

F18 CS20A Assignment 5

STEWART DULANEY

TOTAL POINTS

23 / 23

QUESTION 1

6 pts

1.1 1 / 1

✓ - 0 pts Correct

- 0.5 pts Click here to replace this description.

- 1 pts Incorrect

1.2 1 / 1

✓ - 0 pts Correct

- 0.5 pts Click here to replace this description.

- 1 pts Click here to replace this description.

1.3 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

1.4 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

1.5 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

1.6 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

QUESTION 2

8 pts

2.1 2 / 2

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

- 2 pts Click here to replace this description.

2.2 2 / 2

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

- 2 pts Click here to replace this description.

2.3 2 / 2

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

- 2 pts Click here to replace this description.

2.4 2 / 2

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

- 2 pts Click here to replace this description.

QUESTION 3

4 pts

3.1 All a-d 4 / 4

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

- 2 pts Click here to replace this description.

- 4 pts Click here to replace this description.

QUESTION 4

2 pts

4.1 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

4.2 1 / 1

✓ - 0 pts Correct

- 0.5 pts Click here to replace this description.

- 1 pts Click here to replace this description.

QUESTION 5

3 pts

5.1 1 / 1

✓ - 0 pts Correct

- 1 pts Click here to replace this description.

5.2 1 / 1

✓ - 0 pts Correct

- 1 pts Incorrect

5.3 1 / 1

✓ - **0 pts** Correct

- **1 pts** Click here to replace this description.

Name: Stewart Dulaney

ID: 1545566

Problem 1:

For all of the following, determine the **total operation count** and then the **Big-O** of the given code segments:

a.

```
for (int j = 0; j < n; j++)
    for (int k = 0; k < j; k++)
        sum++;
```

$$\begin{aligned} f(n) &= 1 + n + n + n + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} + \frac{n(n-1)}{2} \\ &= \frac{3n(n-1)}{2} + 3n + 1 \\ &= \frac{3}{2}n^2 - \frac{3}{2}n + 3n + 1 = \boxed{\frac{3}{2}n^2 + \frac{3}{2}n + 1} \end{aligned}$$

$$\boxed{O(n^2)}$$

b.

```
for (int i = 0; i < q*q; i++)
    for (int j = 0; j < i; j++)
        sum++;
```

$$\begin{aligned} f(n) &= 1 + q^2 + q^2 + q^2 + \frac{q^2(q^2-1)}{2} + \frac{q^2(q^2-1)}{2} + \frac{q^2(q^2-1)}{2} \\ &= \frac{3q^2(q^2-1)}{2} + 3q^2 + 1 = \frac{3}{2}q^4 - \frac{3}{2}q^2 + 3q^2 + 1 = \boxed{\frac{3}{2}q^4 + \frac{3}{2}q^2 + 1} \end{aligned}$$

$$\boxed{O(q^4)}$$

For all of the following, just determine the **Big-O** of the given code segments:

c.

```
for (int i = 0; i < n; i++) the 1st loop runs  $\textcircled{n}$  times (fixed)
    for (int j = 0; j < i*i; j++) every time the 1st loop runs once, 2nd loop runs
        for (int k = 0; k < j; k++) roughly  $\textcircled{n^2}$  times (max. value of i is n-1,
            sum++; rounded up to n)
```

$$n \cdot n^2 \cdot n^2$$

$$\boxed{O(n^5)}$$

every time the 2nd loop runs once, the 3rd loop runs roughly $\textcircled{n^2}$ times (max. value of j is $\sim n^2-1$, rounded up to n^2)

Name: Stewart Dulaney

ID: 1543566

d.

```
for (int i = 0; i < p; i++) (P) (fixed)
    for (int j = 0; j < i*i; j++) i_max = p-1, rounded up to p → (P^2)
        for (int k = 0; k < i; k++) (P)
            sum++;
```

$P \cdot P^2 \cdot P$

$O(P^4)$

e.

```
for (int i = 0; i < n; i++) (n) (fixed)
{
    Circ arr[n]; (n) (if calling Circ constructor qualifies as an operation,
    arr[i].setRadius(i); this is done n times. sidenote: n must be a constant)
}
```

$n \cdot n$

$O(n^2)$

f.

```
for (int i = 0; i < n; i++) (n)
{
    int k = i;
    while (k > 1) (log n)
    {
        sum++;
        k = k / 2;
    }
}
```

$O(n \log n)$

Name: Stewart Dulaney

ID: 1545566

Problem 2:

Given a vector of sets of ints, `vector< set<int> > v`, assume the vector `v` has N total sets and that each set has an average of Q items.

- a. What is the Big-O of determining if the first set, `v[0]`, contains the value 7?

$$1 + \log Q$$

$$O(\log Q)$$

- b. What is the Big-O of determining if any set in `v` has the value 7?

$$N \cdot \log Q$$

$$O(N \cdot \log Q)$$

- c. What is the Big-O of determining the number of even values in all of `v`?

$$N \cdot Q$$

$$O(N \cdot Q)$$

- d. What is the Big-O of finding the first set with a value of 7 and then counting the number of even values in that set?

$$N \cdot \log Q + Q$$

$$O(N \cdot \log Q + Q)$$

Name: Stewart Dulaney

ID: 1545566

Problem 3:

Determine the data structure needed if we wanted to maintain a bunch of peoples' names and for each person, allows us to easily get all of the streets they lived on. Assume there are P total people and each person has lived on average E former streets.

