Data Structures in Java

Shikhar Bakhda

ssb2189

# Question 1

Exercise 2.1

Following is the order of the functions from fastest to slowest growth rate:

1. 37

The functions and grow at the same rate:

Since 2 is a constant, the equality is true.

# Question 2

Exercise 2.6

Part a

If the fine on day N was given by F(N),

Part b

The number of days it will take will be of the order

# Question 3

Part a

|  |  |
| --- | --- |
| 1 | int sum = 0; |
| 1 + 24 + 23 | for ( int i = 0; i < 23; i ++) |
| 1 + (N + 1) + N | for ( int j = 0; j < n ; j ++) |
| 2 | sum = sum + 1; |

Total time

Part b

|  |  |
| --- | --- |
| 1 | int sum = 0; |
| 1 + N + 1 + N | for ( int i = 0; i < n ; i ++) |
| N + (1 + 2 + 3 + … + N) +1+(1 + 2 + 3 + … + N) | for ( int k = i ; k < n ; k ++) |
| 2 | sum = sum + 1; |

Total time

Part c

Since in every iteration, we call the function with n = n/k, only a fraction of n is sent each time, so we have:

# Question 4

Exercise 2.11

1. 3.374 ms

# Question 5

Exercise 2.15

Since the array is sorted, we can use binary search:

1. Check if the middle element A[mid] equals i. If yes, return True
2. If the i > A[mid], then the lower boundary is shifted to mid + 1
3. If the i < A[mid], then the upper boundary is shifted to mid – 1
4. Repeat the process till the lower boundary is greater than the upper boundary. Return False.

Every iteration, we divide the array into half. Thus the runtime is O(log N).