Lab 8

Graded

Group

Aharon Zingman
Anthony Johnson
Tyler Kennedy
...and 1 more
View or edit group

Total Points

1.5 / 3 pts

Question 1

P1 **0.5** / 1 pt

- **0 pts** Correct
- **0.25 pts** incorrect or missing worst case input description
- ✔ **0.25 pts** incorrect or missing summation used to calculate the number of steps for the worst case
- **0.25 pts** incorrect or missing explanation (or recursion tree) of how the code related to the summation
- ✔ **0.25 pts** incorrect or missing theta bound

Question 2

P2 **0.5** / 1 pt

- **0 pts** Correct
- **0.25 pts** incorrect or missing worst case input description
- ✔ **0.25 pts** incorrect or missing summation used to calculate the number of steps for the worst case
- **0.25 pts** incorrect or missing explanation of how the code related to the summation
- ✔ **0.25 pts** incorrect or missing theta bound

Question 3

P3 **0.5** / 1 pt

- **0 pts** Correct
- **0.25 pts** incorrect or missing worst case input description
- ✔ **0.25 pts** incorrect or missing summation used to calculate the number of steps for the worst case
- **0.25 pts** incorrect or missing explanation of how the code related to the summation
- ✔ **0.25 pts** incorrect or missing theta bound

CS 2230 Data Structures

Lab 8: Algorithmic Analysis

This lab should be done in teams of two or three people. Find a partner or two.

This is a written assignment. You can write your answers with your app of choice as long as you can produce a PDF version of it. Then submit the PDF on ICON to the Lab 8 assignment. This will take you to Gradescope. **Only one person per group should submit**. When submitting, **make sure to add your teammates**. You can check this short video on how to do that.

All of the questions in this lab will prepare you for completing the algorithmic analysis in Hw4.

For each of the problems below, provide

i. A description of a worst-case input (the input that would cause the largest run time).
ii. A summation used to calculate the number of steps for the worst case.
iii. An explanation of how the code relates to the summation:
   a) For loops, a discussion of the number of iterations and steps per iteration.
   b) For recursion, a recursion diagram as well, as seen in the lecture notes for this week.
iv. A Θ bound for the worst-case runtime $R(n)$.

Problem 1

Let n be the value of input x to foo.

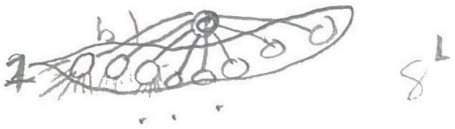
```
int foo(int x) {
  int sum = 0;
  if (x <= 1) return 0;
  for (int i = 0; i < 8; i++) {
    sum += foo(x/2);
  }
  return sum;
}
```

i: Any input of $x > 1$

ii: $\sum_{i=1}^{\log_2 n} \sum_{i=0}^{7} 1 = \sum_{i=1}^{\log_2 n} 1 = \log_2 n$

iii: a.) Summation starts at i=0, and goes to i<8, or i=7. Inside of the for loop, it is just the recursive call

b)



8^L

iv.) $\log_2 n = \Theta$

Problem 2

Let n be the size of the input list `ls` to `strange_sum`.

n = size

Assume that ls is an ArrayList. Also

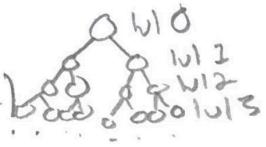
```
static int strange_sum(List<E> ls) {
  return strange_sum_helper(ls, 0, ls.size() - 1);
}

static int strange_sum_helper(List<E> ls, int left, int right) {
 1if (right - left < 0) return 0;
 1else if (right - left == 0) return ls.get(left);
 1else {
   1int mid = (left + right) / 2;
     return strange_sum_helper(ls, left, mid)
            + strange_sum_helper(ls, mid + 1, right);
   }
}
```

i.) list size > 1

ii.) $\sum_{i=0}^{\log_2 n} \sum_{i=1}^{2} 1 = \sum_{i=0}^{\log_2 n} 1 = \log_2 n$

iii.)



iv.) $\Theta = \log_2 n$

Problem 3

Redo the previous problem but now assume that ls is a LinkedList.

i.) list size > 1 from get function of linkedlist

ii.) $\sum_{i=0}^{\log_2 n} \sum_{i=1}^{2} n = n \log_2 n$

iii.)

iv.) $\Theta = n \log_2 n$