`map < string, set < string >> m;`

What is the Big-O cost of:

- a. Finding the names of all people who lived on "Levering Street"?

$$P \cdot \log E$$

$$O(P \cdot \log E)$$

- b. Determining if "Bill" ever lived on "Westwood Blvd"?

$$\log P + \log E$$

$$O(\log P + \log E)$$

- c. Printing out every name along with each person's street addresses in alphabetical order?

$$P \cdot E$$

$$O(P \cdot E)$$

? ~~X~~ how to account for this



don't need to add time

complexity for sorting

b/c maps and sets

are both sorted

- d. Printing out all the streets that "Tala" has lived on?

$$\log P + E$$

$$O(\log P + E)$$

Name: Stewart Dulaney

ID: 1545566

Problem 4:

Fibonacci numbers are a sequence of numbers given by the relationship:

$$F_n = F_{n-1} + F_{n-2}$$

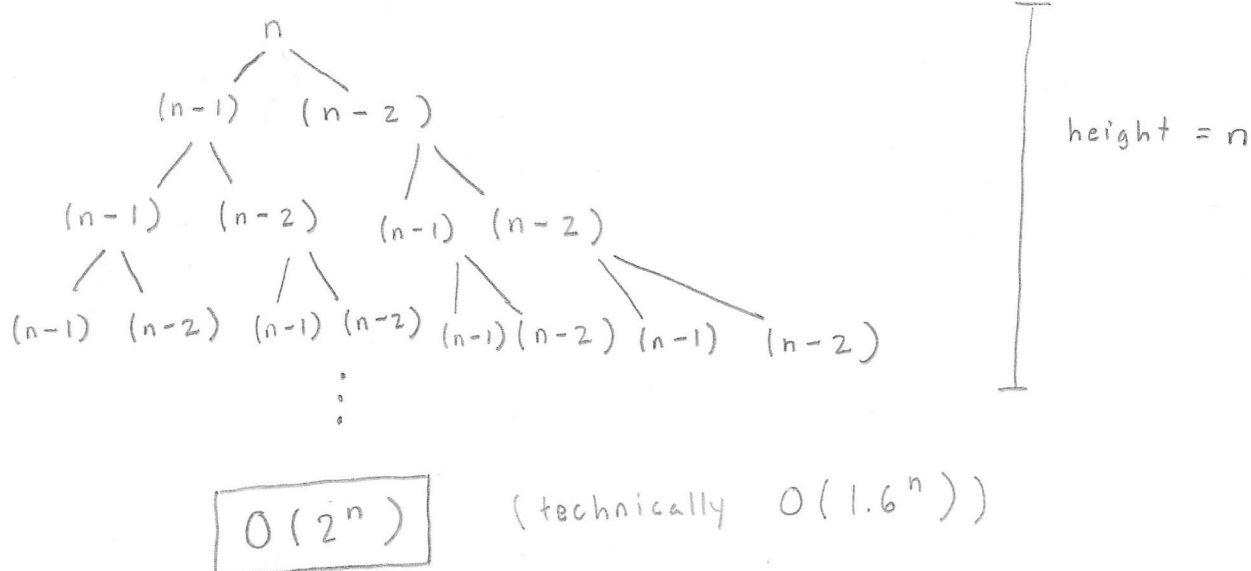
With $F_0 = 0$ and $F_1 = 1$. In other words, the n th Fibonacci number is given by the sum of the two Fibonacci numbers before it. For Example, the first 13 Fibonacci numbers are:

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144

a. Implement a recursive function to compute the n th Fibonacci number:

```
int fibonacci(int n) { // Precondition: n >= 0
    if ( n <= 1 ) {
        return n;
    }
    return ( fibonacci(n-1) + fibonacci(n-2) );
}
```

b. What is the Big-O of the recursive Fibonacci function?



Name: Stewart Dulaney

ID: 1545566

Problem 5:

Given the following array show the result after one round of the each of the sorting algorithms indicated. One round being one full iteration of the algorithm's outer most for/while loop.

a. Selection Sort:

| | | | | | | | | |
|----|----|---|----|----|----|----|----|---|
| 99 | 16 | 3 | 19 | 13 | 0 | 13 | 12 | 6 |
| 0 | 16 | 3 | 19 | 13 | 99 | 13 | 12 | 6 |

b. Insertion Sort:

| | | | | | | | | |
|----|----|---|----|----|---|----|----|---|
| 99 | 16 | 3 | 19 | 13 | 0 | 13 | 12 | 6 |
| 16 | 99 | 3 | 19 | 13 | 0 | 13 | 12 | 6 |

c. Bubble Sort

| | | | | | | | | |
|----|----|----|----|----|----|----|----|----|
| 99 | 16 | 3 | 19 | 13 | 0 | 13 | 12 | 6 |
| 16 | 3 | 19 | 13 | 0 | 13 | 12 | 6 | 99 |