Profile output for 10000 values

```
Index by function name

  [11] _GLOBAL__sub_I__Z4swapPiS_ [4] swap(int*, int*)      [5] frame_dummy
  [12] printArray(int*, int)   [2] partition(int*, int, int)
  [13] __static_initialization_and_destruction_0(int, int) [3] quickSort(int*, int, int)
it@it-OptiPlex-3050:~/Desktop$ gprof -a qs gmon.out> qs11.txt
it@it-OptiPlex-3050:~/Desktop$ cat qs11.txt
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ms/call  ms/call  name
 56.35     0.14      0.14 50004999     0.00     0.00  swap(int*, int*)
 40.25     0.24      0.10     9999     0.01     0.02  partition(int*, int, int)
  4.02     0.25      0.01                              __gmon_start__
  0.00     0.25      0.00        1     0.00     0.00  printArray(int*, int)
  0.00     0.25      0.00        1     0.00   241.49  quickSort(int*, int, int)
  0.00     0.25      0.00        1     0.00   241.49  main


 %           the percentage of the total running time of the
time         program used by this function.

cumulative  a running sum of the number of seconds accounted
 seconds     for by this function and those listed above it.

 self        the number of seconds accounted for by this
seconds      function alone.  This is the major sort for this
             listing.

calls        the number of times this function was invoked, if
             this function is profiled, else blank.

 self        the average number of milliseconds spent in this
ms/call      function per call, if this function is profiled,
             else blank.

 total       the average number of milliseconds spent in this
ms/call      function and its descendents per call, if this
             function is profiled, else blank.

name         the name of the function.  This is the minor sort
```

```
                    Call graph (explanation follows)


granularity: each sample hit covers 2 byte(s) for 3.98% of 0.25 seconds

index % time    self  children    called     name
                                                1             main [1]
                0.00    0.24       1/1          __libc_csu_init [4]
[1]     96.0    0.00    0.24       1+1       main [1]
                0.00    0.24       1/1          quickSort(int*, int, int) [3]
                0.00    0.00       1/1          printArray(int*, int) [10]
-----------------------------------------
                                                1             main [1]
                0.10    0.14    9999/9999       quickSort(int*, int, int) [3]
[2]     96.0    0.10    0.14    9999         partition(int*, int, int) [2]
                0.14    0.00 50004999/50004999    swap(int*, int*) [5]
-----------------------------------------
                                            19998             quickSort(int*, int, int) [3]
                0.00    0.24       1/1          main [1]
[3]     96.0    0.00    0.24    1+19998   quickSort(int*, int, int) [3]
                0.10    0.14    9999/9999       partition(int*, int, int) [2]
                                            19998             quickSort(int*, int, int) [3]
-----------------------------------------
                                                          <spontaneous>
[4]     96.0    0.00    0.24                 __libc_csu_init [4]
                0.00    0.24       1/1          main [1]
-----------------------------------------
                0.14    0.00 50004999/50004999    partition(int*, int, int) [2]
[5]     56.0    0.14    0.00 50004999      swap(int*, int*) [5]
-----------------------------------------
                                                          <spontaneous>
[6]      4.0    0.01    0.00                 __gmon_start__ [6]
-----------------------------------------
                0.00    0.00       1/1          main [1]
[10]     0.0    0.00    0.00       1          printArray(int*, int) [10]
-----------------------------------------

 This table describes the call tree of the program, and was sorted by
 the total amount of time spent in each function and its children.

 Each entry in this table consists of several lines.  The line with the
 index number at the left hand margin lists the current function.
```