

Flat profile:

Each sample counts as 0.01 seconds.

% time	cumulative seconds	self seconds	calls	self ms/call	total ms/call	name
89.16	0.16	0.16	30	5.35	5.35	printQ
5.57	0.17	0.01	600000	0.00	0.00	addQ
2.79	0.18	0.01	600000	0.00	0.00	newEle
2.79	0.18	0.01				lengthQ
0.00	0.18	0.00	600000	0.00	0.00	addMQ
0.00	0.18	0.00	600000	0.00	0.00	newTask
0.00	0.18	0.00	153000	0.00	0.00	isEmptyQueue
0.00	0.18	0.00	102000	0.00	0.00	isEmptyMultiQ
0.00	0.18	0.00	51000	0.00	0.00	delNextMQ
0.00	0.18	0.00	51000	0.00	0.00	delQ
0.00	0.18	0.00	51000	0.00	0.00	front
0.00	0.18	0.00	51000	0.00	0.00	nextMQ
0.00	0.18	0.00	30	0.00	0.00	sizeMQbyPriority
0.00	0.18	0.00	10	0.00	0.00	newQ
0.00	0.18	0.00	3	0.00	53.49	printMQ
0.00	0.18	0.00	3	0.00	0.00	sizeMQ
0.00	0.18	0.00	2	0.00	0.00	testDel
0.00	0.18	0.00	1	0.00	0.00	createMQ
0.00	0.18	0.00	1	0.00	15.05	loadData

%
time the percentage of the total running time of the
 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
seconds the number of seconds accounted for by this
 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
ms/call the average number of milliseconds spent in this
 function per call, if this function is profiled,
 else blank.

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

name the name of the function. This is the minor sort
 for this listing. The index shows the location of
 the function in the gprof listing. If the index is
 in parenthesis it shows where it would appear in
 the gprof listing if it were to be printed.

Copyright (C) 2012-2015 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.

Call graph (explanation follows)

granularity: each sample hit covers 2 byte(s) for 5.54% of 0.18 seconds

index	% time	self	children	called	name
<spontaneous>					
[1]	97.2	0.00	0.18		main [1]
		0.00	0.16	3/3	printMQ [3]
		0.00	0.02	1/1	loadData [4]
		0.00	0.00	3/3	sizeMQ [18]
		0.00	0.00	2/2	testDel [19]
		0.00	0.00	1/1	createMQ [20]

		0.16	0.00	30/30	printMQ [3]
[2]	88.9	0.16	0.00	30	printQ [2]

		0.00	0.16	3/3	main [1]
[3]	88.9	0.00	0.16	3	printMQ [3]
		0.16	0.00	30/30	printQ [2]

		0.00	0.02	1/1	main [1]
[4]	8.3	0.00	0.02	1	loadData [4]
		0.00	0.01	600000/600000	addMQ [5]
		0.00	0.01	600000/600000	newTask [8]

		0.00	0.01	600000/600000	loadData [4]
[5]	5.6	0.00	0.01	600000	addMQ [5]
		0.01	0.00	600000/600000	addQ [6]

		0.01	0.00	600000/600000	addMQ [5]
[6]	5.6	0.01	0.00	600000	addQ [6]

		0.01	0.00	600000/600000	newTask [8]
[7]	2.8	0.01	0.00	600000	newEle [7]

		0.00	0.01	600000/600000	loadData [4]
[8]	2.8	0.00	0.01	600000	newTask [8]
		0.01	0.00	600000/600000	newEle [7]

<spontaneous>					
[9]	2.8	0.01	0.00		lengthQ [9]

		0.00	0.00	51000/153000	nextMQ [15]
		0.00	0.00	102000/153000	isEmptyMultiQ [11]
[10]	0.0	0.00	0.00	153000	isEmptyQueue [10]

		0.00	0.00	51000/102000	nextMQ [15]
		0.00	0.00	51000/102000	delNextMQ [12]
[11]	0.0	0.00	0.00	102000	isEmptyMultiQ [11]
		0.00	0.00	102000/153000	isEmptyQueue [10]

