Program Structures and Algorithms
Spring 2023(SEC –8)

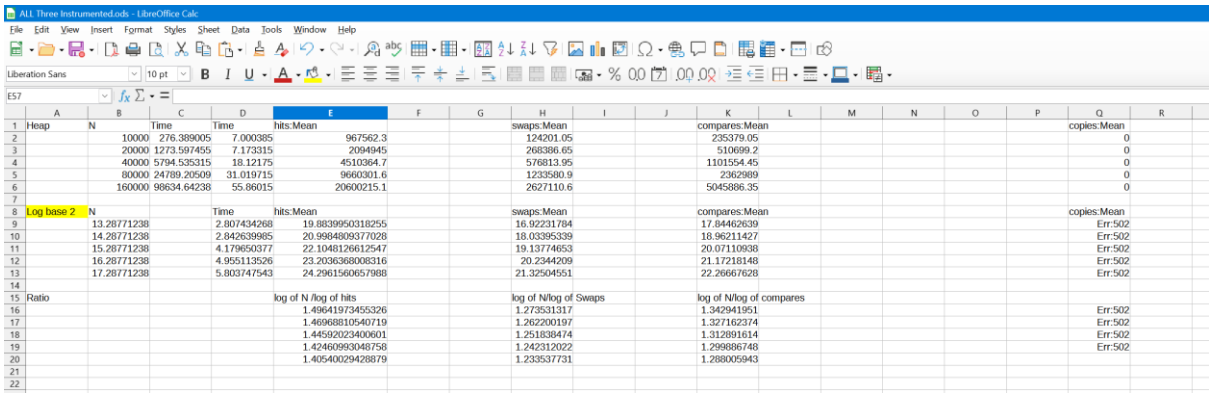
NAME: Mehul Natu
NUID: 002743870

Task: To determine - for sorting algorithms -what is the best predictor of total execution time: comparisons, swaps/copies, hits (array accesses), or something else.

Relationship Conclusion: Hits or array access would be the best predictor for time of an algorithm along with specific factors related to that algorithm in our case it is copies.

Evidence to support that conclusion:
Hits are the most consistent part of a sorting algorithm its used in every operation weather it be copy, swaps, compare, its used everywhere. So it should have a significant impact on sorting the algorithm. Also copies can take a lot of time as and then accessing those copies can create one more problem since we know there would be limited amount of cache available to us now if we are operating on two array then we will be providing less cache to each array thus creating higher probability of cache misses. Which could in-turn hurt our performance. So in general we should look for hits and factors specific to a sort to understand how well it can perform.
Assuming that N is directly co-related to time. Also the relation shown between different factors and N should not be taken as the truth those relations are only meant for comparisons.

Heap Sort
Data for different Algorithms are as shown for Heap Sort -
As seen from the Ratio of Log of N by log of hits. The relation between hits would $N = hits^{1.4}$. And similarly for swaps and compares it is – $N = compares^{1.3}$, $N = swaps^{1.25}$. So for Heap sort best predictor apart from hits is compares. And for copies N = 0



