Homework 3

Nick Alvarez

Q1

a) This algorithm determines if the elements of an array are unique. If not, it returns "false".

b)
```
for i ← 0 to n-2 do                All elements - 1
    for j ← i+1 to n-1 do          All elements - 1
        if A[i] == A[j]            (n-1)^2 → n^2 - 2n + 1
            return false
```

Worst case: Entire array traversed

Average case: Some of the array is searched

So we can conclude the running time is $O(n^2)$, due to the nested "for" loops.

Q2

a) See problem2.cpp

b) The answer will still be 9. As the left side is evaluated first, the first "9" will be the max the computer reports, even though there is a "9" on the right and middle. The > operator ensures this as leftMax is compared to rightMax, instead of using >=.

c) Comparisons$(n) = T(n)$

$T(n) = 2T(n/2) + 2$ when $n \geq 2$

$T(1) = 0$

$$T(n) = 2T(n/2) + 1$$

$\hookrightarrow 2[2T(n/4) + 1] + 1$

$\hookrightarrow 2[2[2T(n/8) + 1] + 1] + 1$

$= 2^3 T(n/8) + 4 + 2 + 1$

$\vdots$

$\hookrightarrow 2^i \underbrace{T(n/2^i)}_{\hookrightarrow T(1) = 0} + 2^{i-1} + 2^{i-2} + \cdots + 2 + 1$

$\hookrightarrow 2^{i-1} + 2^{i-2} + \cdots + 2 + 1$

$\underline{\dfrac{2^i - 1}{2 - 1} = 2^i - 1 \rightarrow n - 1}$

We can equate this to Case 1 of Master's method and find that $T(n) = \underline{\Theta(\log n)}$

Q3

a) See problem3.cpp