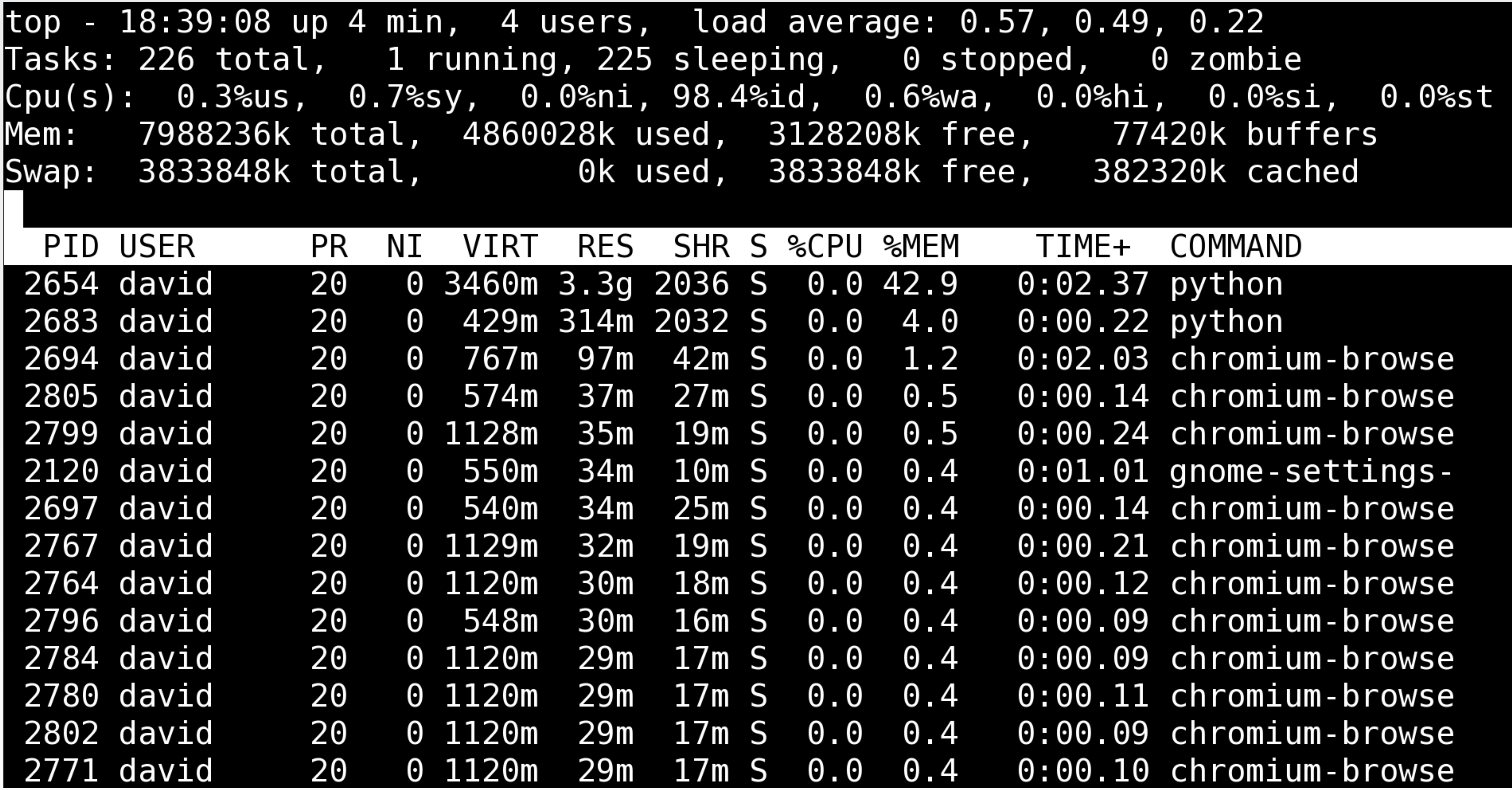
“Dude – Where's My RAM?” A deep dive into how Python uses memory.

David Malcolm, Red Hat

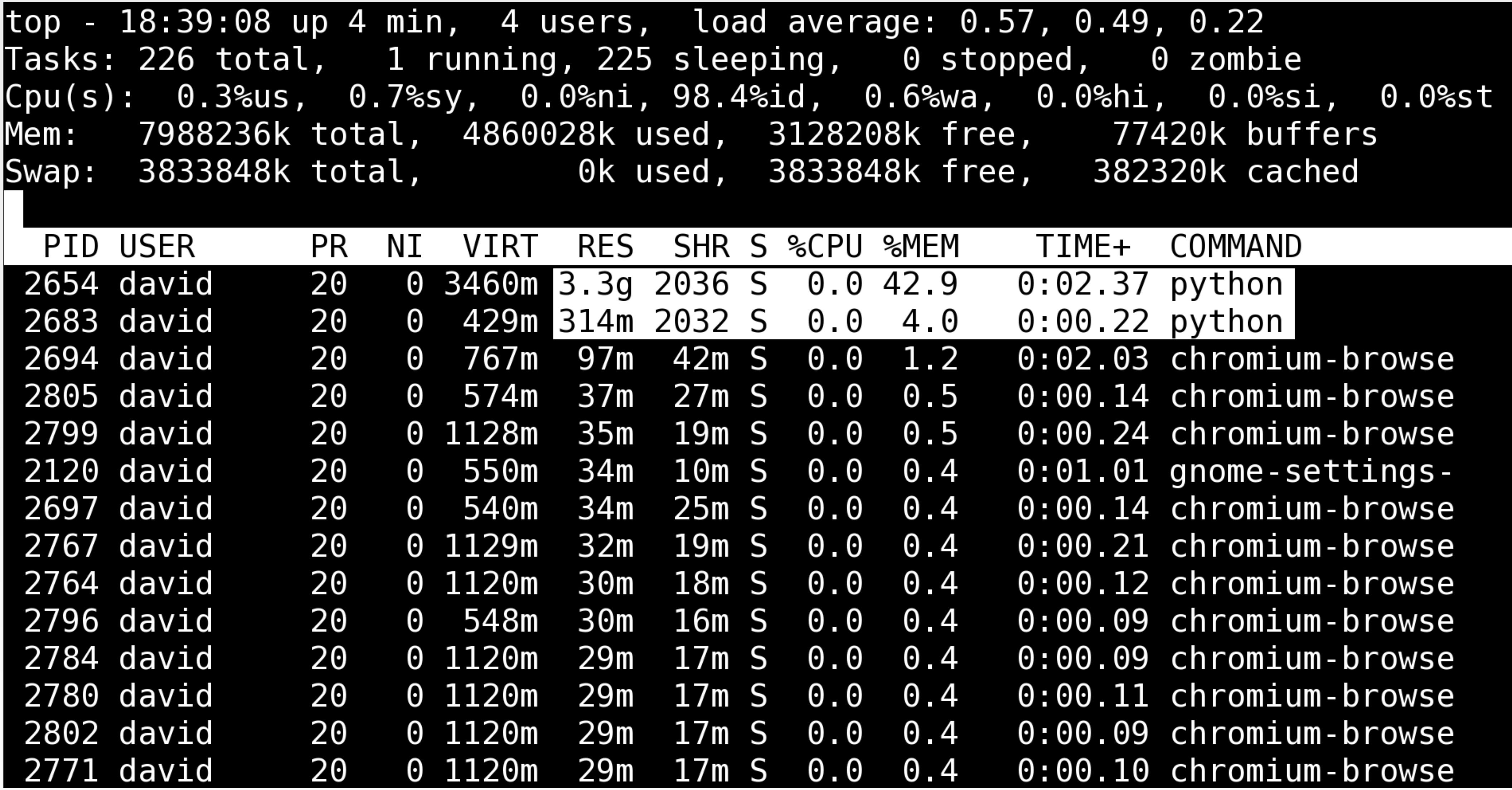
<dmalcolm@redhat.com>

PyCon US 2011



```
top - 18:39:08 up 4 min,  4 users,  load average: 0.57, 0.49, 0.22
Tasks: 226 total,   1 running, 225 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.7%sy,  0.0%ni, 98.4%id,  0.6%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   7988236k total,  4860028k used,  3128208k free,    77420k buffers
Swap:  3833848k total,        0k used,  3833848k free,   382320k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2654 david     20   0 3460m 3.3g 2036 S  0.0 42.9   0:02.37 python
 2683 david     20   0  429m 314m 2032 S  0.0  4.0   0:00.22 python
 2694 david     20   0  767m  97m  42m S  0.0  1.2   0:02.03 chromium-browse
 2805 david     20   0  574m  37m  27m S  0.0  0.5   0:00.14 chromium-browse
 2799 david     20   0 1128m  35m  19m S  0.0  0.5   0:00.24 chromium-browse
 2120 david     20   0  550m  34m  10m S  0.0  0.4   0:01.01 gnome-settings-
 2697 david     20   0  540m  34m  25m S  0.0  0.4   0:00.14 chromium-browse
 2767 david     20   0 1129m  32m  19m S  0.0  0.4   0:00.21 chromium-browse
 2764 david     20   0 1120m  30m  18m S  0.0  0.4   0:00.12 chromium-browse
 2796 david     20   0  548m  30m  16m S  0.0  0.4   0:00.09 chromium-browse
 2784 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.09 chromium-browse
 2780 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.11 chromium-browse
 2802 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.09 chromium-browse
 2771 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.10 chromium-browse
```



```
top - 18:39:08 up 4 min,  4 users,  load average: 0.57, 0.49, 0.22
Tasks: 226 total,   1 running, 225 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.3%us,  0.7%sy,  0.0%ni, 98.4%id,  0.6%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   7988236k total,  4860028k used,  3128208k free,    77420k buffers
Swap:  3833848k total,        0k used,  3833848k free,   382320k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2654 david     20   0 3460m 3.3g 2036 S  0.0 42.9   0:02.37 python
 2683 david     20   0  429m 314m 2032 S  0.0  4.0   0:00.22 python
 2694 david     20   0  767m  97m  42m S  0.0  1.2   0:02.03 chromium-browse
 2805 david     20   0  574m  37m  27m S  0.0  0.5   0:00.14 chromium-browse
 2799 david     20   0 1128m  35m  19m S  0.0  0.5   0:00.24 chromium-browse
 2120 david     20   0  550m  34m  10m S  0.0  0.4   0:01.01 gnome-settings-
 2697 david     20   0  540m  34m  25m S  0.0  0.4   0:00.14 chromium-browse
 2767 david     20   0 1129m  32m  19m S  0.0  0.4   0:00.21 chromium-browse
 2764 david     20   0 1120m  30m  18m S  0.0  0.4   0:00.12 chromium-browse
 2796 david     20   0  548m  30m  16m S  0.0  0.4   0:00.09 chromium-browse
 2784 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.09 chromium-browse
 2780 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.11 chromium-browse
 2802 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.09 chromium-browse
 2771 david     20   0 1120m  29m  17m S  0.0  0.4   0:00.10 chromium-browse
```

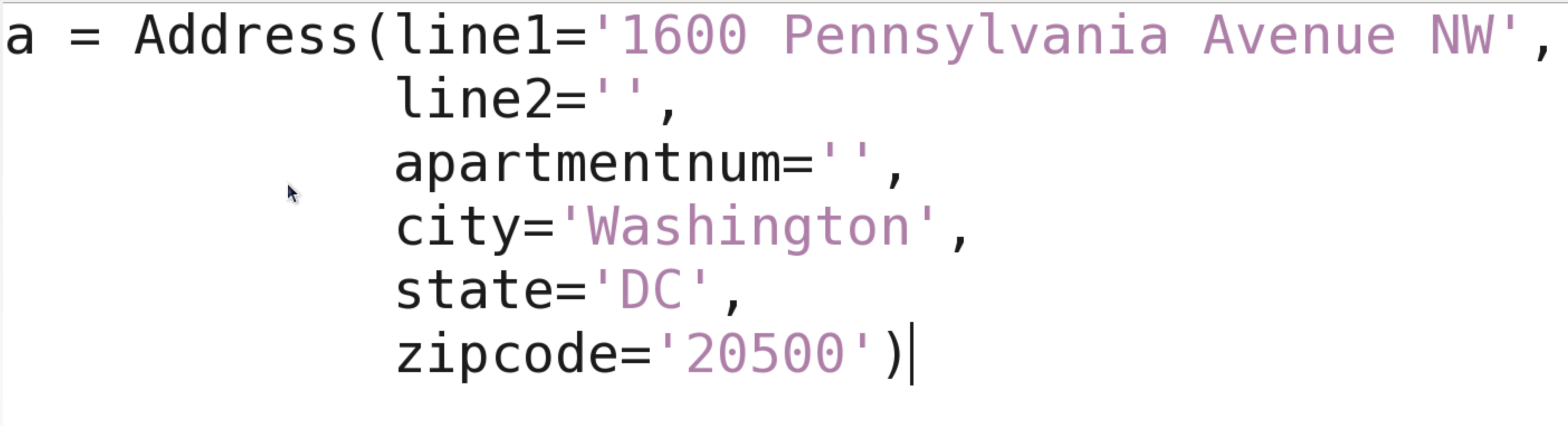
Overview

- The kinds of memory-usage problems
- Deep dive: from Python objects to transistors
  - Mostly just CPython at this stage
- Investigative tools
  - gdb-heap
- How big are different python objects?
- Fixing the problems
- Q & A