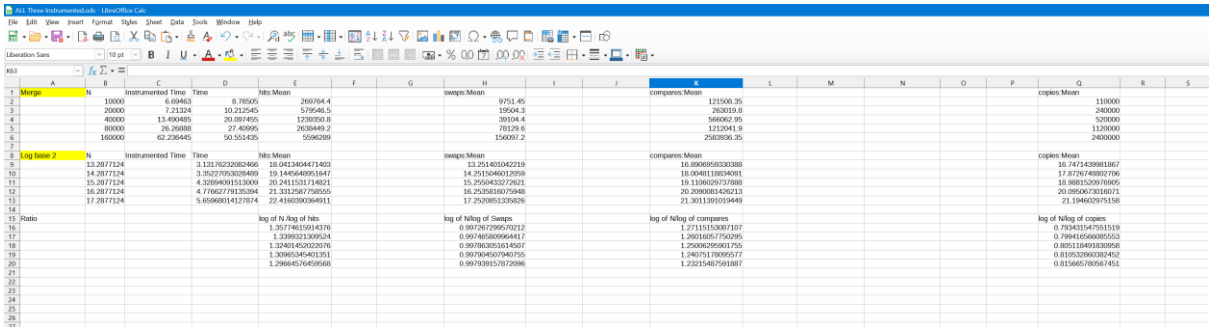
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Heap	N	Time	Time	hits:Mean			swaps:Mean			compares:Mean						copies:Mean	
2		10000	276.389005	7.000385	967562.3			124201.05			235379.05						0	
3		20000	1273.597455	7.173315	2094945			268386.65			510699.2						0	
4		40000	5794.535315	18.12175	4510364.7			576813.95			1101554.45						0	
5		80000	24789.20509	31.019715	9660301.6			1233580.9			2362989						0	
6		160000	98634.64238	55.86015	20600215.1			2627110.6			5045886.35						0	
7																		
8	Log base 2	N		Time	hits:Mean			swaps:Mean			compares:Mean						copies:Mean	
9		13.28771238		2.807434268	19.8839950318255			16.92231784			17.84462639						Err:502	
10		14.28771238		2.842639985	20.9984809377028			18.03395339			18.96211427						Err:502	
11		15.28771238		4.179650377	22.1048126612547			19.13774653			20.07110938						Err:502	
12		16.28771238		4.955113526	23.20363680083 16			20.2344209			21.17218148						Err:502	
13		17.28771238		5.803747543	24.2961560657988			21.32504551			22.26667628						Err:502	
14																		
15	Ratio				log of N /log of hits			log of N/log of Swaps			log of N/log of compares							
16					1.49641973455326			1.273531317			1.342941951						Err:502	
17					1.46968810540719			1.262200197			1.327162374						Err:502	
18					1.44592023400601			1.251838474			1.312891614						Err:502	
19					1.42460993048758			1.242312022			1.299886748						Err:502	
20					1.40540029428879			1.233537731			1.288005943							
21																		
22																		

Merge Sort
For Merge sort it appears that the log ratio LogN to log of hits is the highest, and the follows ratio for compare , then ratio of swaps and then ratio of copies.

Ranking -

1. Hits– $N = hits^{1.3}$
2. Compares – $N = compares^{1.25}$
3. Swaps – $N = swaps^{0.99}$

4. Copies – N – $copies^{0.80}$



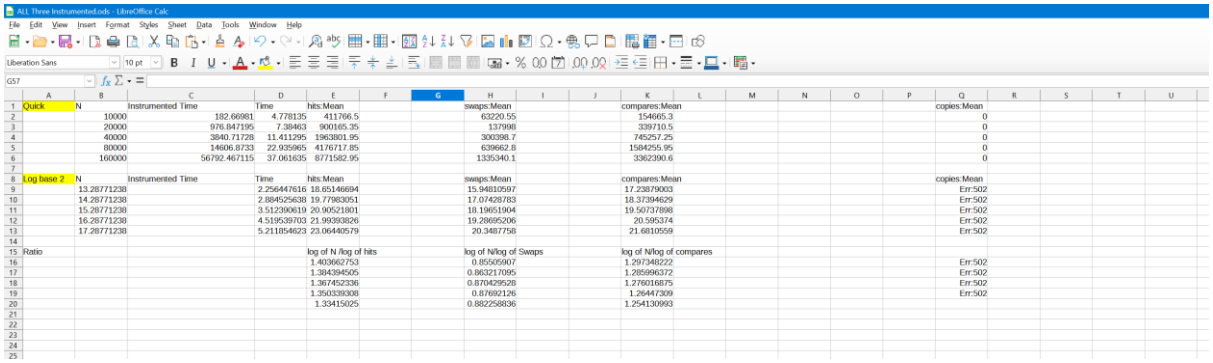
Merge	N	Instrumented Time	Time	hits:Mean	swaps:Mean	compares:Mean	copies:Mean
	10000	6.69463	8.76505	269764.4	9751.45	121508.35	110000
	20000	7.21324	10.212545	579546.5	19504.3	263019.8	240000
	40000	13.490485	20.097455	1239350.8	39104.4	566062.95	520000
	80000	26.26888	27.40995	2638449.2	78129.6	1212041.9	1120000
	160000	62.236445	50.551435	5596289	156097.2	2583936.35	2400000
Log base 2	N	Instrumented Time	Time	hits:Mean	swaps:Mean	compares:Mean	copies:Mean
	13.2877124		3.1317623282466	18.041340447403	13.251401042219	16.8906959330388	16.7471439981867
	14.2877124		3.3522705302889	19.144564895164	14.251046012059	18.0048118834091	17.8726748802706
	15.2877124		4.3289409153009	20.241153171482	15.250433272621	19.1106029737888	18.9881520976905
	16.2877124		4.7766277913528	21.331258775855	16.252581607594	20.2090881426213	20.0950673016071
	17.2877124		5.6596801412787	22.416039036491	17.252085133582	21.3011391019449	21.194602975158
Ratio				log of N /log of hits	log of N/log of Swaps	log of N/log of compares	log of N/log of copies
				1.3577461501437	0.997267299570212	1.27115153087107	0.7934315475515
				1.339932130952	0.997465809964417	1.26016057750295	0.7994165660855
				1.324014520207	0.997983051614507	1.25006295901755	0.8051184918309
				1.309653454013	0.997904507940755	1.24075178095577	0.8105326603824
				1.296845764595	0.997939157872096	1.23215487591887	0.8156657805674

