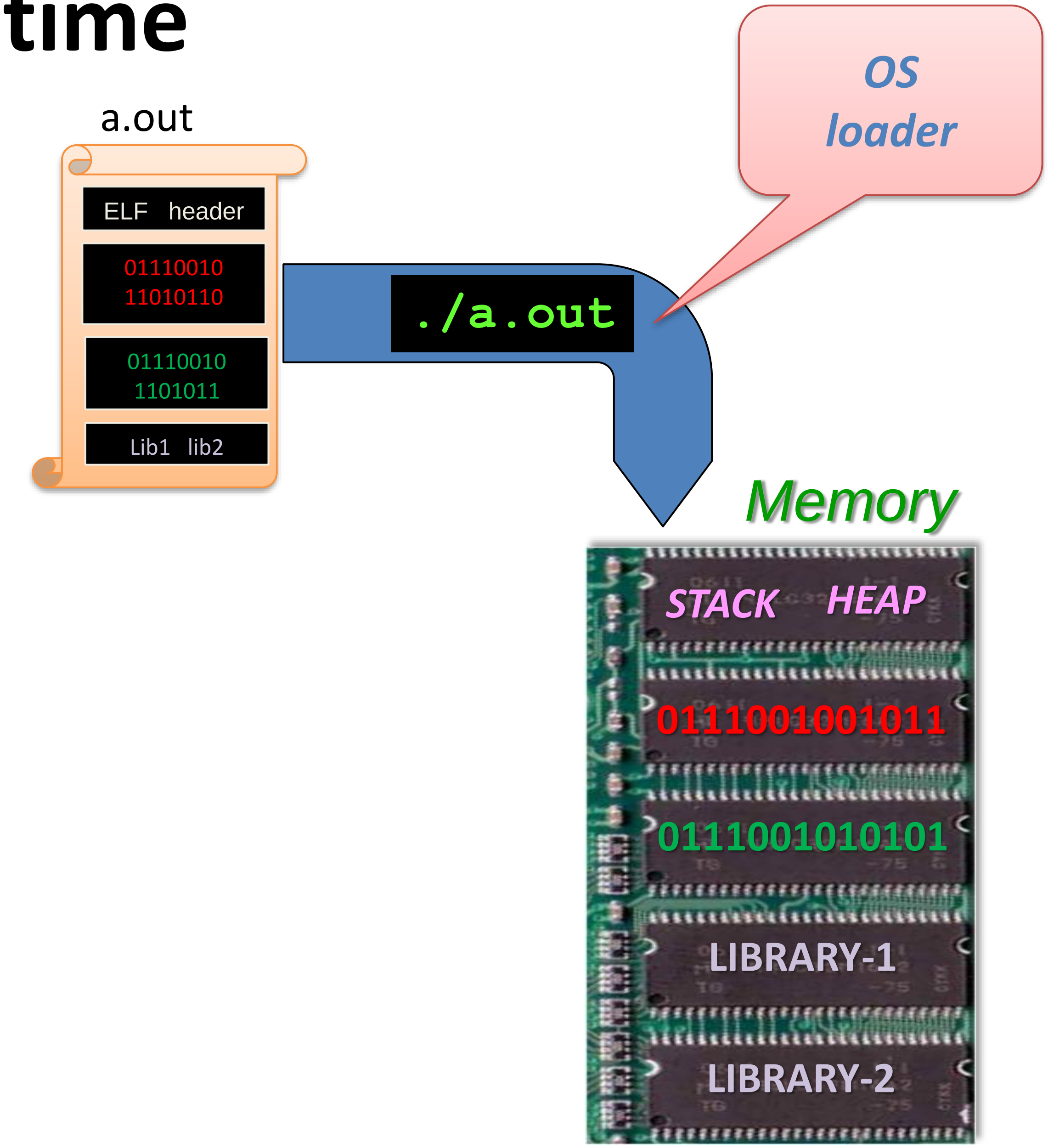




Automaton

Storage

Run-time



a.out

ELF header

01110010
11010110

01110010
1101011

Lib1 lib2

```
./a.out
```

OS
loader

Memory

STACK HEAP

0111001001011

0111001010101

LIBRARY-1

LIBRARY-2

Run-time memory status

```
pmap -d `pidof a.out`
```

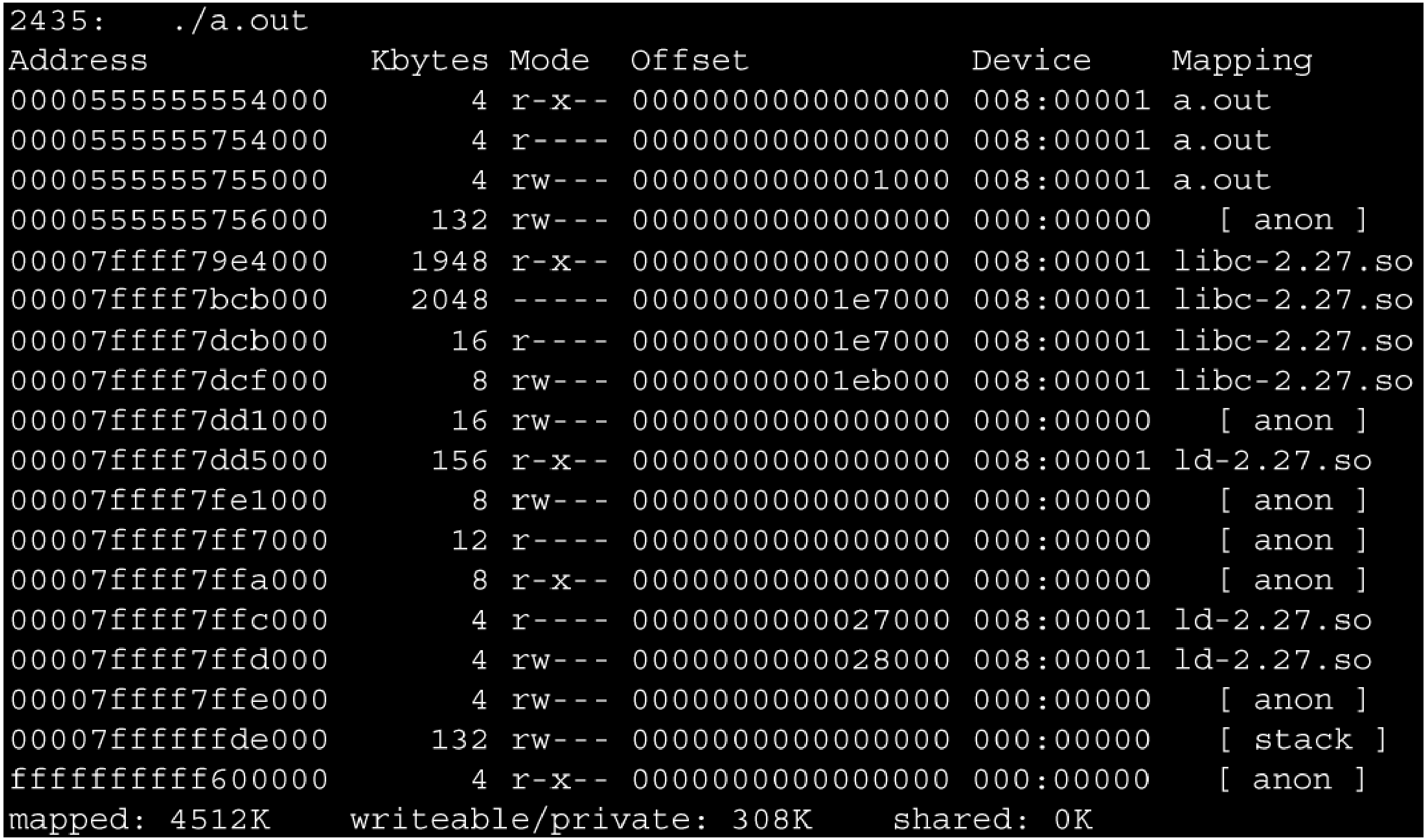
Addresses

Size

Mode

Mapping

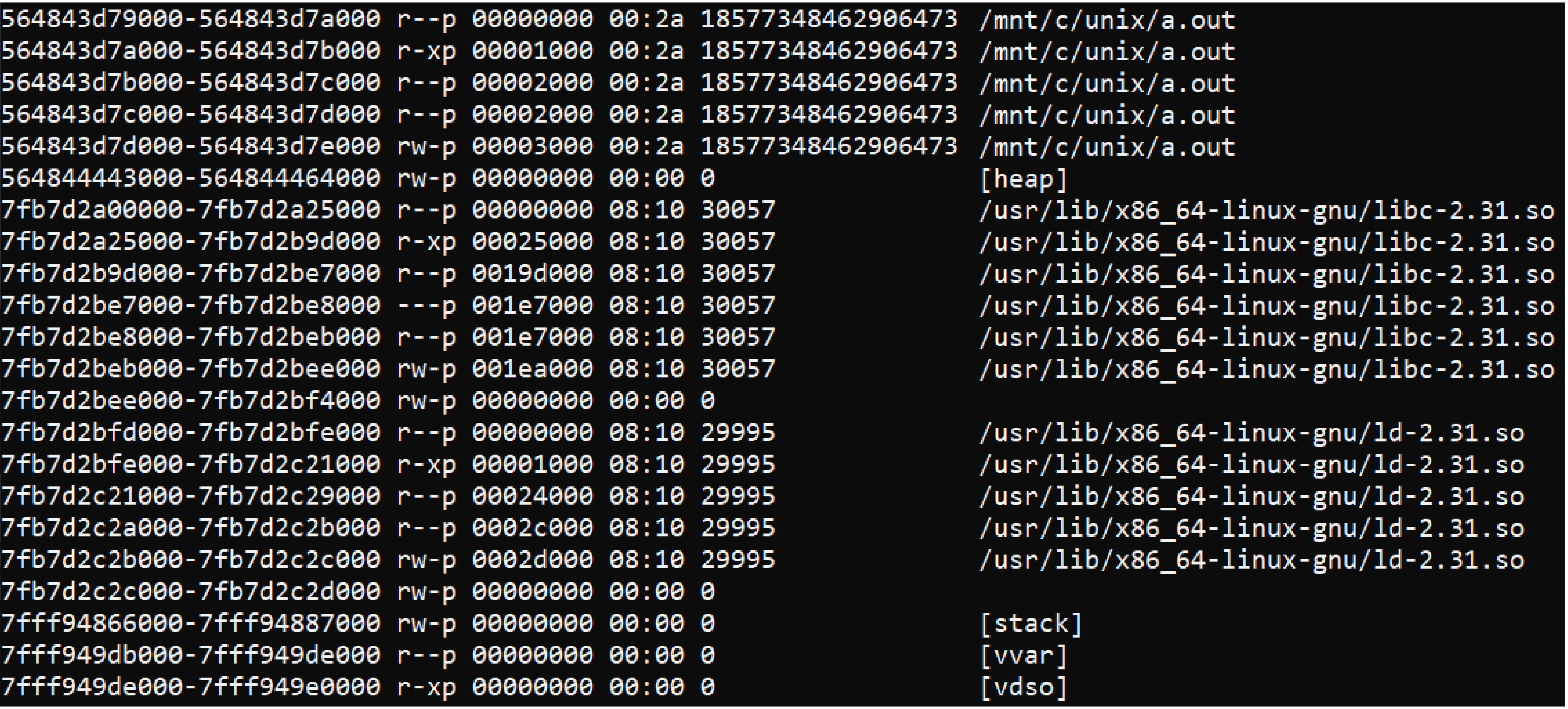
Etc.



```
2435:   ./a.out
Address           Kbytes Mode  Offset           Device    Mapping
0000555555554000       4 r-x-- 0000000000000000 008:00001 a.out