		0.00	0.00	51000/51000	testDel [19]
[12]	0.0	0.00	0.00	51000	delNextMQ [12]
		0.00	0.00	51000/102000	isEmptyMultiQ [11]
		0.00	0.00	51000/51000	nextMQ [15]
		0.00	0.00	51000/51000	delQ [13]

		0.00	0.00	51000/51000	delNextMQ [12]
[13]	0.0	0.00	0.00	51000	delQ [13]

		0.00	0.00	51000/51000	nextMQ [15]
[14]	0.0	0.00	0.00	51000	front [14]

[15]	0.0	0.00	0.00	51000/51000	delNextMQ [12]
		0.00	0.00	51000	nextMQ [15]
		0.00	0.00	51000/102000	isEmptyMultiQ [11]
		0.00	0.00	51000/153000	isEmptyQueue [10]
		0.00	0.00	51000/51000	front [14]

[16]	0.0	0.00	0.00	30/30	sizeMQ [18]
		0.00	0.00	30	sizeMQbyPriority [16]

[17]	0.0	0.00	0.00	10/10	createMQ [20]
		0.00	0.00	10	newQ [17]

[18]	0.0	0.00	0.00	3/3	main [1]
		0.00	0.00	3	sizeMQ [18]
		0.00	0.00	30/30	sizeMQbyPriority [16]

[19]	0.0	0.00	0.00	2/2	main [1]
		0.00	0.00	2	testDel [19]
		0.00	0.00	51000/51000	delNextMQ [12]

[20]	0.0	0.00	0.00	1/1	main [1]
		0.00	0.00	1	createMQ [20]
		0.00	0.00	10/10	newQ [17]

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. The lines above it list the functions that called this function, and the lines below it list the functions this one called. This line lists:

index A unique number given to each element of the table. Index numbers are sorted numerically. The index number is printed next to every function name so it is easier to look up where the function is in the table.

% time This is the percentage of the 'total' time that was spent in this function and its children. Note that due to different viewpoints, functions excluded by options, etc, these numbers will NOT add up to 100%.

self This is the total amount of time spent in this function.

children This is the total amount of time propagated into this function by its children.

called This is the number of times the function was called. If the function called itself recursively, the number only includes non-recursive calls, and is followed by a '+' and the number of recursive calls.

name The name of the current function. The index number is printed after it. If the function is a member of a cycle, the cycle number is printed between the function's name and the index number.

For the function's parents, the fields have the following meanings:

self This is the amount of time that was propagated directly from the function into this parent.

children This is the amount of time that was propagated from the function's children into this parent.

called This is the number of times this parent called the function '/' the total number of times the function was called. Recursive calls to the function are not included in the number after the '/'.

name This is the name of the parent. The parent's index number is printed after it. If the parent is a member of a cycle, the cycle number is printed between the name and the index number.

If the parents of the function cannot be determined, the word '<spontaneous>' is printed in the 'name' field, and all the other fields are blank.

For the function's children, the fields have the following meanings:

self This is the amount of time that was propagated directly from the child into the function.

children This is the amount of time that was propagated from the child's children to the function.

called This is the number of times the function called this child '/' the total number of times the child was called. Recursive calls by the child are not listed in the number after the '/'.

name This is the name of the child. The child's index number is printed after it. If the child is a member of a cycle, the cycle number is printed between the name and the index number.

If there are any cycles (circles) in the call graph, there is an entry for the cycle-as-a-whole. This entry shows who called the cycle (as parents) and the members of the cycle (as children.) The '+' recursive calls entry shows the number of function calls that were internal to the cycle, and the calls entry for each member shows, for that member, how many times it was called from other members of the cycle.

Copyright (C) 2012-2015 Free Software Foundation, Inc.

Copying and distribution of this file, with or without modification,
are permitted in any medium without royalty provided the copyright
notice and this notice are preserved.

Index by function name

[5] addMQ	[10] isEmptyQueue	[3] printMQ
[6] addQ	[9] lengthQ	[2] printQ
[20] createMQ	[4] loadData	[18] sizeMQ
[12] delNextMQ	[7] newEle	[16] sizeMQbyPriority
[13] delQ	[17] newQ	[19] testDel
[14] front	[8] newTask	
[11] isEmptyMultiQ	[15] nextMQ	