Quick Sort

For Quick sort it appears that the log ratio LogN to log of hits is the highest, and the follows ratio for compare , then ratio of swaps and then ratio of copies.

Ranking -

1. Hits– N = $hits^{1.4}$
2. Compares – N = $compares^{1.27}$
3. Swaps – N = $swaps^{0.85}$
4. Copies – N = 0



Quick	N	Instrumented Time	Time	hits:Mean	swaps:Mean	compares:Mean	copies:Mean
	10000	182.66981	4.778135	411766.5	63220.55	154665.3	0
	20000	976.847195	7.38463	900165.35	137998	339710.5	0
	40000	3840.71728	11.411295	1963801.95	300398.7	745257.25	0
	80000	14606.8733	22.935965	4176717.85	639662.8	1584255.95	0
	160000	56792.467115	37.061635	8771582.95	1335340.1	3362390.6	0
Log base 2	N	Instrumented Time	Time	hits:Mean	swaps:Mean	compares:Mean	copies:Mean
	13.28771238		2.256447616	18.65146694	15.94810597	17.23879003	Err:502
	14.28771238		2.884525638	19.77983051	17.07428783	18.37394629	Err:502
	15.28771238		3.512390619	20.90521801	18.19651904	19.50737898	Err:502
	16.28771238		4.519539703	21.99393826	19.28695206	20.595374	Err:502
	17.28771238		5.211854623	23.06440579	20.3487758	21.6810559	Err:502
Ratio				log of N /log of hits	log of N/log of Swaps	log of N/log of compares	
				1.403662753	0.855059007	1.297348222	Err:502
				1.384394505	0.863217095	1.285996372	Err:502
				1.367452336	0.870429528	1.276016875	Err:502
				1.350339308	0.87692126	1.26447309	Err:502
				1.33415025	0.882258836	1.254130993	

