Homework 1

Graded

Student

Abdul Fayeed Abdul Kadir

Total Points

36 / 40 pts

Question 1

Problem 1 — **8** / 8 pts

1.1 Part A — **8** / 8 pts

- ✔ **+ 1 pt** A) Compares to 9 and 12
- ✔ **+ 1 pt** A) Two comparisons
- ✔ **+ 1 pt** B) Compares to 9,12,15, and 20
- ✔ **+ 1 pt** B) Four comparisons
- ✔ **+ 1 pt** C) Compares to 9, 4, and 2
- ✔ **+ 1 pt** C) Three comparisons
- ✔ **+ 1 pt** D) Compares to 8, 4 , and 3
- ✔ **+ 1 pt** D) Not valid because not sorted - this explanation should be explicit

+ 0 pts Fully incorrect/lack of substantial attempt

Question 2

Problem 2 — **8** / 8 pts

- ✔ **+ 1 pt** Iteration 1: (2,4,10,6,20,11)
- ✔ **+ 1 pt** Iteration 2: (2,4,10,6,20,11)
- ✔ **+ 1 pt** Iteration 3: (2,4,6,10,20,11)
- ✔ **+ 1 pt** Iteration 4: (2,4,6,10,20,11)
- ✔ **+ 1 pt** Iteration 5: (2,4,6,10,11,20)
- ✔ **+ 3 pts** 15 comparisons total

+ 0 pts Fully incorrect/lack of substantial attempt

Great job!

Question 3

Problem 3

8 / 8 pts

- ✔ **+ 1 pt** Iteration 1: (4,12,20,6,2,11)
- ✔ **+ 1 pt** Iteration 2: (4,12,20,6,2,11)
- ✔ **+ 1 pt** Iteration 3: (4,6,12,20,2,11)
- ✔ **+ 1 pt** Iteration 4: (2,4,6,12,20,11)
- ✔ **+ 1 pt** Iteration 5: (2,4,6,11,12,20)
- ✔ **+ 3 pts** 12 comparisons total
- **+ 0 pts** Fully incorrect/lack of substantial attempt
- Good job!

Question 4

Problem 4

4 / 8 pts

- ✔ **+ 2 pts** Linear search requires 50,000 comparisons
- **+ 2 pts** Binary search requires 16 comparisons
- ✔ **+ 2 pts** Selection sort requires 1,249,975,000 comparisons
- **+ 2 pts** 25008 searches needed
- **+ 1 pt** 25007 searches reported - incorrect, need to round-up to actually make Selection Sort + Binary Search worth it.
- **+ 6 pts** Correct final answer, but insufficient/no work
- **+ 0 pts** Fully incorrect/lack of substantial attempt
- Binary search takes floor(lg(50,000)) + 1 = 16 comparisons in the worst case (-2 points) so make sure you remember the floor it is super important! This will change your answer so when s = 25008 it is quicker to use selection sort+binary (-2 points)

Question 5

Problem 5

8 / 8 pts

- ✔ **+ 8 pts** Readable and works
- **+ 6 pts** Partial, needs some reworking
- **+ 4 pts** Partial, needs lots of work
- **+ 0 pts** Lack of substantial attempt/no attempt
- Great Job!

Question assigned to the following page: 1.1

Problem 1 Consider the binary search algorithm we saw in lecture. Suppose A=(2,4,5,6,9,11,12,15,20).

a) Suppose we are looking for the value 12. Which numbers will be compared with 12? How many total comparisons were performed?

- Number 9 and 12
- Two comparisons

b) Suppose we are looking for the value 16. Which numbers will be compared with 16? How many total comparisons were performed?

- Number 9, 12, 15, 20
- Four comparisons

c) Suppose we are looking for the value 0. Which numbers will be compared with 0? How many total comparisons were performed?

- Number 9, 4, 2
- Three comparisons

d) Suppose now that A=(1,4,3,8,12,11,5) and we are looking for the value 5. Which numbers will be compared to 5? What happens? Why?

