Aidan O'Cara - CSCI 104 HW#1

Part A

```
void f1(int n)
{
  int t = sqrt(n)
  for(int i=0; i<n; i++){
    for(int j=0; j<n; j++){
      // Do something O(1)
    }
    n -= t;
  }
}
```

Outer loop: $\sum_{0}^{n}$ Inner loop $+1$

Inner loop: $\sum_{0}^{n} 1$

Full Equation: $1+\sum_{1}^{n}(1+\sum_{1}^{n}1)$

$\rightarrow 1+\sum_{1}^{n}(1+n)$

$\rightarrow 1+n+n^2$

These n's will be discounted by the subtraction of $\sqrt{n}$. I'm not sure how to express that.

The n^2 will dominate, meaning this is naively $O(n^2)$ But the fact that n shrinks as the loop runs reduces runtime significantly.

Runtime Plot

n	Outer Loops	Inner Loops	Total Loops
1	1	1	1
2	1	2	2
4	2	3	6
9	3	6	18
16	4	10	40
25	5	15	75

← This case by case analysis shows that runtime complexity is less than $O(n^2)$, closer to $O(n \log n)$ for some small values of n.

Part B

```
for(int i=1; i<=n; i++){          1st Loop: n
    for(int k=1; k<=n; k++){      2nd Loop: → Σ_1^n Σ_1^n
        if(A[k]==i){ ← Could be triggered 0 times, could be n times
            for(int m=1; m<=n; m+m){ 3rd loop happens log_2 n times
                //Something O(1)
            }
        }
    }
}
```

Although no operation happens within the loop, incrementing the loop itself and the if uses a constant runtime

Best Case, if statement is true 0 times: $\sum_1^n \sum_1^n 1$

$\sum_1^n n \rightarrow \boxed{\Omega(n^2)}$ Best Case, Lower Bound

Worst Case, Upper Bound, if statement is always true:

$\sum_1^n \sum_1^n \log_2 n \rightarrow \sum_i^n n \log n \rightarrow \boxed{O(n^2 \log n)}$ Worst Case, Upper Bound

Part C = Recursive

```
void f3(int *A, int n)
{
  if (n <= 1) return;  → 1
  else {
    f3(A, n-2);  → O(f3)
    // Something O(1) → 1
    f3(A, n-2);  → O(f3)
  }
}
```

Each call of f3 is either 1 or 2+2(f3). Because n is decremented in each call, and each call happens twice, the total runtime will be dominated by 2^n.

A single call of f3 has runtime
1 if $n <= 1$ or
$2 + 2(O(f3))$ if $n > 1$

Overall Runtime $\leq \sum_{1}^{2^n} 2 = 2(2^n)$

Part D

n	Runtime
$<= 10$	$n+3$
$10 < n <= 40$	$n+3+14$
$40 < n <= 160$	$n+3+14+44$
$160 < n <= 640$	$n+3+14+44+164$

```
int f(int n){
  int *a = new int[10];  → 1
  int size = 10;  → 1
  for (int i=0; i<n; i++) {  → Σ_1^n 1 + if( Σ_1^size 1 )
    if (i = size){
      int newsize = 4*size  → 1
      int *b = new int[newsize]  → 1
      for(int j=0; j<size; j++){b[j]=a[j]}  Loop happens Σ_0^size 1 times
      delete [] a  → 1
      a = b  → 1
      size = newsize}
    a[i] = i*i;}  → 1
}
```

Overall Runtime: $O(n + kn)$