Assignment 1

Group#33
Fredrik Ullman gusullmfr@student.gu.se
Hassan Mualla gusmuaha@student.gu.se

Dynamic arrays

1. The big O complexity of reverse() is O(logN)
2. The big O complexity of remove() is O(N)
   The big O complexity of remove2() is O(1)
3. The big O complexity of find() is O(N)
4. The big O complexity of maxPalindrome() is O(N²) n^2

Question 5.

a. $\sum_{i=1}^{n} i^2 = 1^2 + 2^2 + 3^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$
   $\rightarrow 2n^3 + n^2 + 2n^2 + n$
   *removing constants
   most dominant term : $O(n^3)$

b. $\sum_{i=1}^{logn} 2^i = 2^1 + 2^2 + 2^3 + \ldots + 2^{logn}$
   $\rightarrow (2^{(logn+1)}) - 2$
   $\rightarrow 2*(2^{(logn)})-2$

   *removing constants
   As $2^{(logn)} = n$, the most dominant term : O (n)

c. $2x^2 + 16x^3$
   *remove constants
   most dominant term : $O(n^3)$

d. $(x-logx^3)(x-2\sqrt{x})+4xlogx^2$
   $\rightarrow x^2 - 2x\sqrt{x} * (-xlogx^3)*(-2\sqrt{x})+4xlogx^2$
   x^2 is the term with the highest complexity degree,
   Hence, the most dominant term : $O(n^2)$

Question 6 : 4^n ∉ 2^n+1

4^n = (2*2)^n = 2²n
T(n) = C * f(n)
We want to show that 4^n ≤ C*2^n is not true.
2^2n ≤ C * 2^n
→ $\frac{2\hat{}2n}{2\hat{}n}$ ≤ C
→2^2n-n ≤ C
→2^n ≤ C
We have proven that C ≥ 2^n, we know that C has to be a constant and 4^n ∉ 2^n is therefore not true.

Question 7 :

```
int calculatePolynomialSum ( int x, int[] a) {
        int sum = a[a.length -1] * x;
        for (int i = a.length - 2, i > 0, i--) {
                sum= x*( sum + a[i] );
        }
        sum += a[0];
        return sum;
}
```