The kinds of memory-usage problems

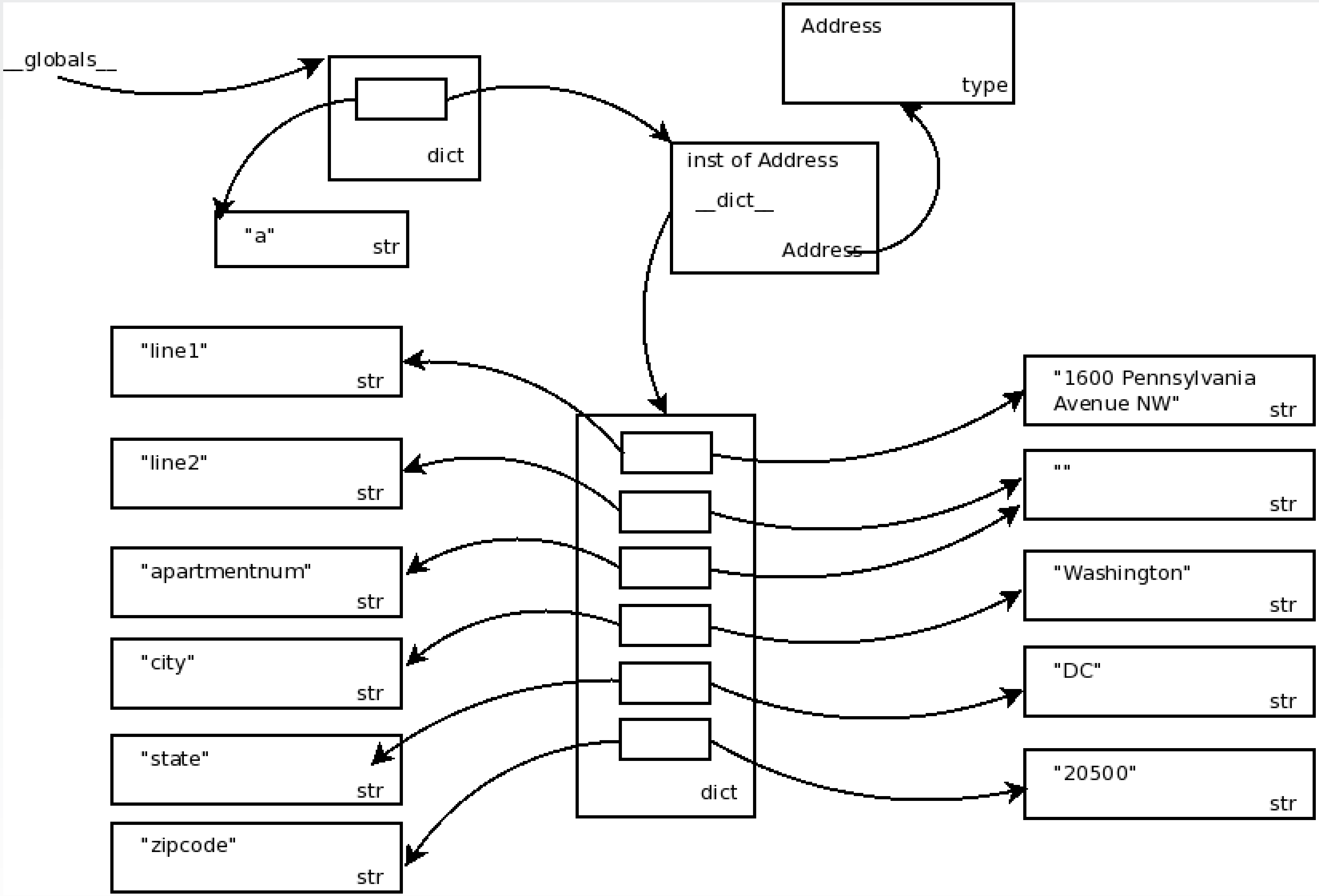
- A given workload uses more memory than you'd like
- Memory leaks: gradual increase in the memory usage of a process
- High-water-mark memory usage
- Heap fragmentation
- CPython: read-only access to objects generates writes to memory (ob_refcnt)

Python...



```python
a = Address(line1='1600 Pennsylvania Avenue NW',
            line2='',
            apartmentnum='',
            city='Washington',
            state='DC',
            zipcode='20500')
```

...as an object reference graph



...as allocated blocks of memory (CPython)

- e.g. a dictionary with more than 5 objects has an additional allocation of a table of (struct PyDictEntry) values
- TODO: diagram

...via per-type allocators (tp_new/tp_alloc) (CPython)

- Specialized allocators for int, float, etc
  - High-water-mark behavior (until 2.6/3.0)
- PyDict_New

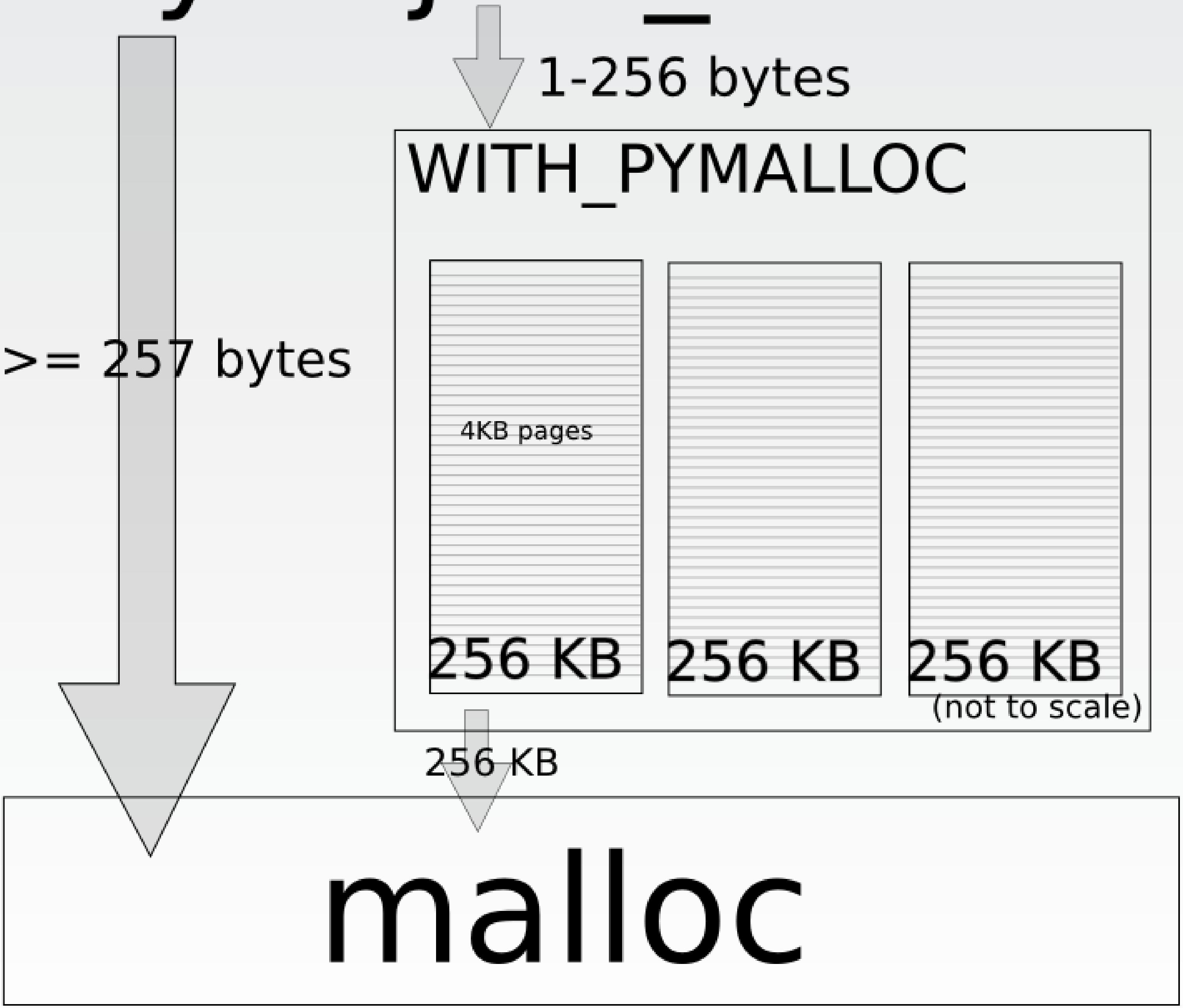
...via _PyObject_GC_Malloc (CPython)

- Reference-owning objects get a hidden PyGC_Head at the front:
    - 12 bytes on a 32-bit build of python
    - 32 bytes on a 64-bit build
- Allocation can trigger a garbage-collection
    - ...every 700 of such allocations, by default

Missing slide: reference counting (CPython)

Missing slide: the garbage collector (CPython)

PyObject_Malloc



1-256 bytes

>= 257 bytes

WITH_PYMALLOC

4KB pages

256 KB

256 KB

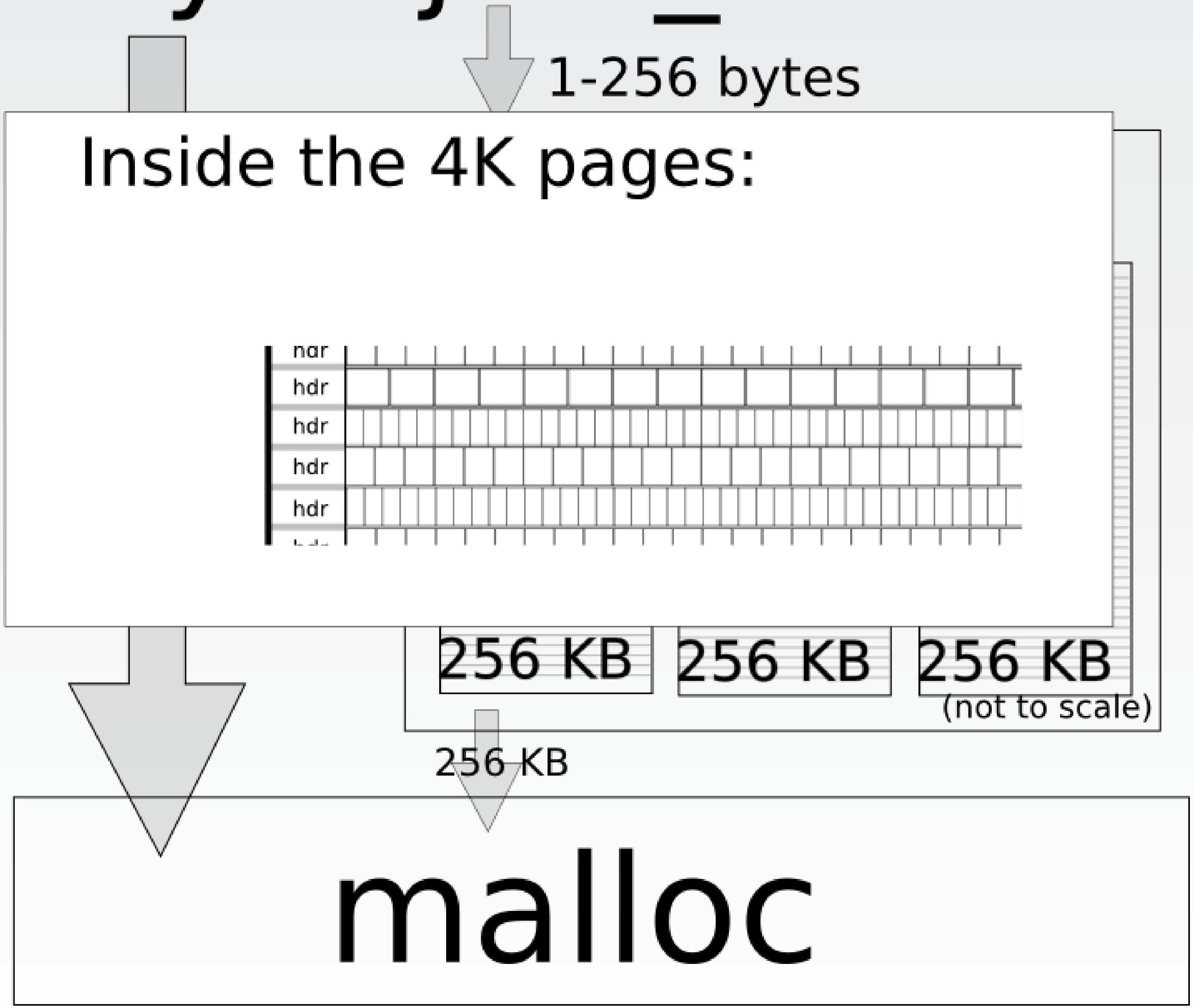
256 KB

(not to scale)

256 KB

malloc

PyObject_Malloc



1-256 bytes

Inside the 4K pages:

256 KB 256 KB 256 KB

(not to scale)