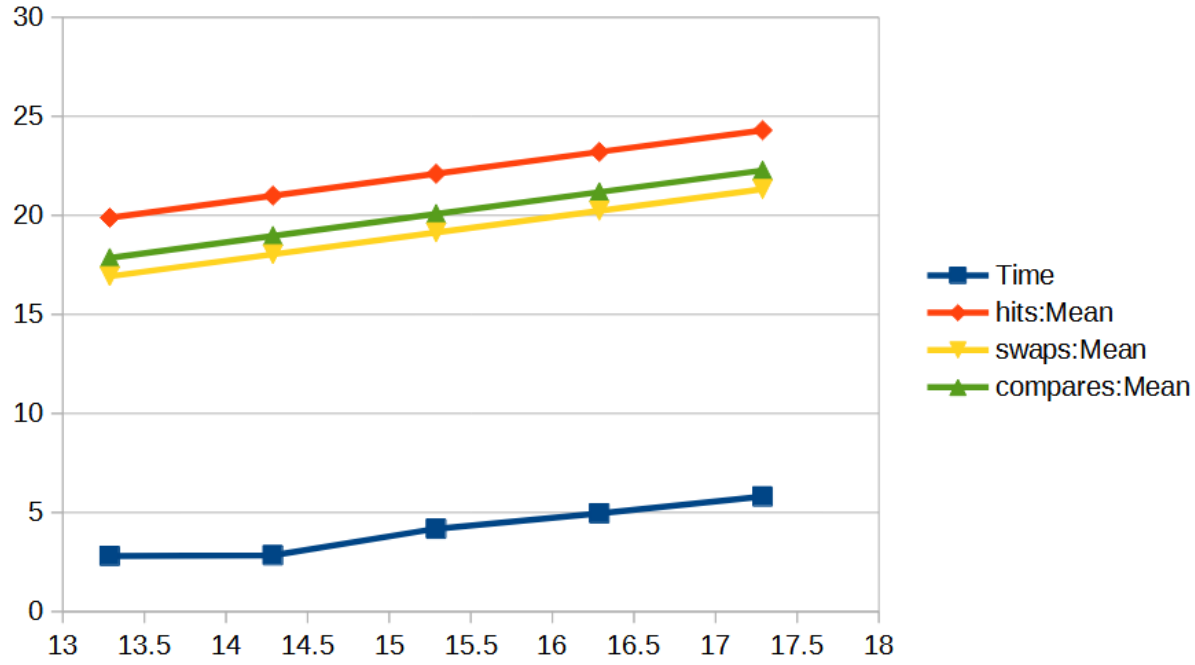
As we can see from this data Best time complexity is of Quick sort in general but why if it has 2^{nd} lowest hit counts and 1^{st} is of Merge sort. This could be explained because Merge sort uses Copies which quick sort does not and copies are related to N with power of 0.80 which could become significant with higher values of N. and since there is not much difference between hits for merge sort and quick sort. This becomes a huge factor since quick sort does not have factor for copies. And heap sort is the worst because it's hit count is relatively greater than merge and quick sort.

Also we can see from all three sorting algo that the most co-related factor with N is hits. And since copies is only the factor which is present in Merge sort and not in quick and heap sort it becomes a significant factor as it is adding to time complexity only for merge sort. So in general we should look for hits and factors specific to a sort algorithm to understand how well it can perform.

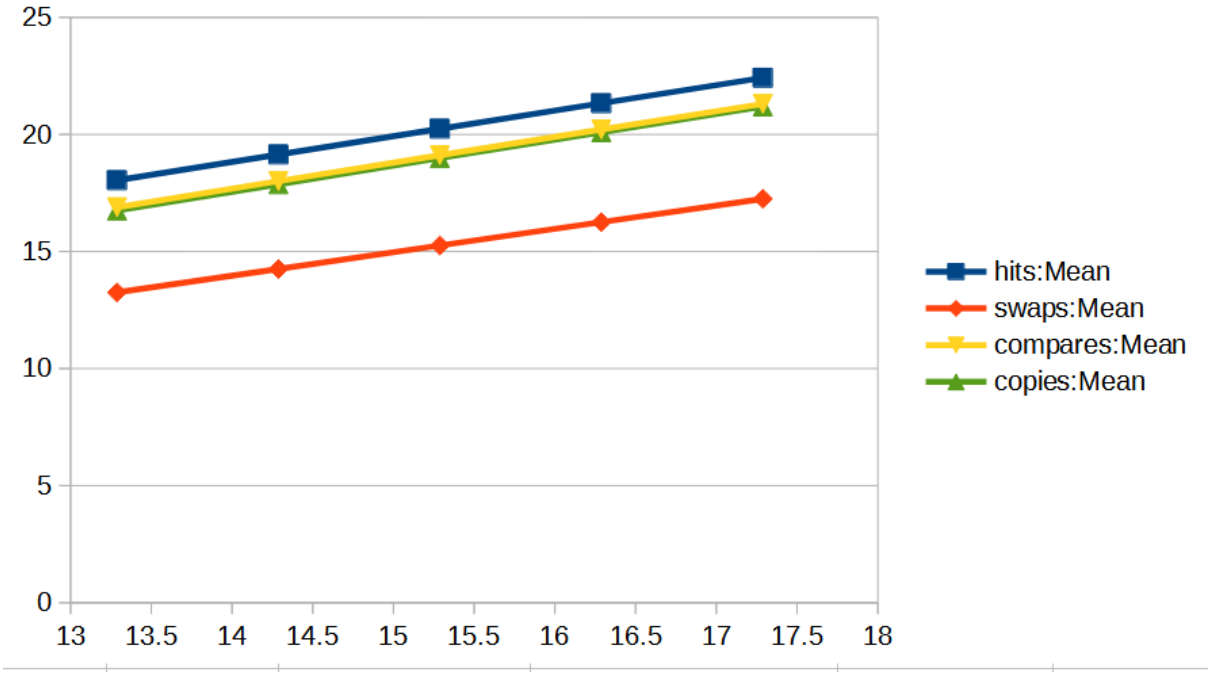
Graphical Representation:

