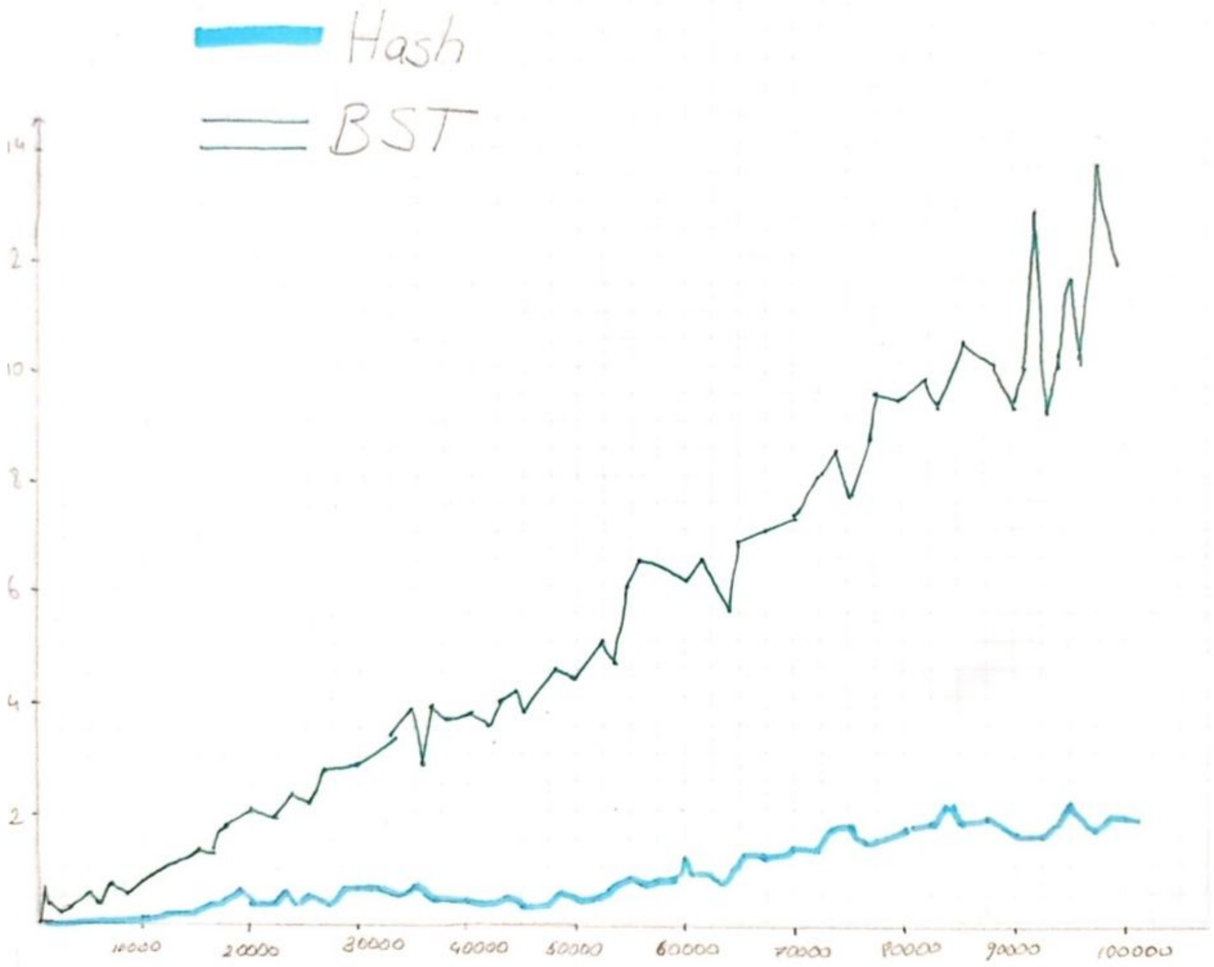
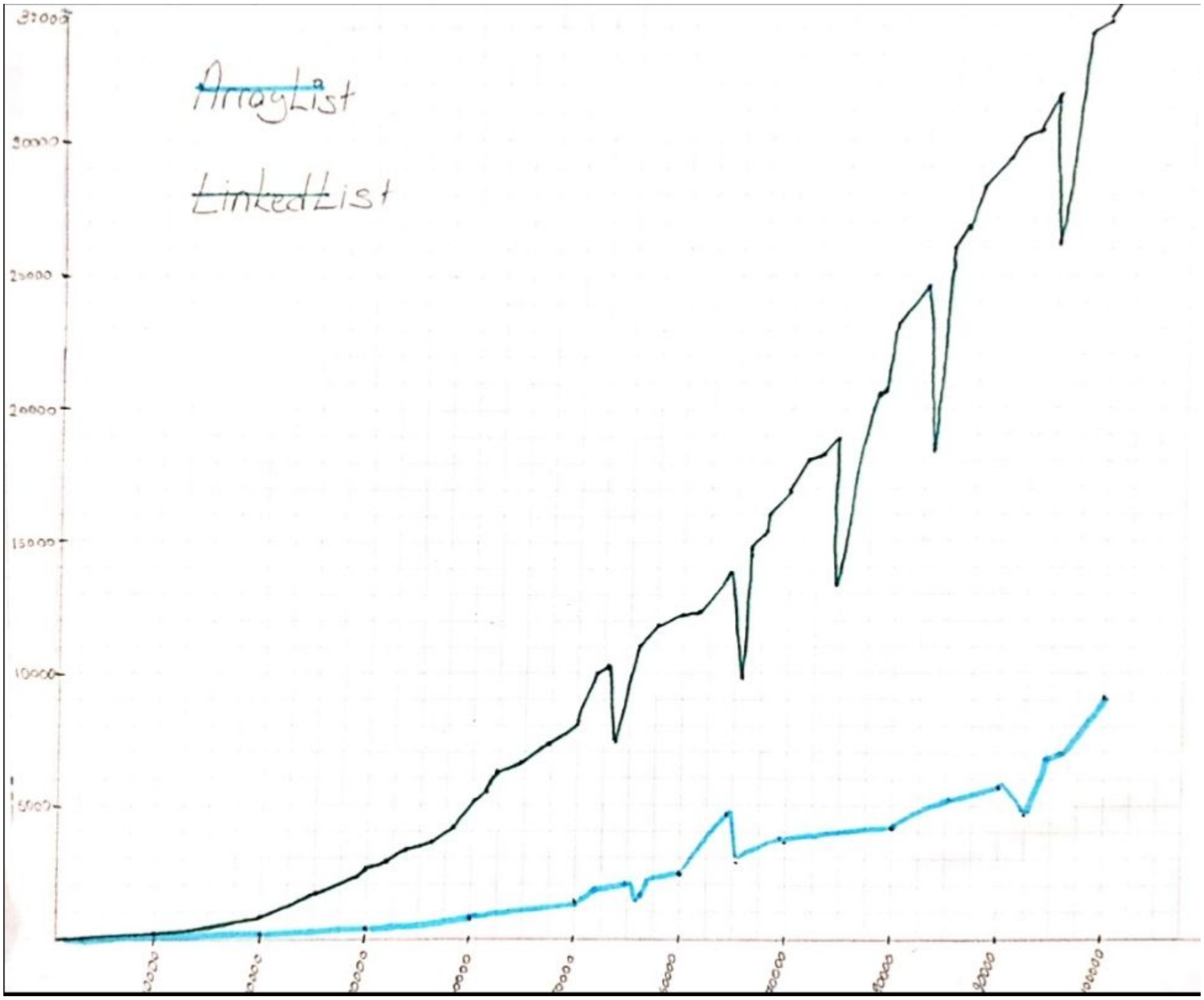
Search Comparison