256 KB

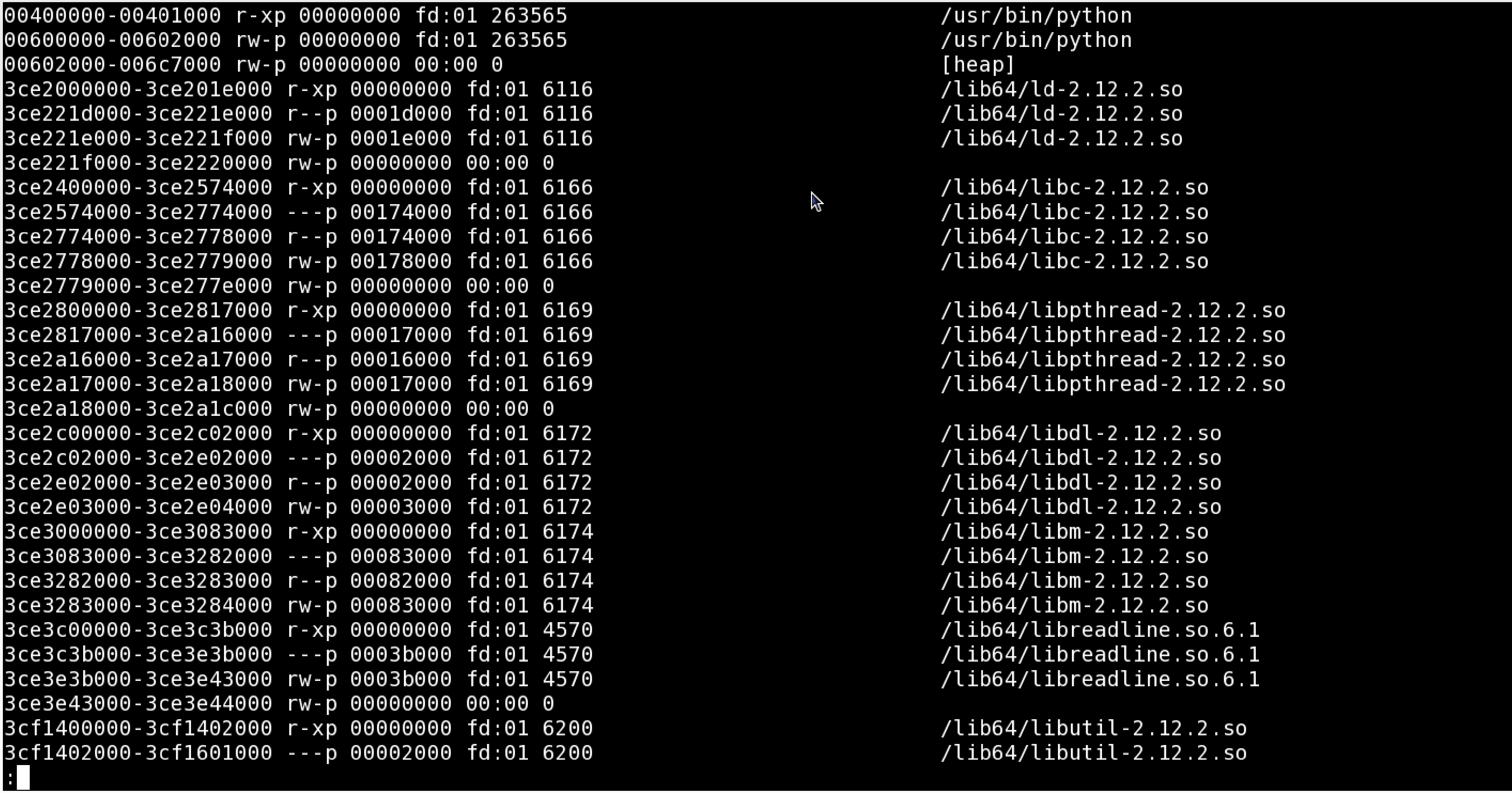
malloc

Differences in a debug build of CPython

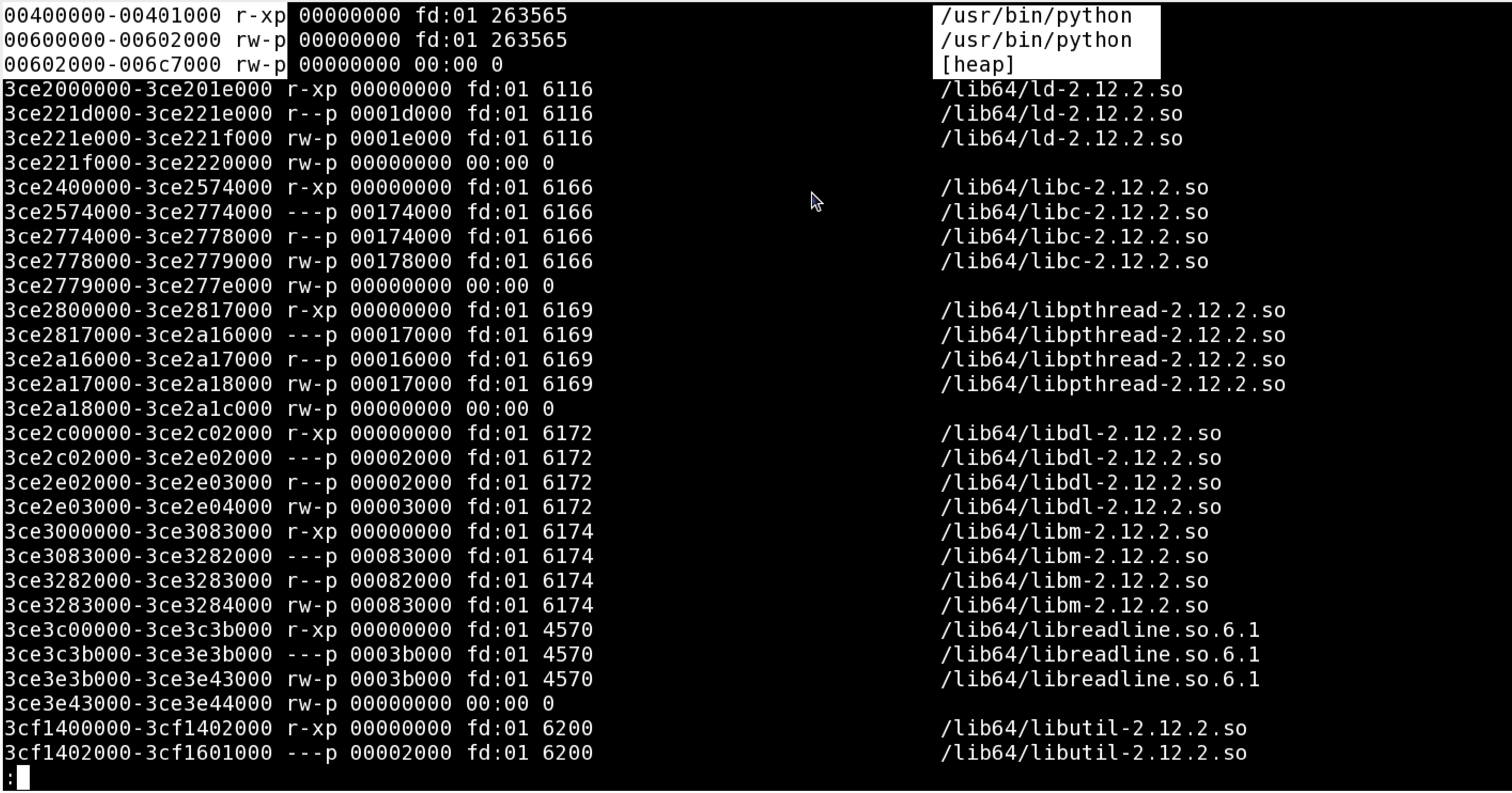
- Obmalloc arenas are disabled
- Extra padding around every allocation

Internals of “malloc”

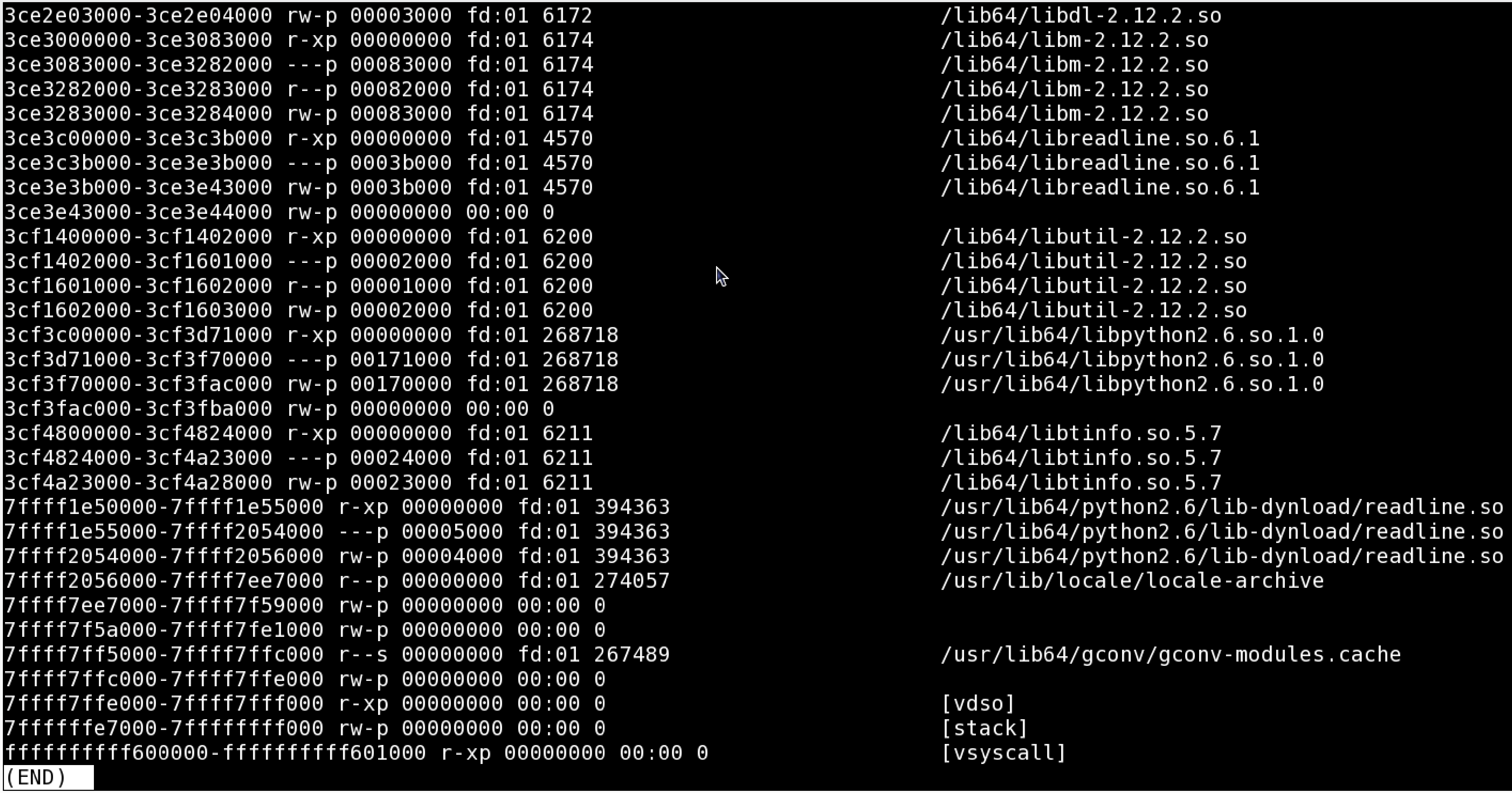
The “memory map” of a process...