0000555555754000       4 r---- 0000000000000000 008:00001 a.out
0000555555755000       4 rw--- 0000000000001000 008:00001 a.out
0000555555756000     132 rw--- 0000000000000000 000:00000   [ anon ]
00007ffff79e4000    1948 r-x-- 0000000000000000 008:00001 libc-2.27.so
00007ffff7bcb000    2048 ----- 00000000001e7000 008:00001 libc-2.27.so
00007ffff7dcb000      16 r---- 00000000001e7000 008:00001 libc-2.27.so
00007ffff7dcf000       8 rw--- 00000000001eb000 008:00001 libc-2.27.so
00007ffff7dd1000      16 rw--- 0000000000000000 000:00000   [ anon ]
00007ffff7dd5000     156 r-x-- 0000000000000000 008:00001 ld-2.27.so
00007ffff7fe1000       8 rw--- 0000000000000000 000:00000   [ anon ]
00007ffff7ff7000      12 r---- 0000000000000000 000:00000   [ anon ]
00007ffff7ffa000       8 r-x-- 0000000000000000 000:00000   [ anon ]
00007ffff7ffc000       4 r---- 0000000000027000 008:00001 ld-2.27.so
00007ffff7ffd000       4 rw--- 0000000000028000 008:00001 ld-2.27.so
00007ffff7ffe000       4 rw--- 0000000000000000 000:00000   [ anon ]
00007ffffffde000     132 rw--- 0000000000000000 000:00000   [ stack ]
ffffffffff600000       4 r-x-- 0000000000000000 000:00000   [ anon ]
mapped: 4512K    writeable/private: 308K    shared: 0K
```