Graph between Log of hits, compares, swaps, copies vs log of N

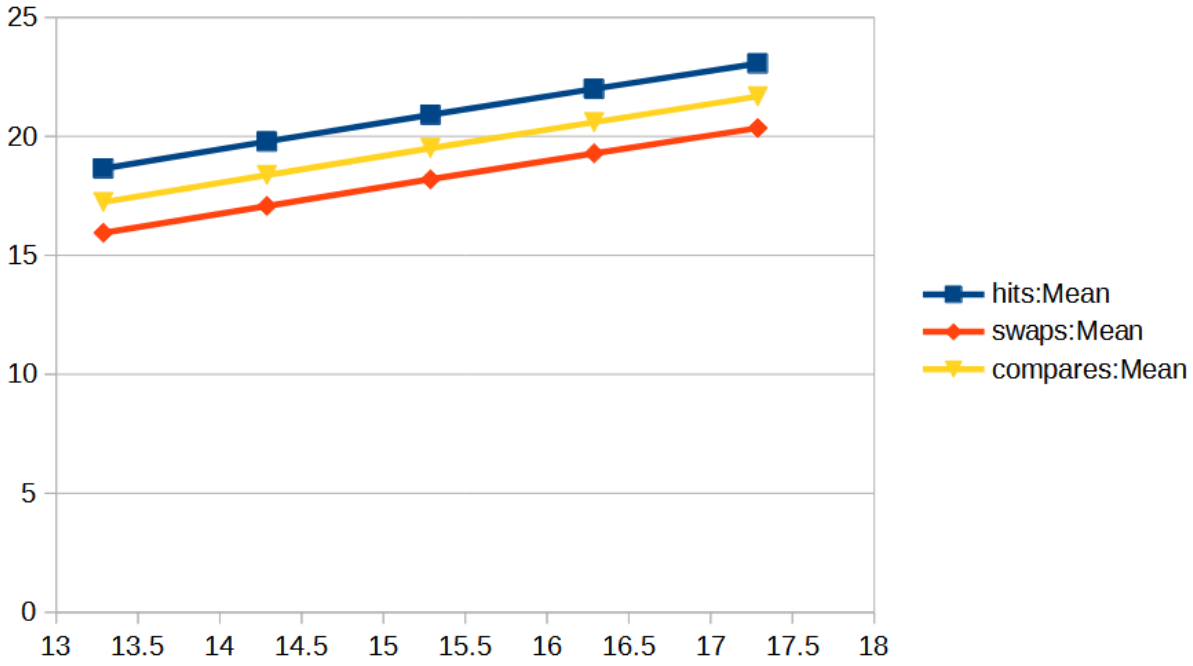
Heap Sort -



Merge Sort -



Quick Sort -


25
20
15
10
5
0
13
13.5
14
14.5
15
15.5
16
16.5
17
17.5
18
hits:Mean
swaps:Mean
compares:Mean