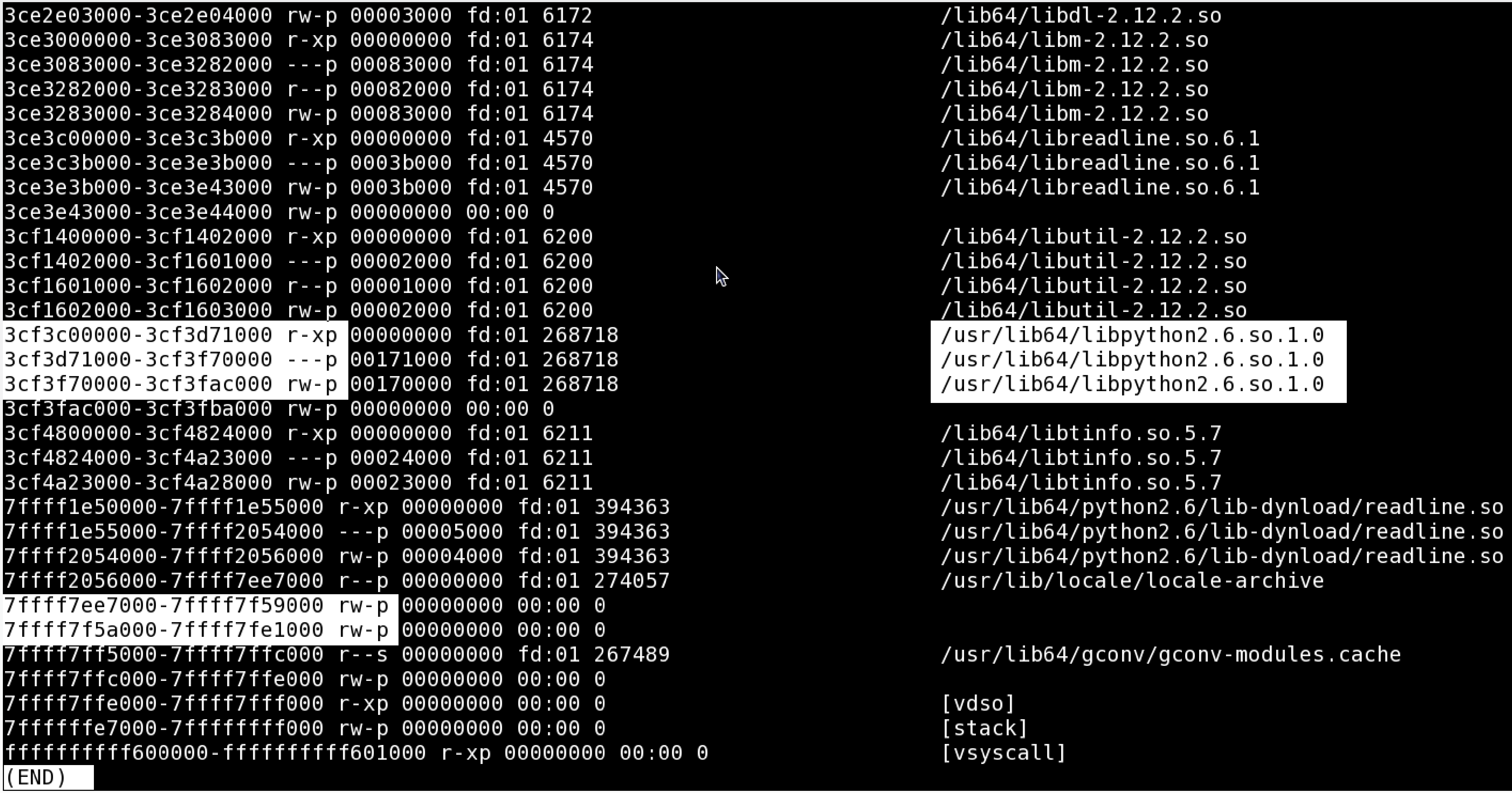
```
00400000-00401000 r-xp 00000000 fd:01 263565                             /usr/bin/python
00600000-00602000 rw-p 00000000 fd:01 263565                             /usr/bin/python
00602000-006c7000 rw-p 00000000 00:00 0                                  [heap]
3ce2000000-3ce201e000 r-xp 00000000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221d000-3ce221e000 r--p 0001d000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221e000-3ce221f000 rw-p 0001e000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221f000-3ce2220000 rw-p 00000000 00:00 0
3ce2400000-3ce2574000 r-xp 00000000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2574000-3ce2774000 ---p 00174000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2774000-3ce2778000 r--p 00174000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2778000-3ce2779000 rw-p 00178000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2779000-3ce277e000 rw-p 00000000 00:00 0
3ce2800000-3ce2817000 r-xp 00000000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2817000-3ce2a16000 ---p 00017000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a16000-3ce2a17000 r--p 00016000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a17000-3ce2a18000 rw-p 00017000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a18000-3ce2a1c000 rw-p 00000000 00:00 0
3ce2c00000-3ce2c02000 r-xp 00000000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2c02000-3ce2e02000 ---p 00002000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2e02000-3ce2e03000 r--p 00002000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2e03000-3ce2e04000 rw-p 00003000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce3000000-3ce3083000 r-xp 00000000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3083000-3ce3282000 ---p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3282000-3ce3283000 r--p 00082000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3283000-3ce3284000 rw-p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3c00000-3ce3c3b000 r-xp 00000000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3c3b000-3ce3e3b000 ---p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e3b000-3ce3e43000 rw-p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e43000-3ce3e44000 rw-p 00000000 00:00 0
3cf1400000-3cf1402000 r-xp 00000000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1402000-3cf1601000 ---p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
:
```



```
00400000-00401000 r-xp 00000000 fd:01 263565                             /usr/bin/python
00600000-00602000 rw-p 00000000 fd:01 263565                             /usr/bin/python
00602000-006c7000 rw-p 00000000 00:00 0                                  [heap]
3ce2000000-3ce201e000 r-xp 00000000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221d000-3ce221e000 r--p 0001d000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221e000-3ce221f000 rw-p 0001e000 fd:01 6116                           /lib64/ld-2.12.2.so
3ce221f000-3ce2220000 rw-p 00000000 00:00 0
3ce2400000-3ce2574000 r-xp 00000000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2574000-3ce2774000 ---p 00174000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2774000-3ce2778000 r--p 00174000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2778000-3ce2779000 rw-p 00178000 fd:01 6166                           /lib64/libc-2.12.2.so
3ce2779000-3ce277e000 rw-p 00000000 00:00 0
3ce2800000-3ce2817000 r-xp 00000000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2817000-3ce2a16000 ---p 00017000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a16000-3ce2a17000 r--p 00016000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a17000-3ce2a18000 rw-p 00017000 fd:01 6169                           /lib64/libpthread-2.12.2.so
3ce2a18000-3ce2a1c000 rw-p 00000000 00:00 0
3ce2c00000-3ce2c02000 r-xp 00000000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2c02000-3ce2e02000 ---p 00002000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2e02000-3ce2e03000 r--p 00002000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce2e03000-3ce2e04000 rw-p 00003000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce3000000-3ce3083000 r-xp 00000000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3083000-3ce3282000 ---p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3282000-3ce3283000 r--p 00082000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3283000-3ce3284000 rw-p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3c00000-3ce3c3b000 r-xp 00000000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3c3b000-3ce3e3b000 ---p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e3b000-3ce3e43000 rw-p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e43000-3ce3e44000 rw-p 00000000 00:00 0
3cf1400000-3cf1402000 r-xp 00000000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1402000-3cf1601000 ---p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
:
```



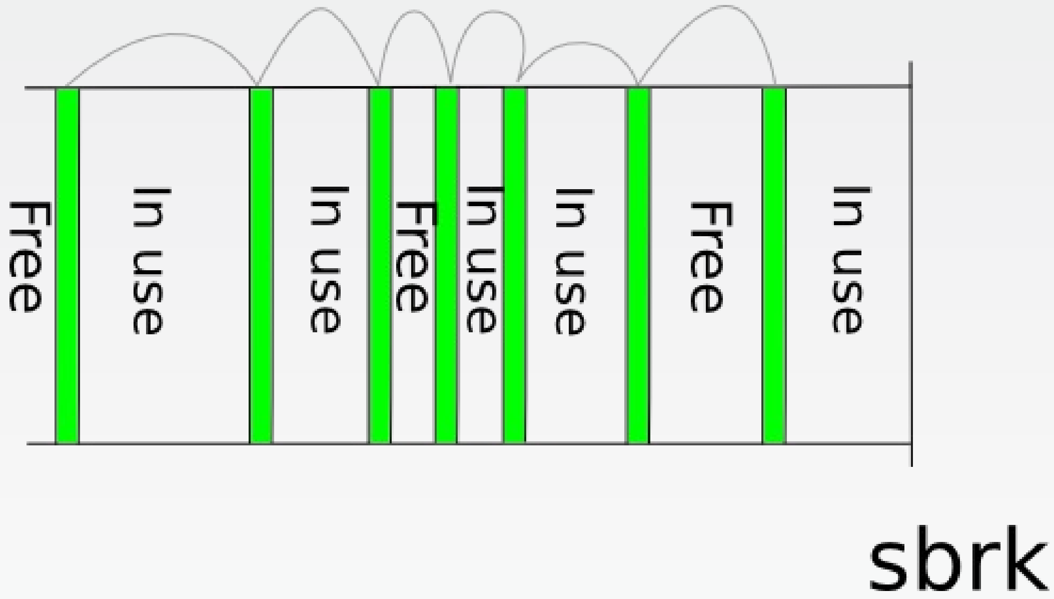
```
3ce2e03000-3ce2e04000 rw-p 00003000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce3000000-3ce3083000 r-xp 00000000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3083000-3ce3282000 ---p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3282000-3ce3283000 r--p 00082000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3283000-3ce3284000 rw-p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3c00000-3ce3c3b000 r-xp 00000000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3c3b000-3ce3e3b000 ---p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e3b000-3ce3e43000 rw-p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e43000-3ce3e44000 rw-p 00000000 00:00 0
3cf1400000-3cf1402000 r-xp 00000000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1402000-3cf1601000 ---p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1601000-3cf1602000 r--p 00001000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1602000-3cf1603000 rw-p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf3c00000-3cf3d71000 r-xp 00000000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3d71000-3cf3f70000 ---p 00171000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3f70000-3cf3fac000 rw-p 00170000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3fac000-3cf3fba000 rw-p 00000000 00:00 0
3cf4800000-3cf4824000 r-xp 00000000 fd:01 6211                           /lib64/libtinfo.so.5.7
3cf4824000-3cf4a23000 ---p 00024000 fd:01 6211                           /lib64/libtinfo.so.5.7
3cf4a23000-3cf4a28000 rw-p 00023000 fd:01 6211                           /lib64/libtinfo.so.5.7
7ffff1e50000-7ffff1e55000 r-xp 00000000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff1e55000-7ffff2054000 ---p 00005000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff2054000-7ffff2056000 rw-p 00004000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff2056000-7ffff7ee7000 r--p 00000000 fd:01 274057                     /usr/lib/locale/locale-archive
7ffff7ee7000-7ffff7f59000 rw-p 00000000 00:00 0
7ffff7f5a000-7ffff7fe1000 rw-p 00000000 00:00 0
7ffff7ff5000-7ffff7ffc000 r--s 00000000 fd:01 267489                     /usr/lib64/gconv/gconv-modules.cache
7ffff7ffc000-7ffff7ffe000 rw-p 00000000 00:00 0
7ffff7ffe000-7ffff7fff000 r-xp 00000000 00:00 0                          [vdso]
7ffffffe7000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
(END)
```



```
3ce2e03000-3ce2e04000 rw-p 00003000 fd:01 6172                           /lib64/libdl-2.12.2.so
3ce3000000-3ce3083000 r-xp 00000000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3083000-3ce3282000 ---p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3282000-3ce3283000 r--p 00082000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3283000-3ce3284000 rw-p 00083000 fd:01 6174                           /lib64/libm-2.12.2.so
3ce3c00000-3ce3c3b000 r-xp 00000000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3c3b000-3ce3e3b000 ---p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e3b000-3ce3e43000 rw-p 0003b000 fd:01 4570                           /lib64/libreadline.so.6.1
3ce3e43000-3ce3e44000 rw-p 00000000 00:00 0
3cf1400000-3cf1402000 r-xp 00000000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1402000-3cf1601000 ---p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1601000-3cf1602000 r--p 00001000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf1602000-3cf1603000 rw-p 00002000 fd:01 6200                           /lib64/libutil-2.12.2.so
3cf3c00000-3cf3d71000 r-xp 00000000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3d71000-3cf3f70000 ---p 00171000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3f70000-3cf3fac000 rw-p 00170000 fd:01 268718                         /usr/lib64/libpython2.6.so.1.0
3cf3fac000-3cf3fba000 rw-p 00000000 00:00 0
3cf4800000-3cf4824000 r-xp 00000000 fd:01 6211                           /lib64/libtinfo.so.5.7
3cf4824000-3cf4a23000 ---p 00024000 fd:01 6211                           /lib64/libtinfo.so.5.7
3cf4a23000-3cf4a28000 rw-p 00023000 fd:01 6211                           /lib64/libtinfo.so.5.7
7ffff1e50000-7ffff1e55000 r-xp 00000000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff1e55000-7ffff2054000 ---p 00005000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff2054000-7ffff2056000 rw-p 00004000 fd:01 394363                     /usr/lib64/python2.6/lib-dynload/readline.so
7ffff2056000-7ffff7ee7000 r--p 00000000 fd:01 274057                     /usr/lib/locale/locale-archive
7ffff7ee7000-7ffff7f59000 rw-p 00000000 00:00 0
7ffff7f5a000-7ffff7fe1000 rw-p 00000000 00:00 0
7ffff7ff5000-7ffff7ffc000 r--s 00000000 fd:01 267489                     /usr/lib64/gconv/gconv-modules.cache
7ffff7ffc000-7ffff7ffe000 rw-p 00000000 00:00 0
7ffff7ffe000-7ffff7fff000 r-xp 00000000 00:00 0                          [vdso]
7ffffffe7000-7ffffffff000 rw-p 00000000 00:00 0                          [stack]
ffffffffff600000-ffffffffff601000 r-xp 00000000 00:00 0                  [vsyscall]
(END)
```

Malloc in GNU libc: boundary tags

- sbrk and anonymous mmap regions:



Virtual memory

- All of the above are divided into 4K pages
- Not all of them are necessarily used by physical memory
- Some of them may be shared
  - Not as many as you'd like (ob_refcnt)
- KVM, KSM, and ob_refcnt

Investigative tools

- sys.getsizeof()
- Debug build
- Meliae
- Heapy
- valgrind
- gdb-heap

sys.getsizeof(obj)

- Added in Python 2.6
- Calls __sizeof__ on the object
  - (apart from instances of old-style classes)
- Adds the GC header size if necessary
- All the builtin types implement __sizeof__
  - e.g. for PyDictObject, it adds in the size of the PyDictEntry table, if it's external
- 3rd party extension modules might get this wrong

Object sizes (optimized CPython)

Type	32-bit	64-bit	
int (python2) (Single instances for range [-5, 257); custom allocator)	12	24	
long (python2) / **int** (python3) (Single instances for range [-5, 257) on py3k)	14 + (2 * num “digits”)	30 + (2 * num “digits”)	
str (python2) / **bytes** (python3) (Single instances for empty string and all length-1 values)	24 + length	40 + length	
unicode (python2) / **str** (python3) (Single instances for empty string and all length-1 values for Latin-1)	28 + (2 or 4 * length)	52 + (2 or 4 * length)	
list	32 + (4 * length)	72 + (8 * length)	
tuple (Shared instance of empty tuple; custom allocator)	24 + (4 * length)	64 + (8 * length)	
float (Custom allocator)	16	24	