This project calculates search times for ArrayList, LinkedList, Binary Search Tree and HashMap. I first declared my ArrayList then shuffled. I wanted to get the best results of my project so I copied that ArrayList and pasted the other 3 of them.

I calculated the time individually for every method. I used a while loop which increases array size 1000 every time it completes the search.



This is a graph shows the increasement of HashMap and BST by millisecond-arraySize.



This is a graph shows the increasement of ArrayList and LinkedList by millisecond-arraySize.

OUTPUT OF MY ASSIGNMENT:

```
1000   4.5183      4.3519      0.6398      0.1984
2000   8.1217      3.7281      0.3959      0.1125
3000   18.209      3.4155      0.291      0.1781
4000   32.714      6.1848      0.3736      0.1882
5000   50.994      9.793      0.541      0.2834
6000   73.447      14.0054      0.4775      0.18
7000   98.338      19.1685      0.7811      0.2464
8000   128.326      25.2478      0.6015      0.1203
9000   166.6268      32.0211      0.6664      0.1272
10000  204.55      40.3818      0.7655      0.1552
11000  243.4565      48.8557      0.9045      0.1843
12000  298.0954      59.4335      0.9838      0.213
13000  353.1946      73.5288      1.1297      0.2056
14000  411.4877      82.1074      1.221      0.2418
15000  477.9282      96.0191      1.3334      0.2459
16000  554.2455      111.4266      1.4113      0.2732
17000  636.115      126.5892      1.4835      0.3076
18000  720.8769      143.1898      1.5312      0.5552
19000  815.3436      161.0607      1.6915      0.3446
20000  934.4583      180.7657      1.7543      0.406
21221  1038.723      200.7033      1.8929      0.3988
22000  1194.606      222.2201      2.0425      0.3933
```