- Number 8, 4, 3
- Performed three comparisons
- Printed out “Sorry, 5 is not on the list”
- The list should have been sorted first before using Binary search

Question assigned to the following page: 2

Problem 2 Consider the selection sort algorithm we saw in lecture. Suppose A=(10,4,2,6,20,11). Execute selection sort on A and write down what the list looks like after each iteration of the outer loop. How many comparisons were performed before the algorithm terminated?

i = 1, A = (2,4,10,6,20,11) - 5 comparisons
i = 2, A = (2,4,10,6,20,11) - 4 comparisons
i = 3, A = (2,4,6,10,20,11) - 3 comparisons
i = 4, A = (2,4,6,10,20,11) - 2 comparisons
i = 5, A = (2,4,6,10,11,20) - 1 comparison

Total comparisons = 5 + 4 + 3 + 2 + 1 = 15 = n*(n-1)/2 where n = 6

Question assigned to the following page: 3

Problem 3 Consider the insertion sort algorithm we saw in lecture. Suppose A=(12,4,20,6,2,11). Execute insertion sort on A and write down what the list looks like after each iteration of the outer loop. How many comparisons were performed before the algorithm terminated?

j = 2, A = (4,12,20,6,2,11) - 1 comparison
j = 3, A = (4,12,20,6,2,11) - 1 comparison
j = 4, A = (4,6,12,20,2,11) - 3 comparisons
j = 5, A = (2,4,6,12,20,11) - 4 comparisons
j = 6, A = (2,4,6,11,12,20) - 3 comparisons

Total comparisons = 1 + 1 + 3 + 4 + 3 = 12

Question assigned to the following page: 4

Problem 4 In lecture we talked about the trade-off between using linear search on an unsorted list (the phone book) and sorting the list first and then using binary search. We sort the phone book because we search it enough times that it's worth the computational expense involved in sorting it. Suppose we have an unsorted list of length 50,000. How many worst case searches would be necessary for it to make it worth it to first sort the list with selection sort so that we could then use binary search instead of linear search.

- The cost of doing linear search alone should outweigh the cost of sorting plus the cost of doing binary search, in order to be worthwhile to do selection sort then binary search.
- The cost of sorting for the worst case scenario using selection sort is $\frac{n(n-1)}{2}$, where n is the number of items in the list. (this case $n = 50,000$)
- For a worst case scenario, the cost of sorting a linear search and a binary search is n and $\lfloor log_2 n \rfloor + 1$ respectively.

Let x be the "number of searches".

$$xn > x\left(\lfloor log_2 n \rfloor + 1\right) + \frac{n(n-1)}{2}$$

Solving for x:

$$x > \frac{1}{n-1-\lfloor log_2 n \rfloor} \cdot \frac{n(n-1)}{2}$$

$x > 25,000\ number\ of\ searches$ (at least, for worst case scenario)

Question assigned to the following page: 5

Problem 5 Design an algorithm that takes a list of integers as input A=(a1, a2, a3, a4, a5, . . . , an) along with a target value x. Your algorithm must locate a pair of integers in A that sum to the value x or report that there is no such pair. For example if A=(1,11,-2,12,8,9) and x=10, then your algorithm should report back either (1,9) or (-2,12). Write your algorithm out in pseudocode (bulleted english is fine).

PairSearch_sumx:(A,x)

```
n = A.length;
found = “no”;
i = 1;

while (found == “no” and i < n)      // remain true if can’t find the pair and i remains < n
{
        j = i + 1;                   // initializing, to compare item i to the next item j

        while (j <= n and found == “no”) // next item can go to n, remain true if still can’t
                                         // find the pair
        {
                if (A[i] + A[j] == x)
                {
                        print(“Pair “ + A[i] + “ and “ + A[j] + “ sums to “ + x);
                        found = “yes”; // set to “yes” so will exit both while loop
                                       // ignore other pair that can exist
                }
                j = j + 1; // increment until n if found still “no”
        }
        i = i + 1; // increment until n - 1 if found still “no” to keep comparing
}

if (found == “no”) // if no pair at all
{
        print(“No pair sums to ” + x);
}
```