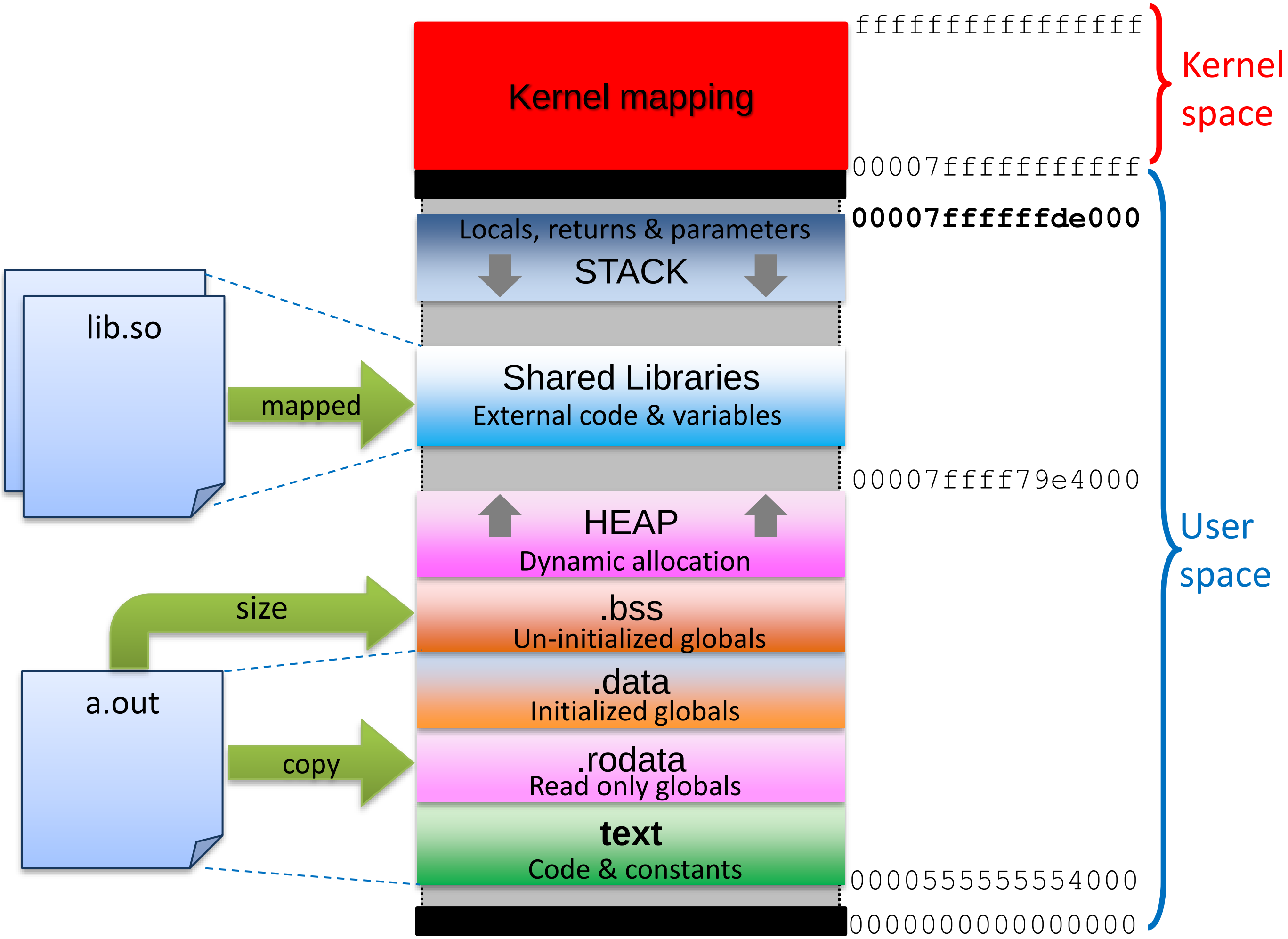
Run-time memory status (bis)

```
cat /proc/`pidof a.out`/maps
```



```
564843d79000-564843d7a000 r--p 00000000 00:2a 18577348462906473 /mnt/c/unix/a.out
564843d7a000-564843d7b000 r-xp 00001000 00:2a 18577348462906473 /mnt/c/unix/a.out
564843d7b000-564843d7c000 r--p 00002000 00:2a 18577348462906473 /mnt/c/unix/a.out
564843d7c000-564843d7d000 r--p 00002000 00:2a 18577348462906473 /mnt/c/unix/a.out
564843d7d000-564843d7e000 rw-p 00003000 00:2a 18577348462906473 /mnt/c/unix/a.out
564844443000-564844464000 rw-p 00000000 00:00 0                 [heap]
7fb7d2a00000-7fb7d2a25000 r--p 00000000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2a25000-7fb7d2b9d000 r-xp 00025000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2b9d000-7fb7d2be7000 r--p 0019d000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2be7000-7fb7d2be8000 ---p 001e7000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2be8000-7fb7d2beb000 r--p 