Dictionary sizes (optimized CPython)

	32-bit	64-bit
Base size	136 + (12 * per external PyDictEntry tables)	280 + (24 per external PyDictEntry table)
0-5 entries	136	280
6-21 entries	520	1048
22-85 entries	1672	3352

Debug build of Python

- Adds: _ob_next, _ob_prev pointers to the top of every object
- Doubly-linked list of all objects that think they're alive
- Adds sys.gettotalrefcount()
  - If this continually increases you may have a leak
    - Or a cache...
- Set “PYTHONDUMPREFS=1” in the environment to see all objects still alive on exit

Meliae

https://launchpad.net/meliae

- Python extension module for taking a snapshot of the object reference graph, as seen by the GC.
  - Everything, via gc.get_objects()
  - Everything “downstream” of a given object
- Dumps object reference graph in JSON form
    - address, type, __name__, len, value, refs
- Tools for analyzing results

Heapy

- http://guppy-pe.sourceforge.net/#Heapy

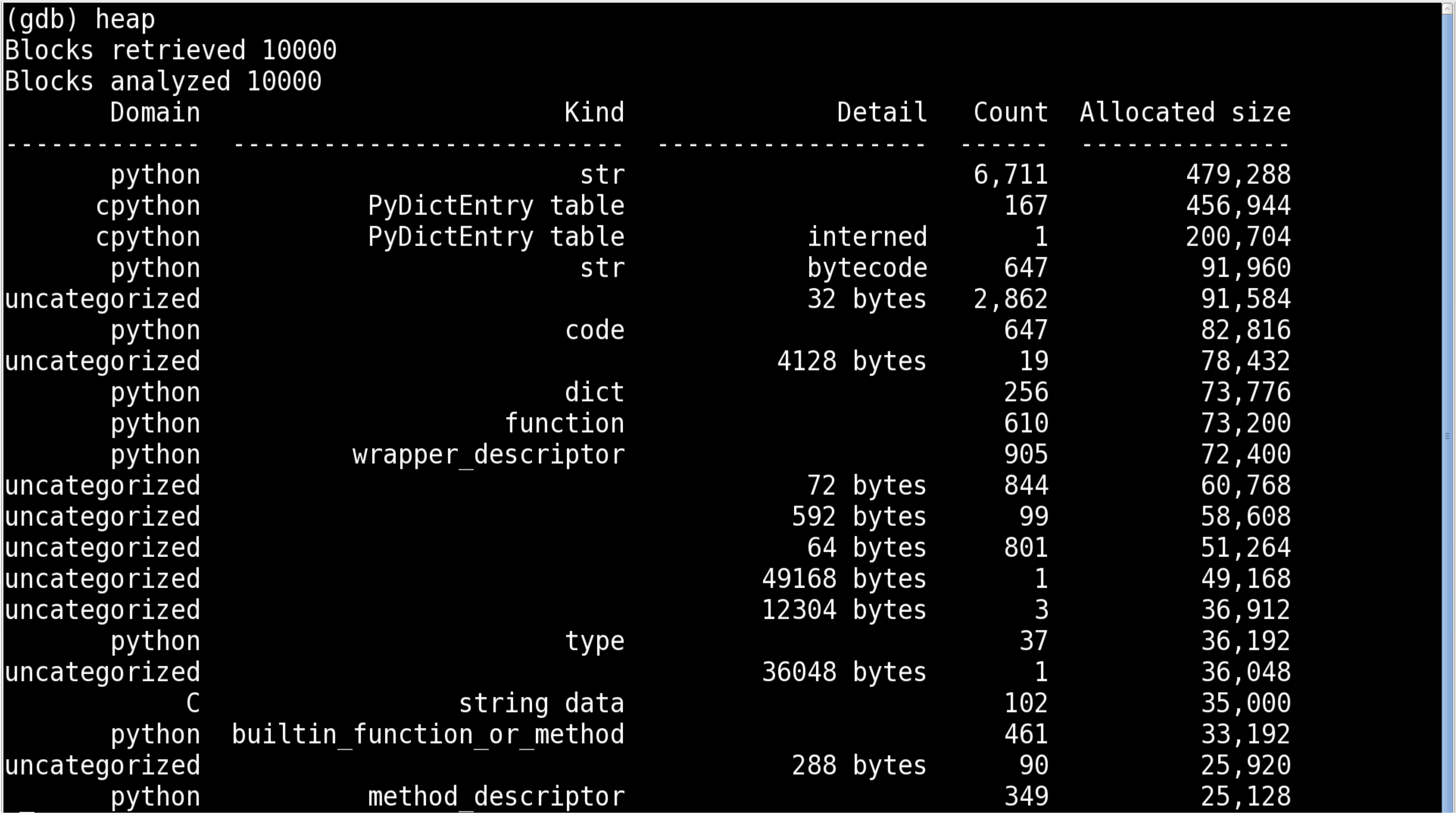
Valgrind (Linux)

- Low-level memory-leak finder
- Instrumented malloc/free
- As of CPython 2.7, 3.2, CPython can detect if it's being run under valgrind and automatically turn off the arena allocator

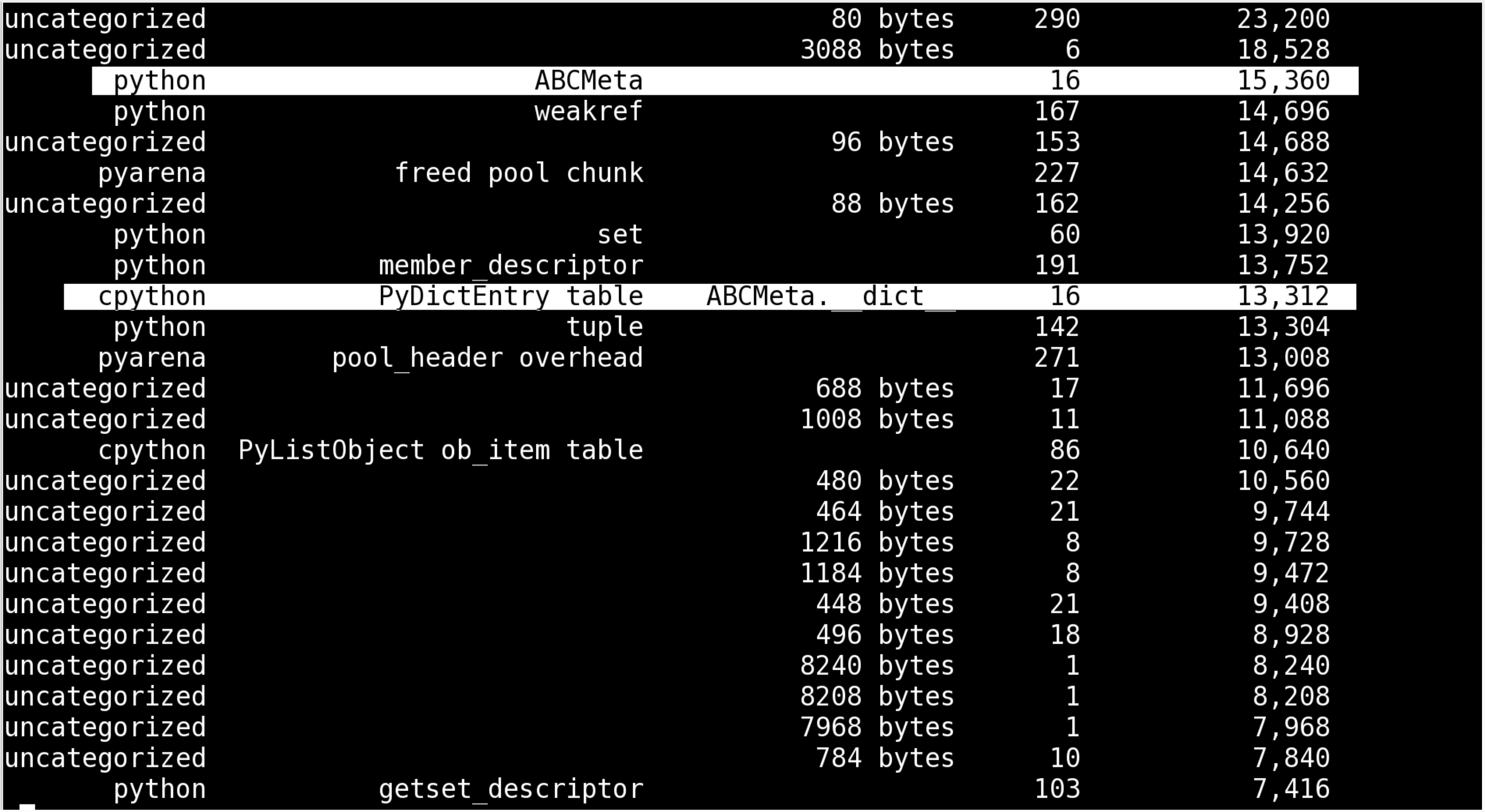
gdb-heap (GNU)

- Extension to the GNU debugger (v7 onwards)
  - GNU libc's malloc/free implementation
  - Python's allocators and objects
    - Other stuff: C++, GTK's Gtype
- Very low level
      - Too low-level?
- Written in Python itself
- Doesn't require co-operation of the process under investigation
  - Usable on coredumps

