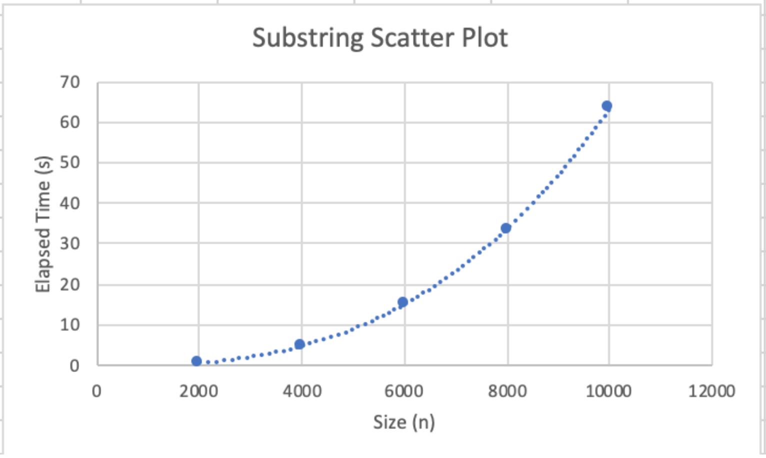
PDF For Project 3

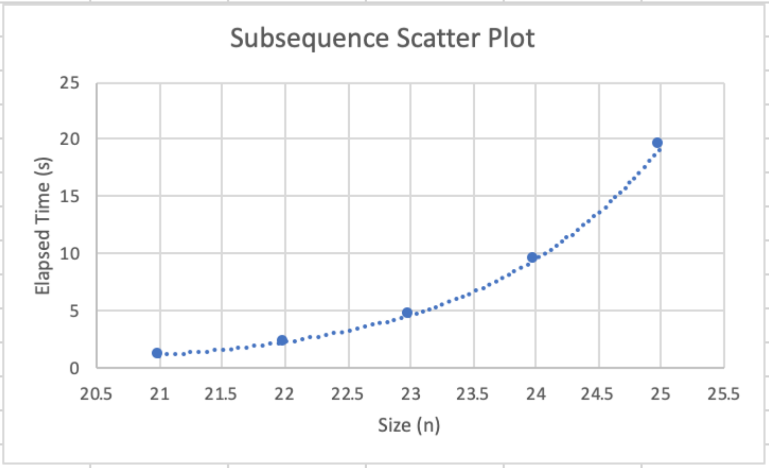
Name: Jesus Flores
Email: jflores2016@csu.fullerton.edu

Two Scatter plots:

String a size	Time	String b size
2000	0.690913	2300
4000	4.87298	4300
6000	14.9676	6300
8000	33.5026	8300
10000	63.7597	1300



Short s size	Time (s)	Long s size
21	1.13585	30
22	2.27703	30
23	4.66915	30
24	9.48023	30
25	19.5394	30



Pseudocode for subsequence detection:

```
detect_subsequence(string cand_sub, string cand_superseq)
    i = 0
    k = 0
    while i < cand_superseq.length and k< cand_sub.length
        if cand_sub[k] == cand_superseq[i]
            k++
        i++
    if k == cand_sub.length
        return true
    return false
```

<u>Time complexity for subsequence detection:</u>
1 + 1 + n (1+1+1 +1) + 1 +1 = 4n+4
I can prove that 4n+4 is an element of O(n) by using properties of O:
4n+4 is an element of O(4n+4) (trivial)
O(4n+4) = O(4n) (dropping additive constants)
O(3n) = O(n) (dropping multiplicative constants)

Yes, there is a noticeable difference between the two algorithms. The subsequence algorithm is much faster than the subset algorithm. At first it did surprise me since the subsequence algorithm was running at a much faster pace than subset, and that was with subsequences input's strings being much larger than the input strings of the subset algorithm. However, I remembered that I was creating all of the subsets of an n element set in the subset algorithm, which takes O(2^n) time to create, so I was no longer surprised.

Yes, my empirical analysis is consistent with my mathematical analysis. The time complexity that I traced through, proved and double checked with the time complexity that you said both functions should run at matched the best fit lines in my scatter plots.

Yes, the evidence is consistent with hypothesis 1. The exhaustive search algorithms were easy to implement, and after testing it with the make file, also produces the correct outputs.

Yes, the evidence is consistent with hypothesis 2. The subset algorithm was extremely slow. In the time that it took the subsequence algorithm to process 4,000 letters for string a, and 4,300 letters for string b, it took subset algorithm to process only 23 letters for one string and 30 letters for another string. This is of course due to the fact that you are generating all subsets of the shorter string from the subset function, which would take 2^n time. Apart from it being extremely time consuming, it also consumed a lot of space. When I was timing this function, I realized that if the length of the shorter string was greater than 25, it would give me an error called “std:: bad_alloc”, which after a quick google search I discovered that it meant that there was a failure to allocate storage.