001e7000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2beb000-7fb7d2bee000 rw-p 001ea000 08:10 30057             /usr/lib/x86_64-linux-gnu/libc-2.31.so
7fb7d2bee000-7fb7d2bf4000 rw-p 00000000 00:00 0
7fb7d2bfd000-7fb7d2bfe000 r--p 00000000 08:10 29995             /usr/lib/x86_64-linux-gnu/ld-2.31.so
7fb7d2bfe000-7fb7d2c21000 r-xp 00001000 08:10 29995             /usr/lib/x86_64-linux-gnu/ld-2.31.so
7fb7d2c21000-7fb7d2c29000 r--p 00024000 08:10 29995             /usr/lib/x86_64-linux-gnu/ld-2.31.so
7fb7d2c2a000-7fb7d2c2b000 r--p 0002c000 08:10 29995             /usr/lib/x86_64-linux-gnu/ld-2.31.so
7fb7d2c2b000-7fb7d2c2c000 rw-p 0002d000 08:10 29995             /usr/lib/x86_64-linux-gnu/ld-2.31.so
7fb7d2c2c000-7fb7d2c2d000 rw-p 00000000 00:00 0
7fff94866000-7fff94887000 rw-p 00000000 00:00 0                 [stack]
7fff949db000-7fff949de000 r--p 00000000 00:00 0                 [vvar]
7fff949de000-7fff949e0000 r-xp 00000000 00:00 0                 [vdso]
```