23000	1442.4916	245.3246	2.1666	0.4691
24000	1576.4054	269.741	2.3487	0.5335
25000	1781.5957	298.547	2.2999	0.4227
26000	1954.3875	319.5695	2.3104	0.4488
27000	2151.7755	347.49	2.4769	0.4672
28000	2321.3462	376.0406	2.5775	0.5673
29000	2506.2546	405.9915	2.7562	0.5214
30000	2689.5324	436.818	2.8297	0.5757
31000	2896.7787	465.8864	2.9349	0.5652
32000	3113.8396	501.1259	3.2365	0.6544
33000	3343.7051	533.5196	3.0668	0.6108
34000	3571.8819	570.2228	3.4489	0.6994
35000	3792.1127	605.7116	3.2761	0.6452
36000	4018.7777	644.34	3.4827	0.7714
37000	4297.0469	695.7247	3.5011	0.6729
38000	4538.2368	722.6127	3.3223	0.4529
39000	4809.9055	763.1779	3.4585	0.4663
40000	5077.5363	805.0887	3.7254	0.4983
41000	5350.1508	849.9933	4.0741	0.7327
42000	5630.3135	893.6156	3.6586	0.4991
43000	5961.2945	939.5516	4.0518	0.7012
44000	6238.3648	985.7952	4.4322	0.5525
45000	6509.5142	1035.2691	3.8666	0.6263
46000	6810.4123	1083.1073	4.1271	0.6507
47000	7146.1118	1134.1365	4.0109	0.6607
48000	7521.1708	1198.1223	4.1948	0.6546
49000	7874.036	1243.8752	4.437	0.6831
50000	8156.2826	1303.3656	4.3734	0.494
51000	8597.2019	1366.5153	4.5547	0.6042
52000	7262.7353	1496.1094	4.859	0.6006
53000	9237.5272	1647.1371	4.5969	0.7202
54000	9700.9204	1696.5386	4.8892	0.5967
55000	9970.0933	1723.1493	5.2393	0.7915
56000	10587.0061	1867.9293	4.934	0.6487
57000	10918.5488	1955.35	5.422	0.9085
58000	11306.2411	2081.1305	5.2719	0.7225
59000	11816.4405	2178.4621	5.3071	0.7748
60000	12254.9545	2353.3213	6.171	1.1315
61000	12587.4067	2477.5668	5.5554	0.7809
62000	12925.8155	2582.2261	5.5183	0.7898
63000	13397.4152	2676.6341	5.5938	0.8447
64000	9349.0273	4879.7492	14.5222	0.7888
65000	14415.2913	2955.4319	5.8409	0.8395
66000	14915.3875	3072.0874	6.215	0.8218
67000	15359.1359	3197.5313	6.3749	0.8823
68000	15871.4074	3341.815	6.2274	0.8424
69000	16431.5150	3488.1458	6.2924	0.8735
70000	17076.2436	3626.3667	7.261	1.3494
71000	17329.7165	3760.6817	7.8911	1.3982
72000	18203.1387	3907.9427	8.5701	1.4292
73000	18591.1643	4086.5895	8.8749	1.2612
74000	15587.5072	4175.1106	7.0071	0.9891
75000	19503.0942	4383.4527	8.2514	1.5662
76000	20014.3143	4508.3932	8.8901	1.6531
77000	20954.8338	4637.2248	7.8062	1.3638
78000	21763.6377	4793.8566	7.4398	1.0914
79000	22294.2936	4967.3775	8.1546	1.2984
80000	23288.8212	5135.3382	9.3142	1.6236
81000	23456.9946	5263.166	8.2847	1.2711
82000	24140.7578	5473.7997	9.5154	1.6562
83000	11393.0831	4273.1303	10.748	1.7466
84000	26791.7092	5774.6443	8.383	1.2532
85000	26302.7086	6015.5362	10.6328	1.754
86000	26784.2851	6175.3936	8.2451	1.3165
87000	26954.6939	6341.9378	10.0872	1.7617
88000	28412.678	6473.525	10.3929	1.6529
89000	28135.4691	6635.0471	10.6955	1.5029
90000	29343.5742	6852.5613	9.1766	1.5694
91000	13856.0253	4721.4906	9.8605	1.6316
92000	30517.918	7236.0318	12.6305	1.8979
93000	31129.5049	7403.9066	8.9954	1.5108
94000	32424.7865	7624.0457	9.643	1.6337
95000	32492.1104	7858.1731	12.1585	2.0419
96000	33428.0075	8062.8235	10.1793	1.8463
97000	33997.2597	8325.837	11.4932	2.1076
98000	25631.8805	8509.0361	10.1439	1.6798
99000	35804.2019	8741.4908	13.5438	2.0485
100000	36691.1439	9072.7239	11.6194	1.8738

As we can see from those graphs and output, the HASH is faster than all of them. Second one is BST while third is ArrayList. LinkedList is our fourth since it takes too long to search.

HashMap < Bst < ArrayList < LinkedList