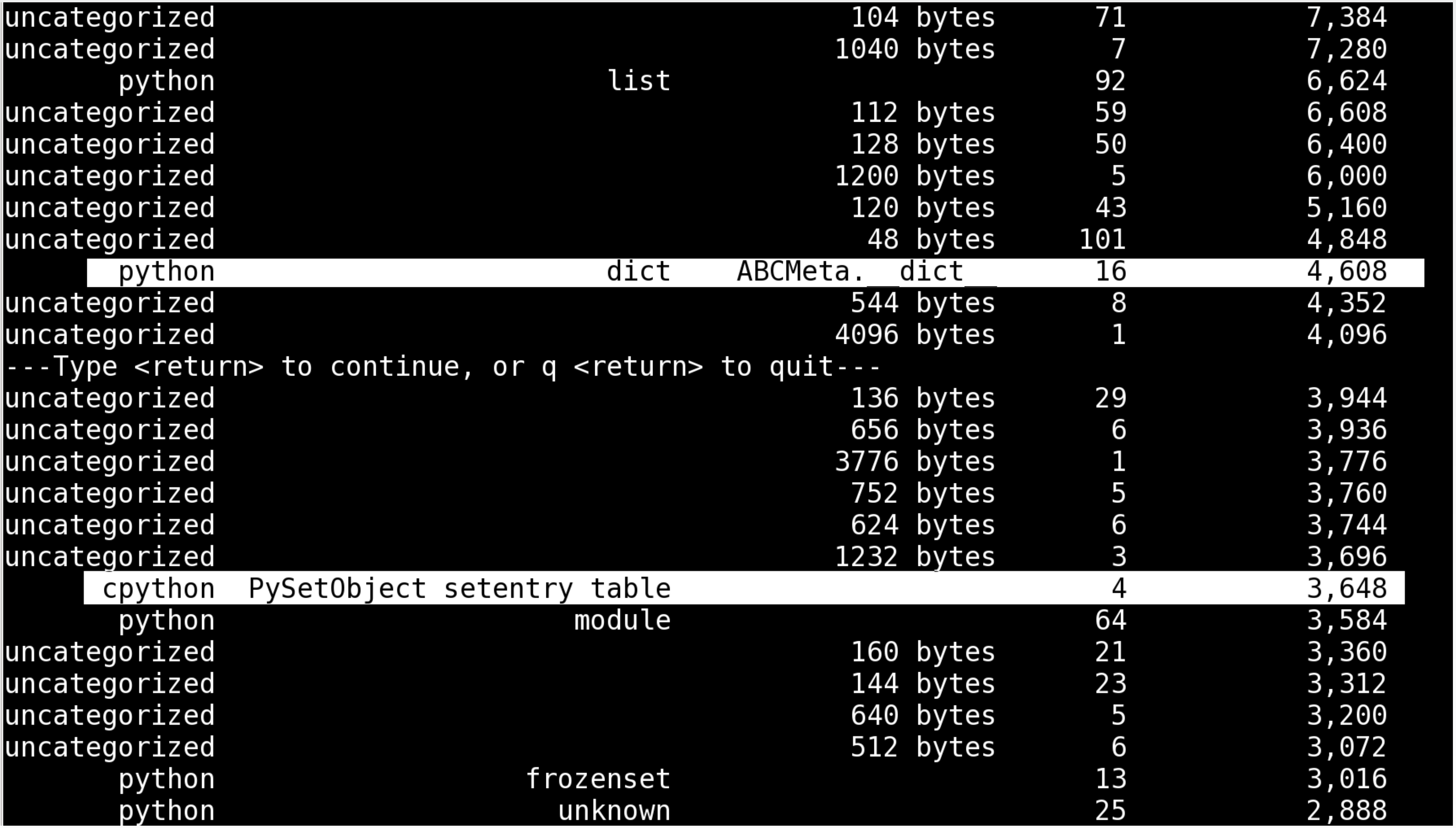
Demo



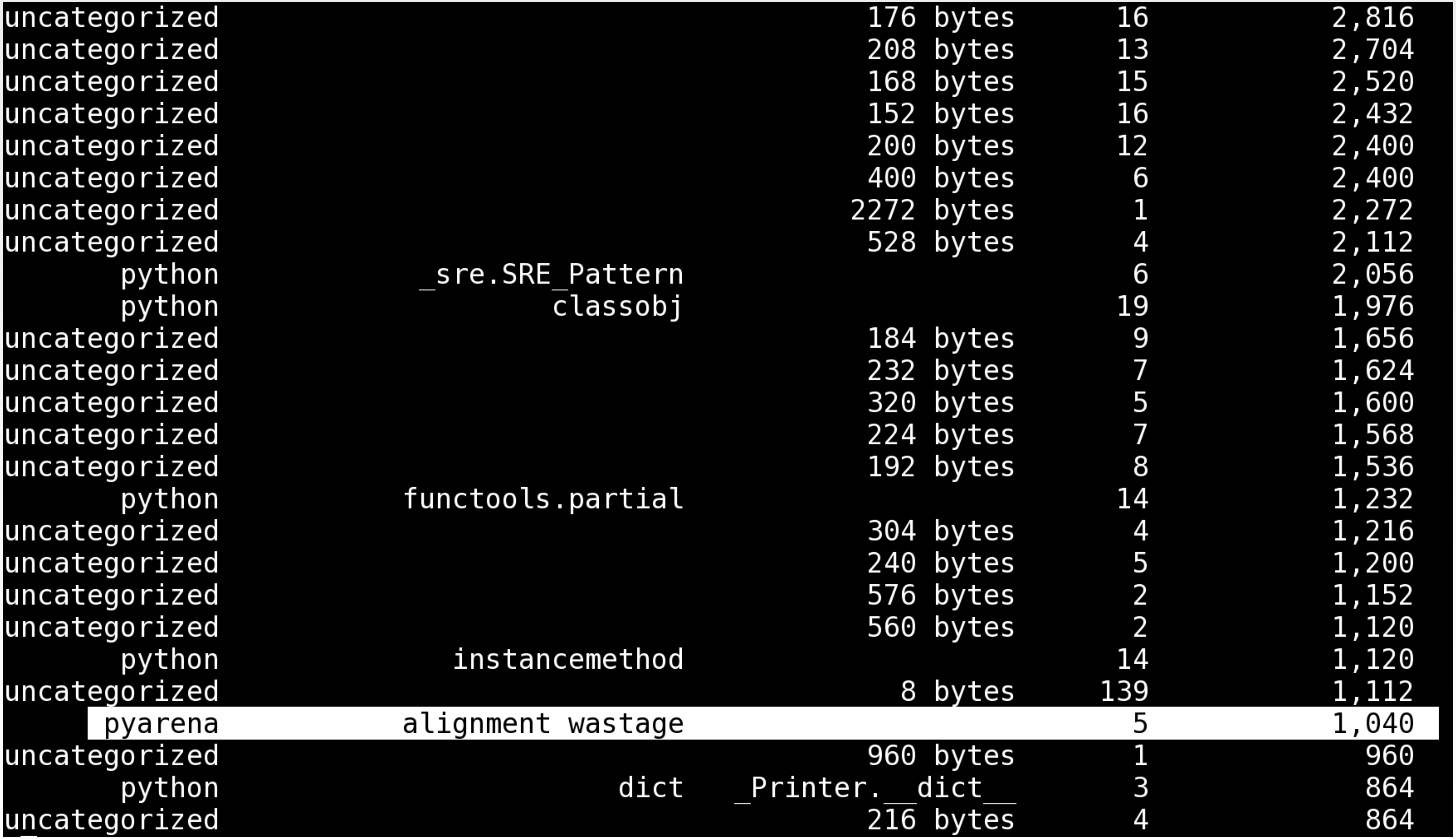
```
(gdb) heap
Blocks retrieved 10000
Blocks analyzed 10000
       Domain                        Kind              Detail   Count  Allocated size
-------------  --------------------------  ------------------  ------  --------------
       python                         str                       6,711         479,288
      cpython           PyDictEntry table                         167         456,944
      cpython           PyDictEntry table            interned       1         200,704
       python                         str            bytecode     647          91,960
uncategorized                                        32 bytes   2,862          91,584
       python                        code                         647          82,816
uncategorized                                      4128 bytes      19          78,432
       python                        dict                         256          73,776
       python                    function                         610          73,200
       python          wrapper_descriptor                         905          72,400
uncategorized                                        72 bytes     844          60,768
uncategorized                                       592 bytes      99          58,608
uncategorized                                        64 bytes     801          51,264
uncategorized                                     49168 bytes       1          49,168
uncategorized                                     12304 bytes       3          36,912
       python                        type                          37          36,192
uncategorized                                     36048 bytes       1          36,048
            C                 string data                         102          35,000
       python  builtin_function_or_method                         461          33,192
uncategorized                                       288 bytes      90          25,920
       python           method_descriptor                         349          25,128
```



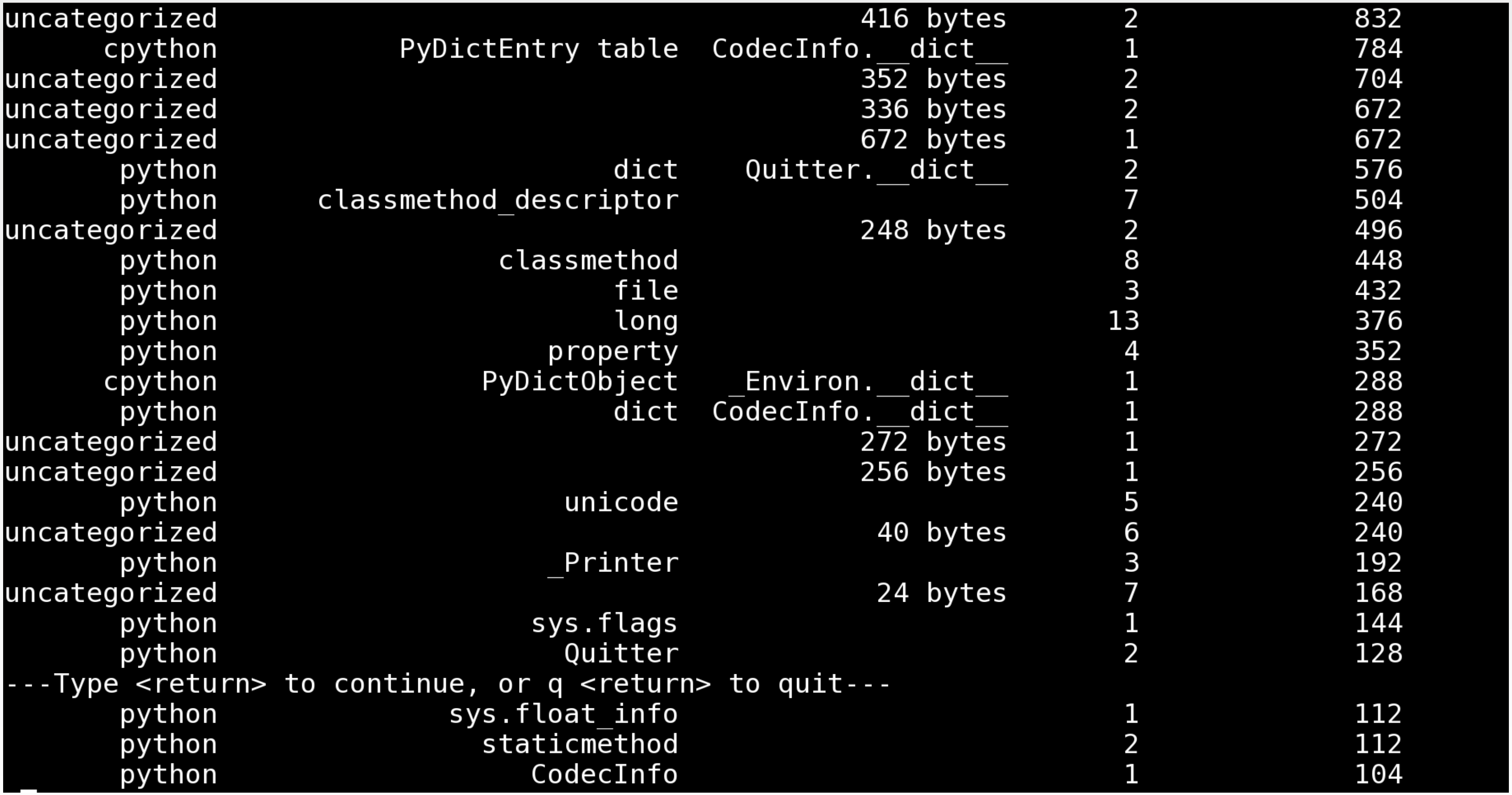
```
uncategorized                                        80 bytes     290          23,200
uncategorized                                      3088 bytes       6          18,528
       python                     ABCMeta                          16          15,360
       python                     weakref                         167          14,696
uncategorized                                        96 bytes     153          14,688
      pyarena            freed pool chunk                         227          14,632
uncategorized                                        88 bytes     162          14,256
       python                         set                          60          13,920
       python           member_descriptor                         191          13,752
      cpython           PyDictEntry table    ABCMeta.__dict__      16          13,312
       python                       tuple                         142          13,304
      pyarena        pool_header overhead                         271          13,008
uncategorized                                       688 bytes      17          11,696
uncategorized                                      1008 bytes      11          11,088
      cpython  PyListObject ob_item table                          86          10,640
uncategorized                                       480 bytes      22          10,560
uncategorized                                       464 bytes      21           9,744
uncategorized                                      1216 bytes       8           9,728
uncategorized                                      1184 bytes       8           9,472
uncategorized                                       448 bytes      21           9,408
uncategorized                                       496 bytes      18           8,928
uncategorized                                      8240 bytes       1           8,240
uncategorized                                      8208 bytes       1           8,208
uncategorized                                      7968 bytes       1           7,968
uncategorized                                       784 bytes      10           7,840
       python           getset_descriptor                         103           7,416
```



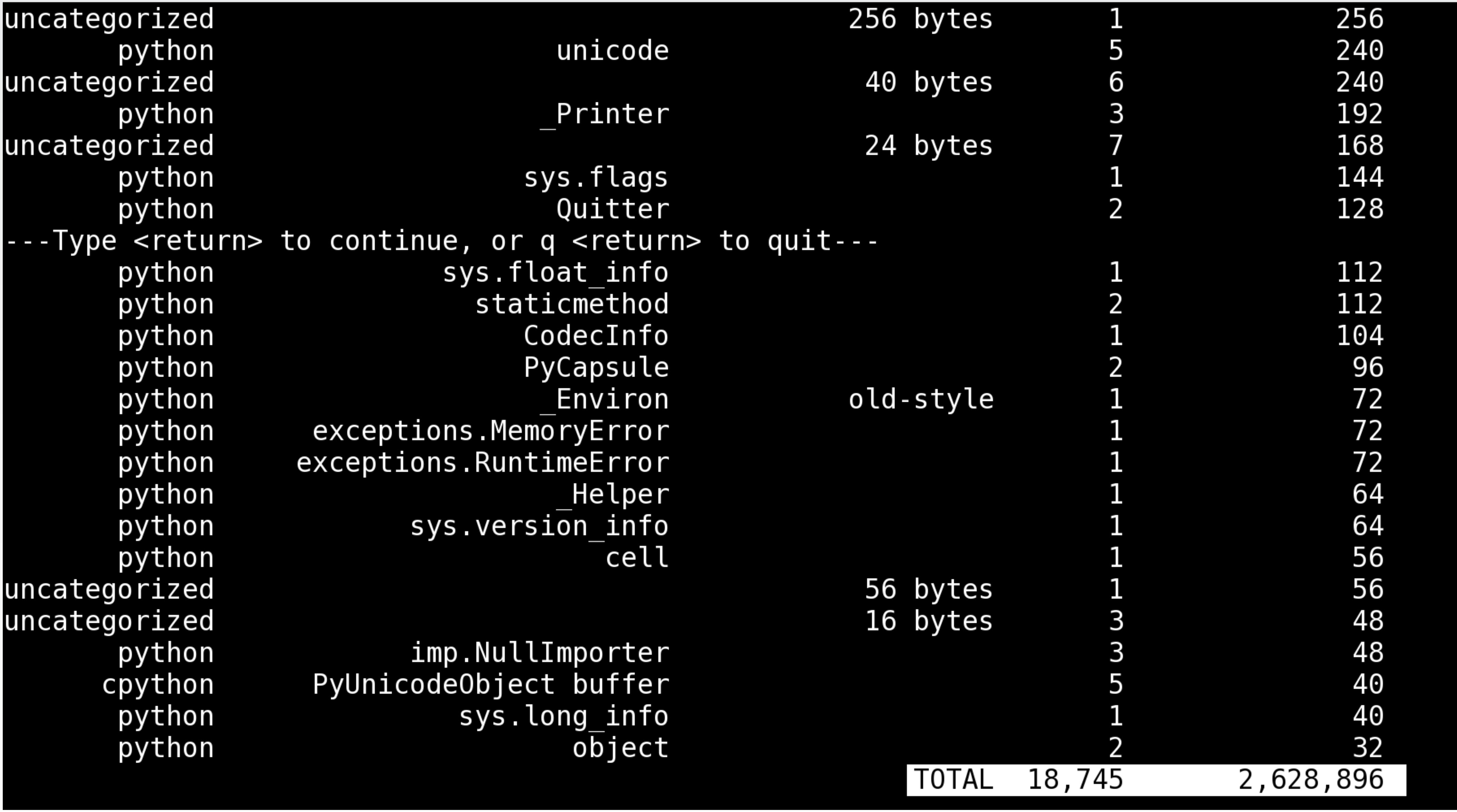
```
uncategorized                                       104 bytes      71            7,384
uncategorized                                      1040 bytes       7            7,280
       python                        list                          92            6,624
uncategorized                                       112 bytes      59            6,608
uncategorized                                       128 bytes      50            6,400
uncategorized                                      1200 bytes       5            6,000
uncategorized                                       120 bytes      43            5,160
uncategorized                                        48 bytes     101            4,848
       python                        dict    ABCMeta.__dict__      16            4,608
uncategorized                                       544 bytes       8            4,352
uncategorized                                      4096 bytes       1            4,096
---Type <return> to continue, or q <return> to quit---
uncategorized                                       136 bytes      29            3,944
uncategorized                                       656 bytes       6            3,936
uncategorized                                      3776 bytes       1            3,776
uncategorized                                       752 bytes       5            3,760
uncategorized                                       624 bytes       6            3,744
uncategorized                                      1232 bytes       3            3,696
      cpython  PySetObject setentry table                           4            3,648
       python                      module                          64            3,584
uncategorized                                       160 bytes      21            3,360
uncategorized                                       144 bytes      23            3,312
uncategorized                                       640 bytes       5            3,200
uncategorized                                       512 bytes       6            3,072
       python                   frozenset                          13            3,016
       python                     unknown                          25            2,888
```