Anatomy of a process

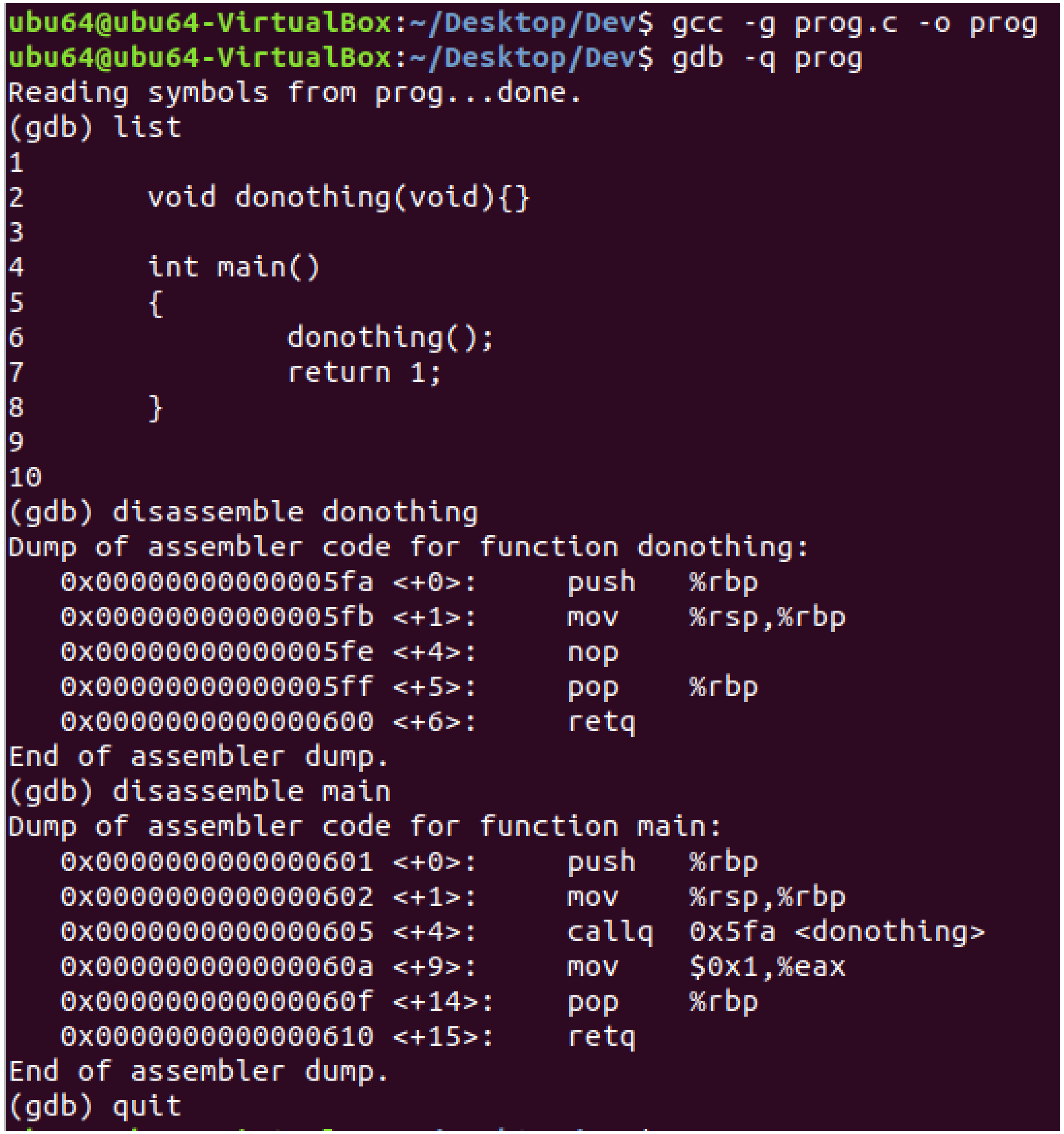


Run-time assembly dump

- gdb
  - Source (gcc –g)
  - symbols
  - Assembly Code
  - Breakpoint
  - Etc.

