Grayson Buchholz
Professor Taylor
CSCI 1112
15 November 2019

Homework 6

2.1 Calculate the Big-O run time of these loops

2.1.1

```
i=0;
While i < n {
        For(j=0; j<n; j++) {
                Print(this is a loop)
        }
        i = i*2;
}
```

O(logn)

2.1.2

```
Array x = new array[n]
Count = 0;
For(i=0; i<n; i++) {
        Array[Count] = i;
        Count ++;
        If( i%3 = 0 ) {
                While (Count != 0) {
                        Count --;
                        Array[Count] = 0;
                }
        }
}
```

O(2n)

2.2 Algorithm run-time (Big-O) questions

2.2.1
What is the run-time of bubble sort? Explain why in detail.

In the best case, where the array is already sorted in ascending order, the run-time is O(n). This is because no swap occurred in 1 iteration of n elements.
In the worst case, where the array is already sorted but in descending order, the run-time is $O(n^2)$. The first iteration looks at n elements, and the next iteration would look at n-1 elements and so on until only a single comparison occurs.

2.2.2
What is the run-time of running binary search on a sorted array? Explain why in detail.

In the best case, the run-time is O(1), where the item we search for is found in the first iteration.

In the worst case, the run-time is O(logn). This is because after each iteration, the number of n items to look through is halved.

2.2.3
What is the worst-case run-time of QuickSort? Why? What is the average case run-time of Quicksort? Why?

Worst case: $O(n^2)$, when the partitions are completely unbalanced, there is a recursive call on each element.

Average case: O(nlogn), occurs when partitions are evenly balanced.

2.3 Fibonacci Sequence

2.3.1
Find the Big-O run-time of this program

```
Public int Fibb (int x) {
        If(x==1) return 1;
        If(x==0) return 0;
        Return Fibb(x-1)+Fibb(x-2);
}
```

$O(2^n)$

2.3.2
Rewrite this program so that it has a much smaller run-time.
Hint: use an array to store information. O(n) is the optimal solution

```
Public int Fibb(int x) {
        Int secPrevious;
        Int Previous = 0;
        Int current = 1;
        For(int i=1; i<x; i++) {
                secPrevious = Previous;
                Previous = current;
                current = secPrevious + Previous;
        }
        Return current;
}
```

2.3.3
Explain the run-time of your program.

There is only one loop in the program that iterates over n elements, so the run time is O(n).