Ralph Paul

Exercise 2

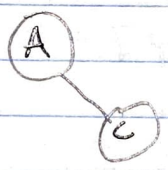
1. When looking at these two functions if we plug 2 and 3 in both we see $f(2) = 4$ $g(2) = 16$, $f(3) = 9$ $g(3) = 0.00015$ So because depending on odd or even the bigger value changes so either can't be big O of another

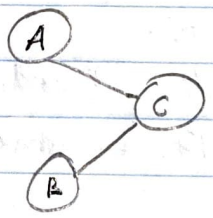
2. The Big-O would be n^2 because the for loop is Herating through $n \cdot n$ times and with the while loop it iterates $n^2 + n(\frac{n}{4})^2 + (\frac{n}{4^2})^2 + (\frac{n}{4^3})^2 + \ldots$ which sums up to $O(n^2)$ Since we ignore constants

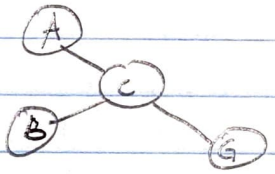
3. Since we need to partition both sides to do the sorting, so the sorting would take $n \log n$ to do the merging we are doing n comparisons So the total "work done" is $n(n \log n)$

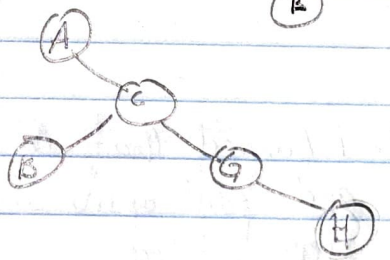
So $W(n) = n^2 \log n$

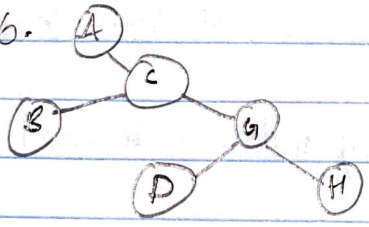
4. 1. 

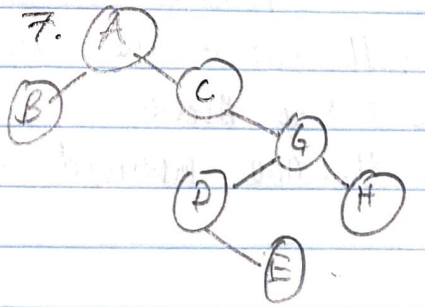
2. 

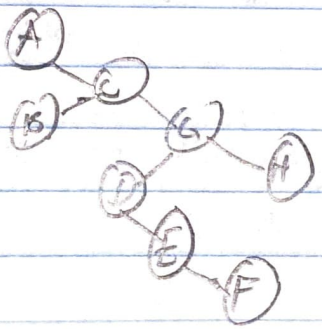
3. 

4. 

5. 

6. 

7. 

8. 

5. Algorithm: quickSortIter (data, L, R)

Input: data – an array of size n with a subrang L and R

```
1.  int stack [R - L + 1];
2.  int top = -1;
3.  stack [++top] = L
4.  stack [++top] = R
5.  while (top >= 0)
6.      R = stack [top--]
7.      L = stack [top--]
8.      int part = partition (A, L, R)
9.      if (p-1 > low)
10.         stack [++top] = L
11.         stack [++top] = part-1
12.     endif
13.     if [p+1 < R)
14.         stack [++top] = p+1
15.         stack [++top] = R
16.     endif
17. endwhile
```

6.

1. I learned that in college he studied Classics and philosophy for undergrad. Not even anything stem related
2. He was in the royal navy and he learned Russian there
3. He also invented the null reference

7. Algorithm: findSection: (A, i, j, B)
Input: arrays A and B, indices i and j

```
1. for k = 0 downto B.length
2.     if A[i] == B[k]
3.         for n = 1 down to j - i
4.             if A[i+n] != B[k+n]
5.                 count = 0; break;
6.             endif
7.             count++;
8.         end for
9.     end if
10. end for
11. if count = j - i
12.     return true;
13. else
14.     return false;
15. end if
```

Algorithm: findLargestSection (A, B) Input arrays A and B

```
1. int len = A.length
   while len > 0
      for i = 0 to i <= B.length - len
         if (findSection (A, i, i+len-1, B))
            return (i, i+len-1)
         end if
      end for
      len--
   endwhile
```

The Big Oh of findLargestSection would be $O(n^4)$ this is because in our inner loop we are iterating $n^2/2$ times and we are iterating $n^2/2$ in our outter loop as well so it simplifies to $n^4/4$ and would be $O(n^4)$