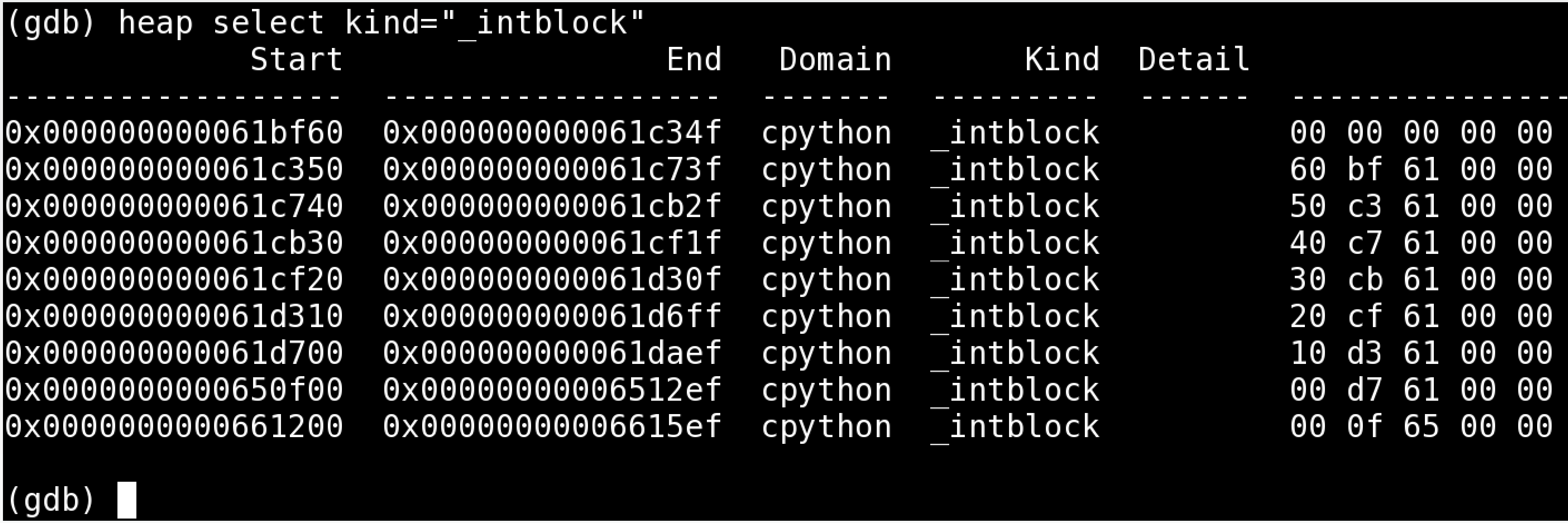
```
uncategorized                                       176 bytes      16          2,816
uncategorized                                       208 bytes      13          2,704
uncategorized                                       168 bytes      15          2,520
uncategorized                                       152 bytes      16          2,432
uncategorized                                       200 bytes      12          2,400
uncategorized                                       400 bytes       6          2,400
uncategorized                                      2272 bytes       1          2,272
uncategorized                                       528 bytes       4          2,112
       python            _sre.SRE_Pattern                           6          2,056
       python                    classobj                          19          1,976
uncategorized                                       184 bytes       9          1,656
uncategorized                                       232 bytes       7          1,624
uncategorized                                       320 bytes       5          1,600
uncategorized                                       224 bytes       7          1,568
uncategorized                                       192 bytes       8          1,536
       python           functools.partial                          14          1,232
uncategorized                                       304 bytes       4          1,216
uncategorized                                       240 bytes       5          1,200
uncategorized                                       576 bytes       2          1,152
uncategorized                                       560 bytes       2          1,120
       python              instancemethod                          14          1,120
uncategorized                                         8 bytes     139          1,112
      pyarena           alignment wastage                           5          1,040
uncategorized                                       960 bytes       1            960
       python                        dict   _Printer.__dict__       3            864
uncategorized                                       216 bytes       4            864
```



```
uncategorized                                       416 bytes       2             832
      cpython           PyDictEntry table  CodecInfo.__dict__       1             784
uncategorized                                       352 bytes       2             704
uncategorized                                       336 bytes       2             672
uncategorized                                       672 bytes       1             672
       python                        dict    Quitter.__dict__       2             576
       python      classmethod_descriptor                           7             504
uncategorized                                       248 bytes       2             496
       python                 classmethod                           8             448
       python                        file                           3             432
       python                        long                          13             376
       python                    property                           4             352
      cpython                PyDictObject   _Environ.__dict__       1             288
       python                        dict  CodecInfo.__dict__       1             288
uncategorized                                       272 bytes       1             272
uncategorized                                       256 bytes       1             256
       python                     unicode                           5             240
uncategorized                                        40 bytes       6             240
       python                    _Printer                           3             192
uncategorized                                        24 bytes       7             168
       python                   sys.flags                           1             144
       python                     Quitter                           2             128
---Type <return> to continue, or q <return> to quit---
       python              sys.float_info                           1             112
       python                staticmethod                           2             112
       python                   CodecInfo                           1             104
```

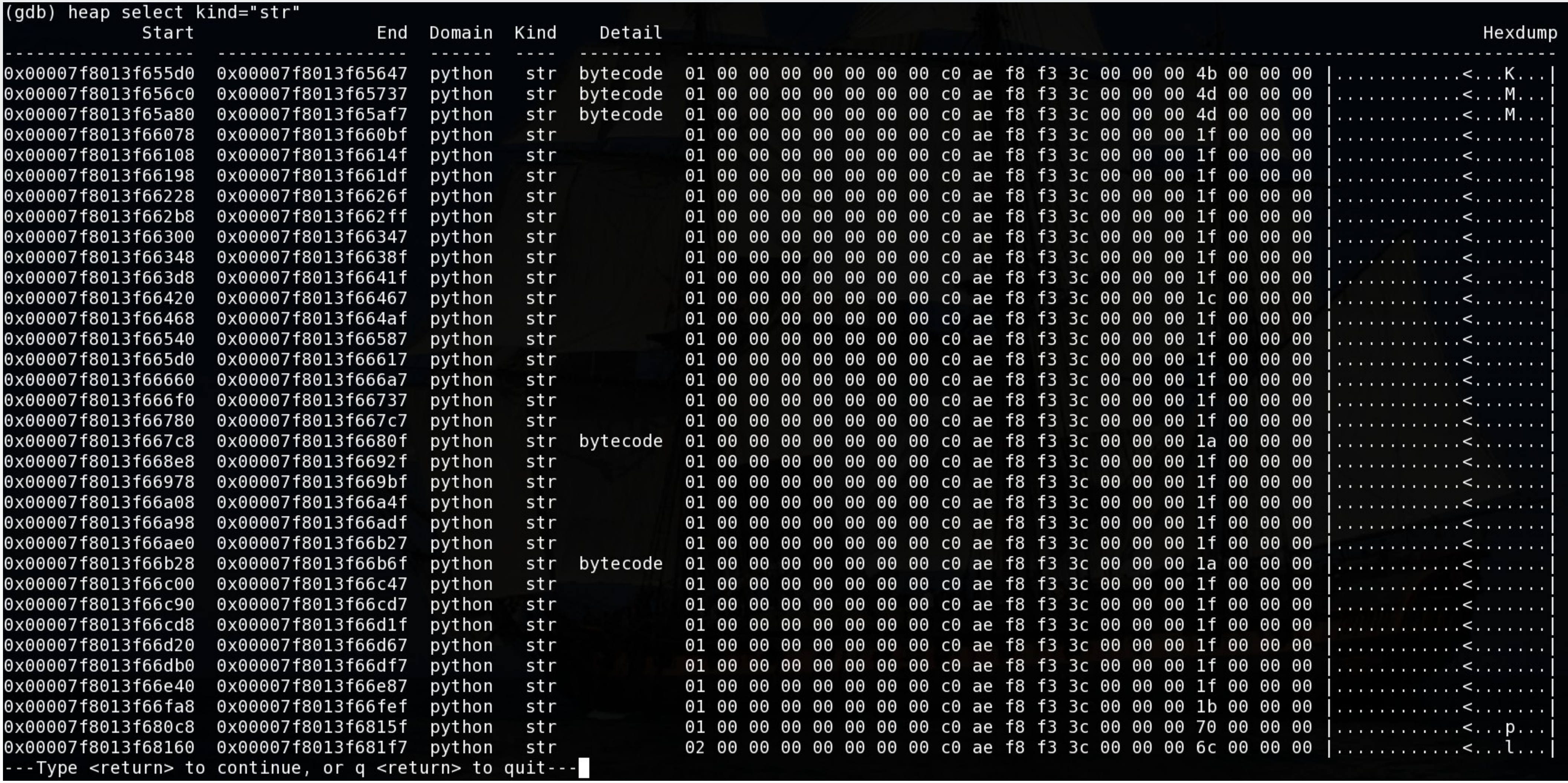


```
uncategorized                                       256 bytes       1             256
       python                     unicode                           5             240
uncategorized                                        40 bytes       6             240
       python                    _Printer                           3             192
uncategorized                                        24 bytes       7             168
       python                   sys.flags                           1             144
       python                     Quitter                           2             128
---Type <return> to continue, or q <return> to quit---
       python              sys.float_info                           1             112
       python                staticmethod                           2             112
       python                   CodecInfo                           1             104
       python                   PyCapsule                           2              96
       python                    _Environ           old-style       1              72
       python      exceptions.MemoryError                           1              72
       python     exceptions.RuntimeError                           1              72
       python                     _Helper                           1              64
       python            sys.version_info                           1              64
       python                        cell                           1              56
uncategorized                                        56 bytes       1              56
uncategorized                                        16 bytes       3              48
       python            imp.NullImporter                           3              48
      cpython      PyUnicodeObject buffer                           5              40
       python               sys.long_info                           1              40
       python                      object                           2              32
                                                        TOTAL  18,745       2,628,896
```

