Brendan Morgenstern

CUS 715

Problem Set 3

1

```
int k = 0;
int i = n;
while (i >= 1) {
   int j = i;
   while (j <= n) {
      k++;
      j = 2 * j;
   }
   i = floor( i/2.0)
}
```

This function performs operations equal to the sum of $\log_2 n$ or $\sum_{k=0}^{\log 2n} k$ therefore T(n) = $\frac{logn*(logn+1)}{2}$

2. If n = 5 and A = { 2,5,3,7,8 }, the second function will return 57 because it returns the sum of A's elements + 2^n which in this case is 25+32. The T(n) for the function is $2n$ because n iterations are performed twice. This can be improved by performing all necessary tasks in one iteration as opposed to two separate iterations like so : This way the T(n) becomes simply n.

```
int add_them (int n, int[] A)
{
   int i,j,k;
   k = 1;
   for (i = 0; i < n; i++) {
      j = j + A[i];
      k = k + k;
   }
   return j + k;
}
```

3. To belong to the class O(f(n)), f(n) must satisfy this inequality

$f(n) = 3n^3 + n^2 \leq cn^3$

If all the terms are raised to the highest power and combined, we get

$f(n) = 3n^3 + n^2 \leq 4n^3$

Therefore c = 4 and f(n) belongs to O(f(n))

To belong to the class Ω(f(n)), f(n) must satisfy this inequality

$f(n) = 3n^3 + n^2 \geq cn^3$

In f(n) the n^3 term is being multiplied by 3, therefore this condition can be satisfied if c $\leq$ 3; let's assume 1.

$f(n) = 3n^3 + n^2 \geq n^3$

This inequality holds therefore f(n) belongs to Ω(f(n))

Because f(n) belongs to Ω(f(n)) as well as O(f(n)), it also belongs to θ(f(n)).