Brandon Wong

CSCI104: Written Homework #1

1. Problem 1: Runtime Analysis
    a. Part A: f(n) = Θ(log(n))
        i. In the loop, the loop runs until ‘i’ is greater than or equal to n
        ii. In each iteration, the variable ‘i’ is squared, which makes the number of iterations equivalent to log(n)
        iii. Therefore, the runtime is Θ(log(n))

    b. Part B: f(n) = Θ(n)
        i. The outer loop runs until ‘i’ is greater than n, which runs for n times.
        ii. If ‘i’ is the multiple of a square root of n, an inner loop iterates until ‘k’ is greater than i^3, which runs for i^3 times.
        iii. The outer loop runs n times + the inner loop for i^3 times
        iv. Therefore, the runtime is Θ(n)

    c. Part C: f(n) = Θ(n^2 * log(n))
        i. The first loop runs until ‘i’ is greater than n, which runs for n times.
        ii. The second loop runs until ‘k’ is greater than n, which runs for n times as well.
        iii. If A[k] is equivalent to i, start another loop until ‘m’ is greater than n, where m doubles every iteration.
            1. Therefore, this loop iterates log(n) times
        iv. The first loop runs for n times, the second loop for n times, plus the final inner loop runs for log(n) times
        v. Therefore, the runtime is Θ(n^2 * log(n))

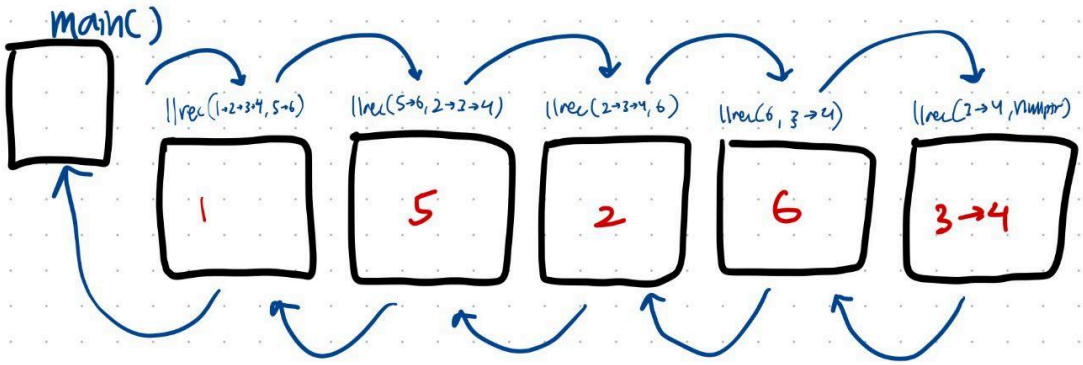
    d. Part D: f(n) = Θ(n)
        i. The first loop runs until ‘i’ is greater than or equal to n
        ii. If ‘i’ reaches the current array size, resize the array, which would take Θ(size) time.
        iii. Therefore, it would iterate n times for the loop plus ‘size’ times for the resizing; Θ(n + size), which is Θ(n)

2. Problem 2: Linked List Recursion Tracing:
    a. Question a: What linked list is returned if llrec is called with the input linked lists in1 = 1,2,3,4 and in2 = 5,6?

b. Question b: What linked list is return if llrec is called with the input linked lists in1 = nullptr and in2 = 2?

Homework 1:

Question a: in1 = 1 → 2 → 3 → 4 , in2 = 5 → 6



llrec (1 → 2 → 3 → 4 [in1], 5 → 6 [in2])

1 → in1→next : llrec (5 → 6 [in1], 2 → 3 → 4 [in2])

5 → in1→next : llrec (2 → 3 → 4 [in1], 6 [in2])

2 → in1→next : llrec (6 [in1], 3 → 4 [in2])

6 → in1→next : llrec (3 → 4 [in1], nullptr)

if in2 = null, return in1 (3 → 4)

∴ return
in1 = 1 → 5 → 2 → 6 → 3 → 4

Question b: in1 = nullptr , in2 = 2

Since if in1 == nullptr return in2,
then it will return 2 since in2 = 2.