Number of Searches 10,000

Seconds to sort	Program 0	Program 3
120,000 elements	2.998 seconds	0.012 seconds
240,000 elements	11.812 seconds	0.02 seconds
480,000 elements	47.889 seconds	0.037 seconds

Number of Searches 10,000=y

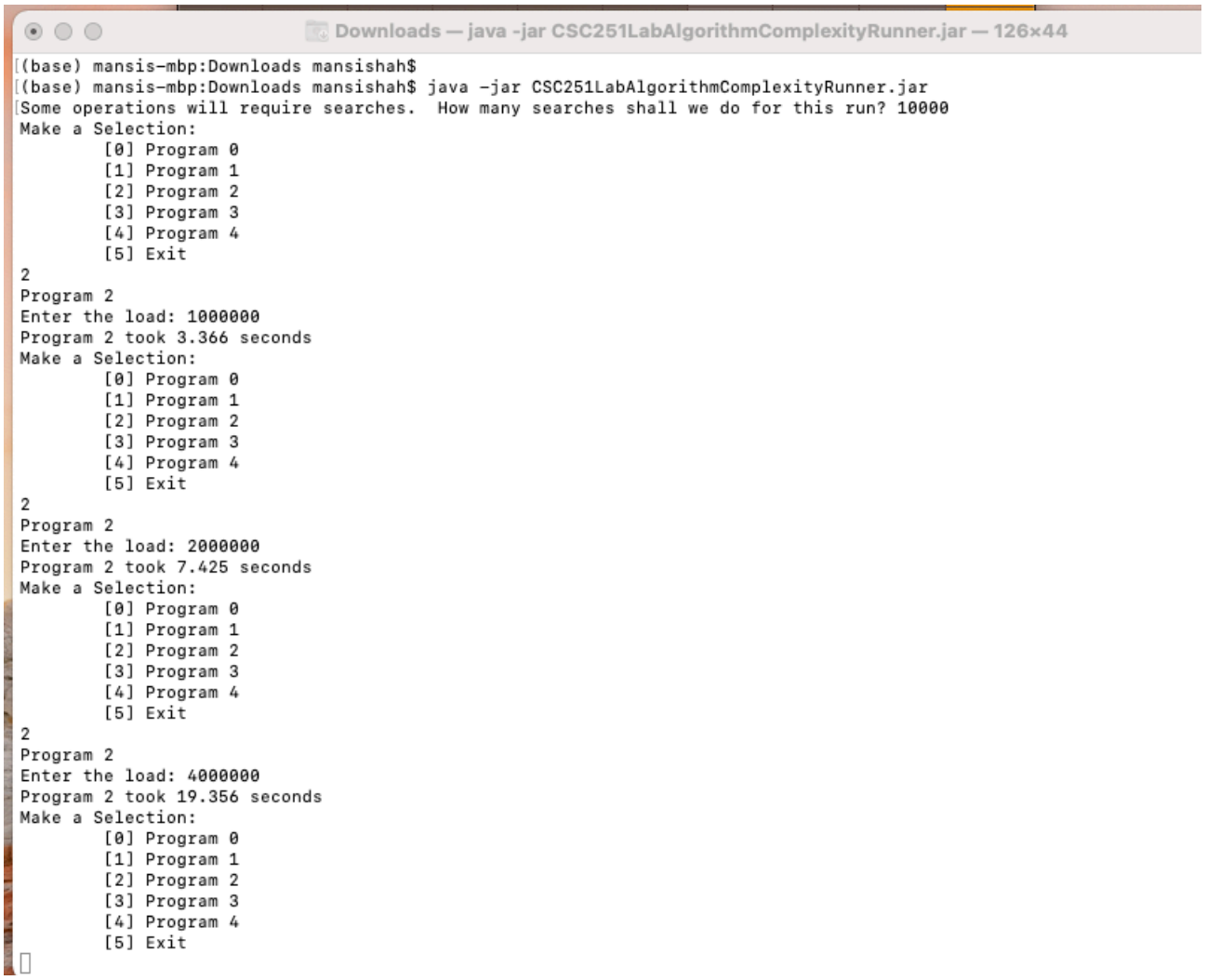
Seconds to search for y elements in a collection of number of elements	Program 2	Program 1	Program 4
1000,000 elements	3.366 seconds	0.004 seconds	0.002 seconds
2000,000 elements	7.425 seconds	0.005 seconds	0.001 seconds
4000,000 elements	19.356 seconds	0.006 seconds	0.001 seconds