prog.c

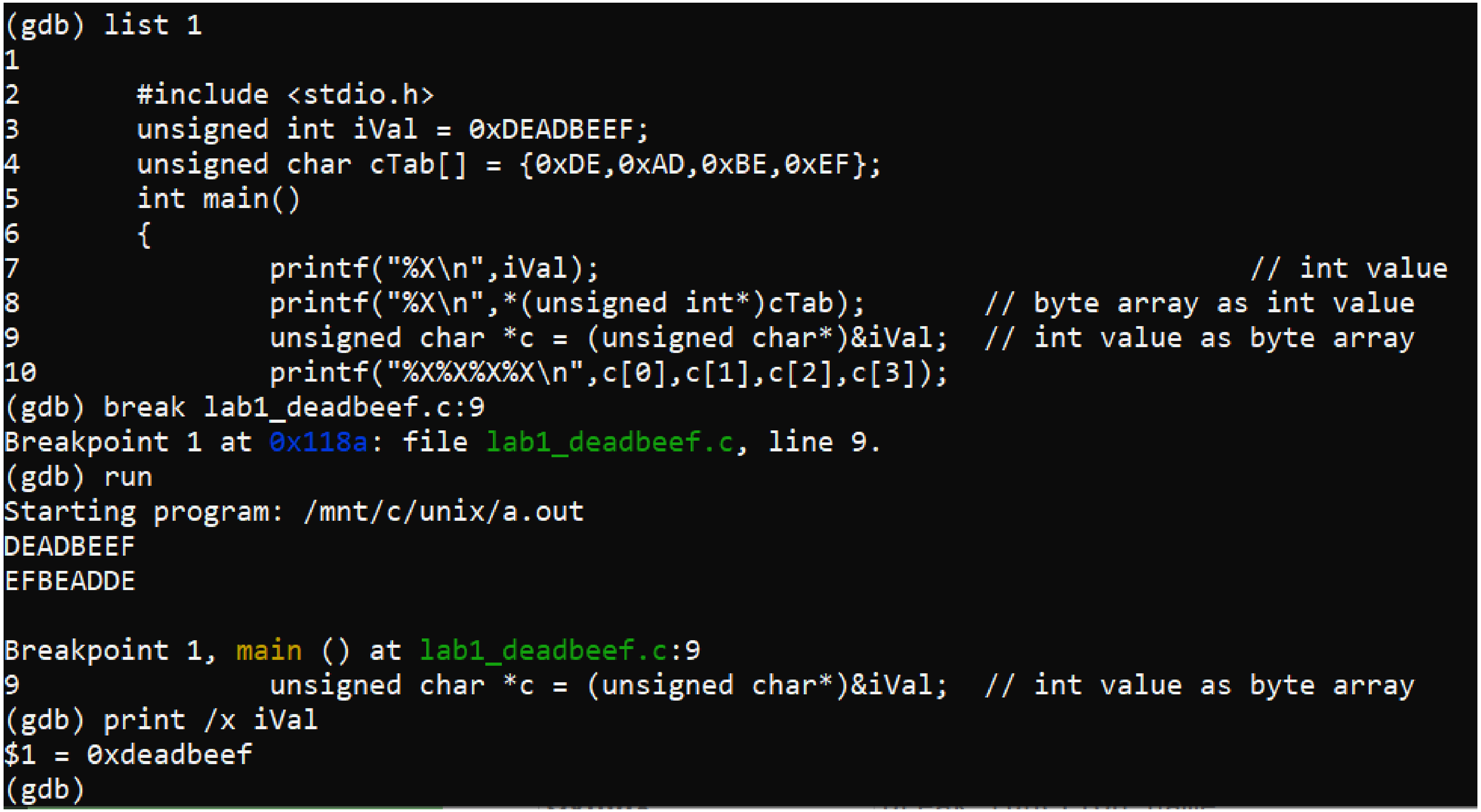
```c
void donothing(void){}

int main()
{
    donothing();
    return 1;
}
```



```
ubu64@ubu64-VirtualBox:~/Desktop/Dev$ gcc -g prog.c -o prog
ubu64@ubu64-VirtualBox:~/Desktop/Dev$ gdb -q prog
Reading symbols from prog...done.
(gdb) list
1
2        void donothing(void){}
3
4        int main()
5        {
6                donothing();
7                return 1;
8        }
9
10
(gdb) disassemble donothing
Dump of assembler code for function donothing:
   0x00000000000005fa <+0>:     push   %rbp
   0x00000000000005fb <+1>:     mov    %rsp,%rbp
   0x00000000000005fe <+4>:     nop
   0x00000000000005ff <+5>:     pop    %rbp
   0x0000000000000600 <+6>:     retq
End of assembler dump.
(gdb) disassemble main
Dump of assembler code for function main:
   0x0000000000000601 <+0>:     push   %rbp
   0x0000000000000602 <+1>:     mov    %rsp,%rbp
   0x0000000000000605 <+4>:     callq  0x5fa <donothing>
   0x000000000000060a <+9>:     mov    $0x1,%eax
   0x000000000000060f <+14>:    pop    %rbp
   0x0000000000000610 <+15>:    retq
End of assembler dump.
(gdb) quit
```

Run-time Break and dump

- gdb
  - list, break, print



```
(gdb) list 1
1
2       #include <stdio.h>
3       unsigned int iVal = 0xDEADBEEF;
4       unsigned char cTab[] = {0xDE,0xAD,0xBE,0xEF};
5       int main()
6       {
7               printf("%X\n",iVal);                                    // int value
8               printf("%X\n",*(unsigned int*)cTab);        // byte array as int value
9               unsigned char *c = (unsigned char*)&iVal;   // int value as byte array
10              printf("%X%X%X%X\n",c[0],c[1],c[2],c[3]);
(gdb) break lab1_deadbeef.c:9
Breakpoint 1 at 0x118a: file lab1_deadbeef.c, line 9.
(gdb) run
Starting program: /mnt/c/unix/a.out
DEADBEEF
EFBEADDE

Breakpoint 1, main () at lab1_deadbeef.c:9
9               unsigned char *c = (unsigned char*)&iVal;   // int value as byte array
(gdb) print /x iVal
$1 = 0xdeadbeef
(gdb)
```