Homework 2

Due 09/17/2024

September 12, 2024

Analyze the worst-case time complexity of the algorithms below. You may express their complexity using Big-Oh or Big-Theta. If you use Big-Oh, you should give a *tight* upper bound on the complexity (the smallest possible valid bound). For example, if an algorithm has a complexity of $\Theta(n^3)$, then $O(n^3)$ is a tight upper bound, but $O(n^4)$ is not.

Show all work.

1.

```
   Input: n: positive integer
   Input: k: positive integer
1  Algorithm: LoopMystery1
2  ret = 0
3  for i = 1 to n do
4  |  for j = i to n step k do
5  |  |  for ℓ = 0 to k do
6  |  |  |  ret = ret + i(j + ℓ)
7  |  |  end
8  |  end
9  end
10 return mystery
```

2.

```
   Input: n: positive integer
1  Algorithm: LoopMystery2
2  ret = 0
3  i = 1
4  max = n · n · n
5  while i ≤ max do
6  |  ret = ret + i
7  |  i = 2i
8  end
9  return ret
```

3. Answer the following questions about the worst-case complexity of the recursive algorithm below.

```
   Input: n: positive integer
 1 Algorithm: RecursionMystery
 2 if n = 1 then
 3 |  return 1
 4 else
 5 |  ret = 0
 6 |  for i = 1 to n − 1 do
 7 |  |  ret = ret + RecursionMystery(i)
 8 |  end
 9 |  return ret
10 end
```

(a) What is the *nonrecursive* complexity of a single call to RecursionMystery with an input of size k?

(b) Sketch a recursion tree for $n = 4$. How many recursive calls are there of size 1, 2, 3, and 4, respectively?

(c) Sketch a recursion tree for $n = 5$. How many recursive calls are there of size 1, 2, 3, 4, and 5, respectively?

(d) Write a summation that approximates the time complexity for RecursionMystery(n). You do not need to simplify this summation.