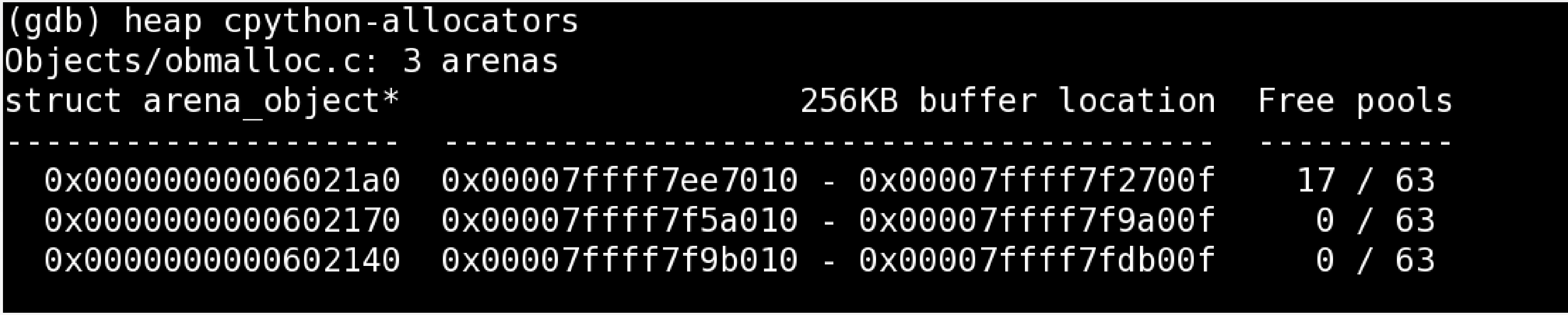


```
(gdb) heap select kind="_intblock"
             Start                  End   Domain       Kind  Detail
------------------  ------------------  -------  ---------  ------  ----------------
0x000000000061bf60  0x000000000061c34f  cpython  _intblock          00 00 00 00 00
0x000000000061c350  0x000000000061c73f  cpython  _intblock          60 bf 61 00 00
0x000000000061c740  0x000000000061cb2f  cpython  _intblock          50 c3 61 00 00
0x000000000061cb30  0x000000000061cf1f  cpython  _intblock          40 c7 61 00 00
0x000000000061cf20  0x000000000061d30f  cpython  _intblock          30 cb 61 00 00
0x000000000061d310  0x000000000061d6ff  cpython  _intblock          20 cf 61 00 00
0x000000000061d700  0x000000000061daef  cpython  _intblock          10 d3 61 00 00
0x0000000000650f00  0x00000000006512ef  cpython  _intblock          00 d7 61 00 00
0x0000000000661200  0x00000000006615ef  cpython  _intblock          00 0f 65 00 00

(gdb) 
```



```
(gdb) heap select kind="str"
             Start                 End  Domain  Kind    Detail                                                                             Hexdump
------------------  ------------------  ------  ----  --------  ----------------------------------------------------------------------------------
0x00007f8013f655d0  0x00007f8013f65647  python   str  bytecode  01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 4b 00 00 00 |............<...K...|
0x00007f8013f656c0  0x00007f8013f65737  python   str  bytecode  01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 4d 00 00 00 |............<...M...|
0x00007f8013f65a80  0x00007f8013f65af7  python   str  bytecode  01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 4d 00 00 00 |............<...M...|
0x00007f8013f66078  0x00007f8013f660bf  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66108  0x00007f8013f6614f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66198  0x00007f8013f661df  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66228  0x00007f8013f6626f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f662b8  0x00007f8013f662ff  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66300  0x00007f8013f66347  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66348  0x00007f8013f6638f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f663d8  0x00007f8013f6641f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66420  0x00007f8013f66467  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1c 00 00 00 |............<.......|
0x00007f8013f66468  0x00007f8013f664af  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66540  0x00007f8013f66587  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f665d0  0x00007f8013f66617  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66660  0x00007f8013f666a7  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f666f0  0x00007f8013f66737  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66780  0x00007f8013f667c7  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f667c8  0x00007f8013f6680f  python   str  bytecode  01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1a 00 00 00 |............<.......|
0x00007f8013f668e8  0x00007f8013f6692f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66978  0x00007f8013f669bf  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66a08  0x00007f8013f66a4f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66a98  0x00007f8013f66adf  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66ae0  0x00007f8013f66b27  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66b28  0x00007f8013f66b6f  python   str  bytecode  01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1a 00 00 00 |............<.......|
0x00007f8013f66c00  0x00007f8013f66c47  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66c90  0x00007f8013f66cd7  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66cd8  0x00007f8013f66d1f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66d20  0x00007f8013f66d67  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66db0  0x00007f8013f66df7  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66e40  0x00007f8013f66e87  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1f 00 00 00 |............<.......|
0x00007f8013f66fa8  0x00007f8013f66fef  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 1b 00 00 00 |............<.......|
0x00007f8013f680c8  0x00007f8013f6815f  python   str            01 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 70 00 00 00 |............<...p...|
0x00007f8013f68160  0x00007f8013f681f7  python   str            02 00 00 00 00 00 00 00 c0 ae f8 f3 3c 00 00 00 6c 00 00 00 |............<...l...|
---Type <return> to continue, or q <return> to quit---
```



```
(gdb) heap cpython-allocators
Objects/obmalloc.c: 3 arenas
struct arena_object*                    256KB buffer location  Free pools
--------------------  ----------------------------------------  ----------
  0x00000000006021a0  0x00007ffff7ee7010 - 0x00007ffff7f2700f    17 / 63
  0x0000000000602170  0x00007ffff7f5a010 - 0x00007ffff7f9a00f     0 / 63
  0x0000000000602140  0x00007ffff7f9b010 - 0x00007ffff7fdb00f     0 / 63
```

Fixing the problems

Fixing memory leaks: the crude and nasty way

- Kill the process at regular intervals
- Let it die, e.g. after 100 requests

For when you need to keep a system running, and haven't diagnosed the root cause of the leak yet

Obviously not ideal, but may save you in an emergency

Fixing memory leaks: the better way

Figure out the root cause

Fix that

How do memory leaks happen?

- High-tide marks: caches
- ob_refcnt errors
- tp_traverse errors
- C and C++ bugs
- SWIG errors

High-tide marks: caches

- in user code: caches that are never purged
- allocator caches that aren't purged (older pythons)
  - the fixed arena high-watermark issue (fixed in 2.5a1; also RHEL 5.6)
  - the per-type allocators (int/float/method/frame/CFunction/tuple/unicode) (fixed in 2.6alpha1 and 3.0a3)
    - See gcmodule.c:clear_freelists: http://svn.python.org/view?view=revision&revision=60797
- One-time initialization (e.g. in wrapped libraries)

Bugs: ob_refcnt

Objects with too high an ob_refcnt:

- think they're alive, but nothing references them.

Detection:

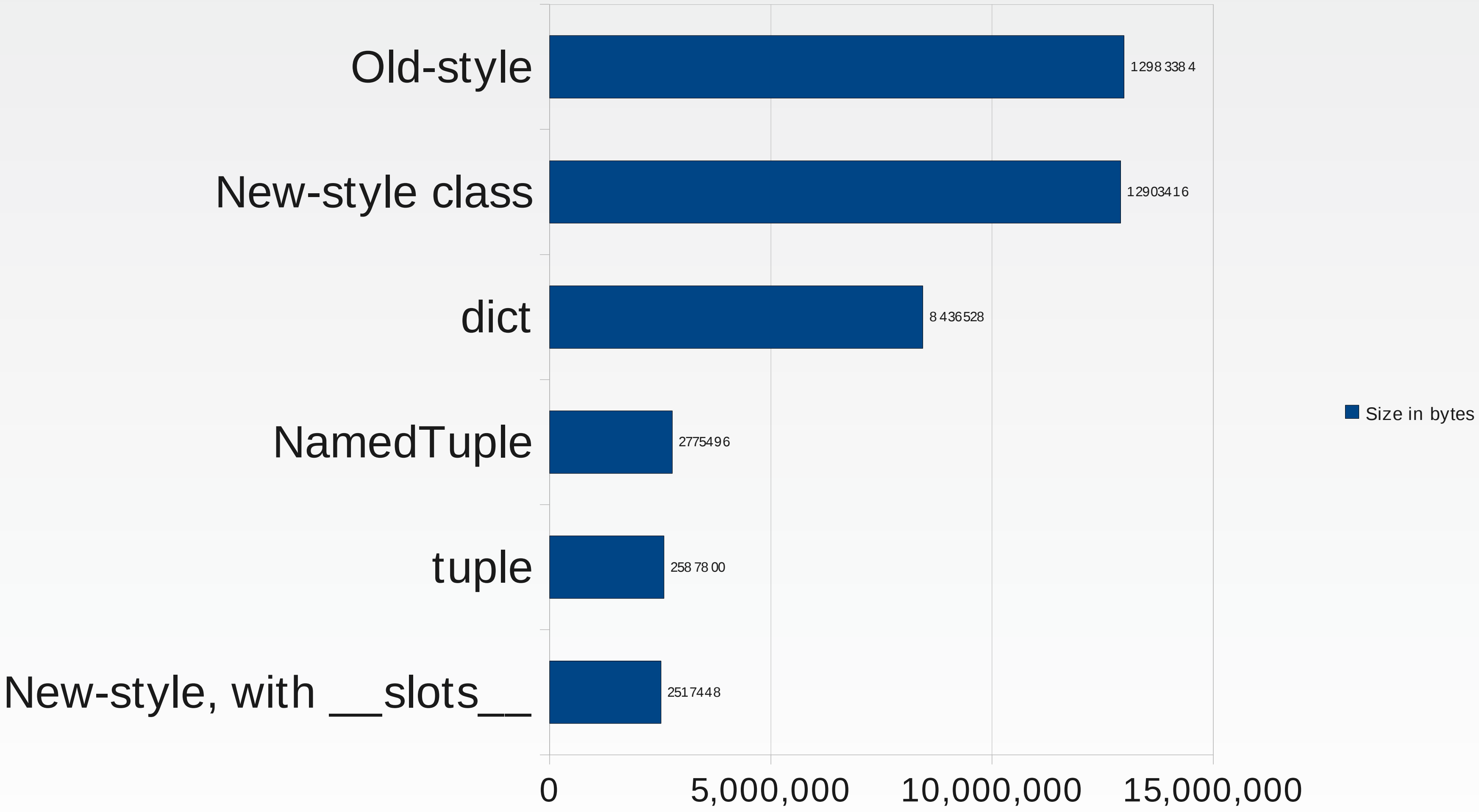
- Debug build of python
- Valgrind
- gdb-heap?
- meliae?

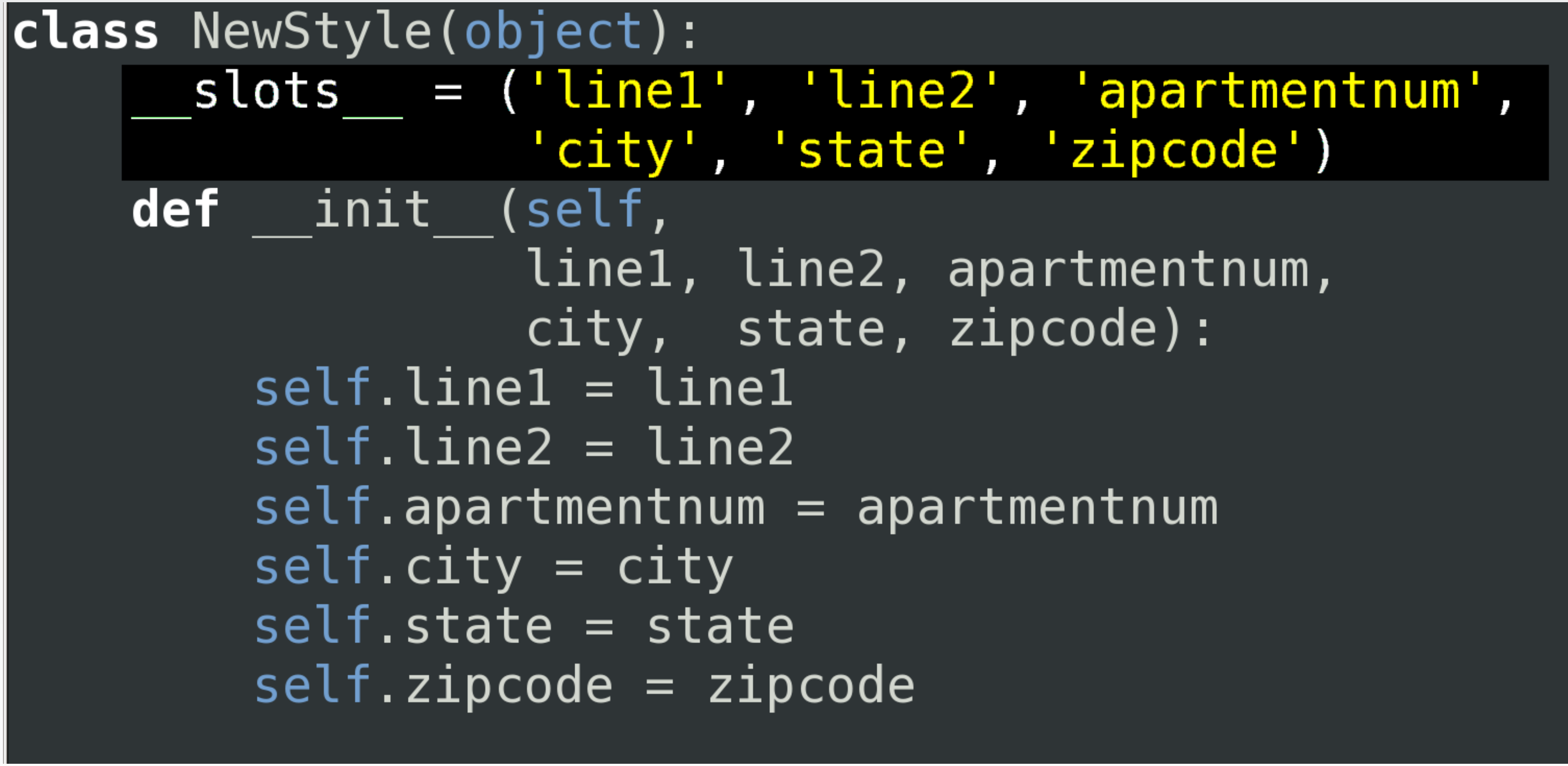
Bugs: tp_traverse

- Classes that don't call Py_VISIT on every held reference in their tp_traverse
- Classes that don't have a tp_traverse callback

Different object representations can have very different sizes

- 10,000 instances of a 6-field type (CPython)





```
class NewStyle(object):
    __slots__ = ('line1', 'line2', 'apartmentnum',
                 'city', 'state', 'zipcode')
    def __init__(self,
                 line1, line2, apartmentnum,
                 city,  state, zipcode):
        self.line1 = line1
        self.line2 = line2
        self.apartmentnum = apartmentnum
        self.city = city
        self.state = state
        self.zipcode = zipcode
```

Q & A

- Further work for gdb-heap?