Screenshot: Number of searches has been kept same for search and sort methods 10,000

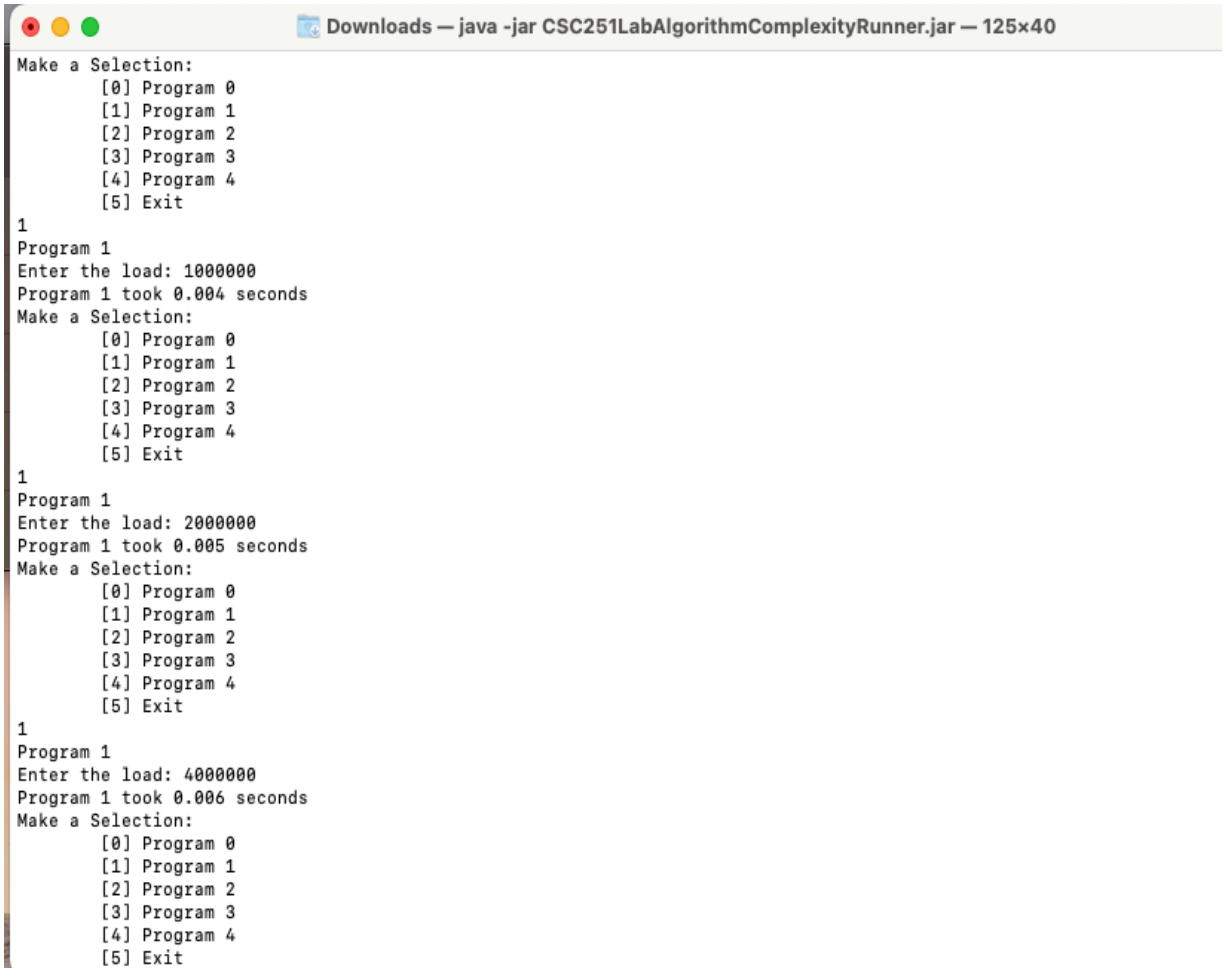
1) Run program 0 for three different workloads

2) Run program 3 for three different workloads

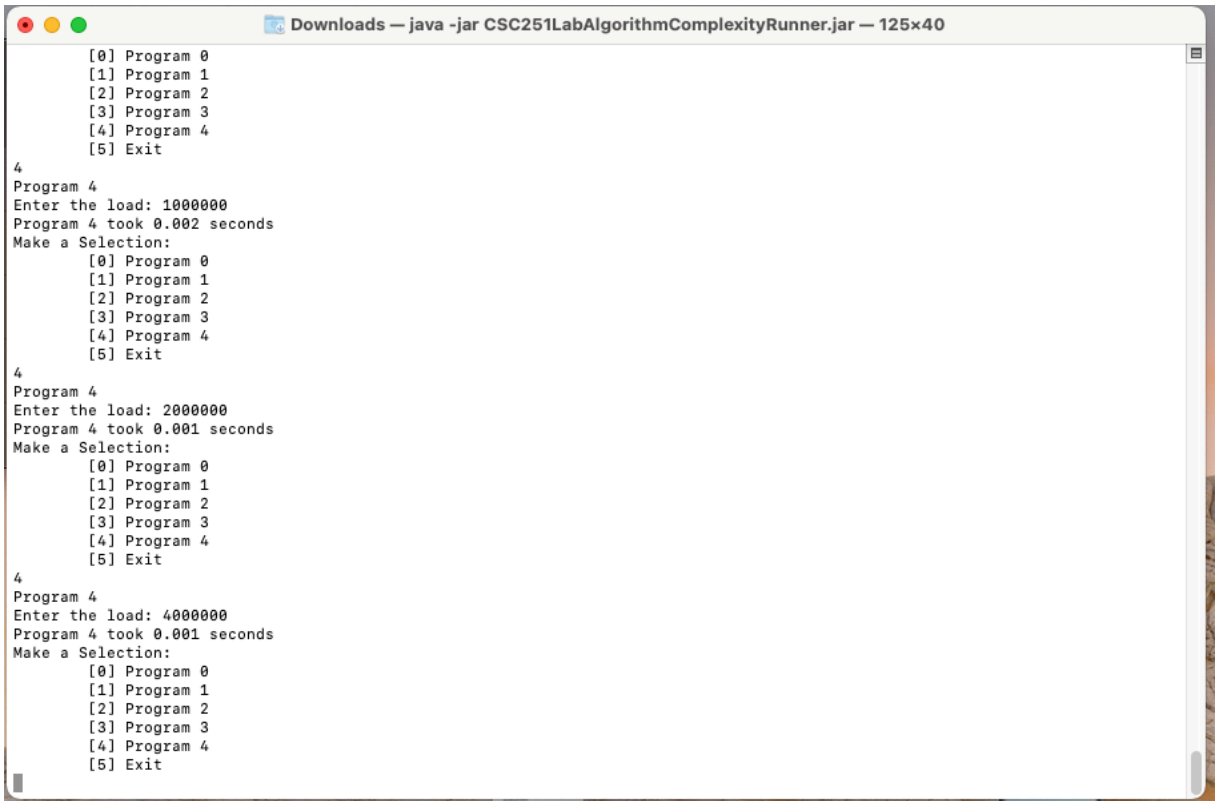
3) Program 2 for three different workloads:



4) Program 1 for three different workloads:



5) Program 4 for three different workloads:



Conclusion:

1) Among the sort algorithms, program 3 is more efficient than program 0.
   Built in java array sort method is more efficient than Selection sort method

2) Among search algorithms, program 4 is the most efficient followed by program 1 and least efficient is program 2.
   Hash-set search method is most efficient followed by
   Binary Search method followed by
   Linear search method

   Following comparison shows how efficiency changes depending on method we choose for same load.

Search Methods:

Linear Search Program 2	Binary Search Program 1	Hash-set Search Program 4
Look at the list, One item at a time	Search for an item in sorted array	Use hash table. Elements are not ordered.
Time required to run program is linearly related to the size of input	Growth rate= logarithmic	Constant growth rate
Double the size= double the time	Time grows slowly as input size increases	No change with size of input

Sort methods

Selection sort method (program 0)	Built-in Java Array sort method (program 3)
Find the smallest element from list, Swap with first element, Find smallest element from remaining of list, Swap with remaining list	Converting elements in string, and then comparing their sequence
Every time it goes through the list, time required to run program grows with power	Growth rate is logarithmic