1st ... runs to ... times

```
void f1 (int n) {
  int t = sqrt(n);
  for (int i = 0; i < n; i++) {        Θ(n)
    for (int j = 0; j < n; j++) {      Θ(n)
      // Θ1
    }
    n -= t
  }
}
```

1) first j loop runs for $[0, n)$

n decrease $\sqrt{n}$ $[\Theta(1)]$

second j loop runs for $[0, n)$

1st: it = i = 0, j = 0-16, n = 16, t = 4

n = 12

2nd: it = i = 1, j = 0-12

n = 8

3rd: it = i = 2, j = 0-8

n = 4

4th: it = i = 3, j = 0-4

n = 0

$i = k = n - k\sqrt{n}$

$k = n - k - \sqrt{n}$

$k + k - \sqrt{n} = n$

$k(\sqrt{n} + 1) = n$

$k = \frac{n}{1 + \sqrt{n}}$

$$\sum_{k=0}^{\frac{n}{1+\sqrt{n}}} \sum_{j=0}^{n-k\sqrt{n}} \Theta(1)$$

$T(0) = \Theta(1)$

$T(1) = \Theta(1)$

$$\Rightarrow \sum_{n=0}^{\frac{n}{1+\sqrt{n}}} n - k\sqrt{n} \Rightarrow n - k\sqrt{n}\left(\frac{n}{1+\sqrt{n}}\right)$$

$$\Rightarrow \frac{n^2 - k\sqrt{n}(n)}{1+\sqrt{n}} \Rightarrow \frac{n^2}{1+\sqrt{n}} - \frac{kn^{3/2}}{1+\sqrt{n}}$$

$$\Rightarrow n^{3/2} - n = \boxed{\Theta(n^{3/2})}$$

smaller

b)
```
void f2 (int* A, int n)
{ for (int i=1; i<=n; i++) {          Θ(n)
    for (int k=1; k<=n; k++) {        Θ(n)
      if (A[k]==i) {                  Θ(1)
        for (int m=1; m<=n; m=m+m) {  Θ(log(n))
        }
      }
    }
  }
}
```

first 2 for loops run (0,n) $\Theta(n)$
if statement $\Theta(1)$
3rd for loop $\Theta(\log n)$

In most optimal worst case scenario, array A $1 \le A[k] \le n$, will be equal to i at a point in time. (Ex all values are 1)
Function enters n times, NOT n^2

$$\sum_{i=1}^{n} \sum_{k=1}^{n} \underbrace{(\Theta(1))}_{\text{if statement}} +$$

$$\sum_{k=1}^{n} \left(0 \sum_{n=1}^{\log n} \Theta(1)\right)$$

$$\Theta(n^2) + \sum_{k=1}^{n} (\log n)$$

$$\Theta(n^2) + \Theta(n \log n)$$

$$= \Theta(n^2)$$

c)
```
void f3(int* A, int n)
{ if(n<=1) return;     Θ(1)
    else {
      f3(A, n-2);      T(n-2)
      f3(A, n-2);      T(n-2)
    }
}
```

$T(0) = \Theta(1)$
$T(1) = \Theta(1)$

$k=1$ $\quad T(n) = \Theta(1) + T(n-2) + T(n-2)$

$k=2$ $\quad \Theta(1) + \Theta(1) + \Theta(1) + T(n-4) + T(n-4) + T(n-4) + T(n-4)$

$= \Theta(3) + 4T(n-4)$

$k=3$ $\quad \Theta(7) + T(n-6) + T(n-6) + \dots$

$k+n$ $\quad \Theta(2^k - 1) + 2^k T(n-2k)$

$n - 2k = 0 \Rightarrow k = \frac{n}{2}$

$(2^{n/2} - 1) + T(0)$

$2^{n/2} \Rightarrow \boxed{\Theta(2^n)}$

d)

```
int f (int n) {
   int *a = new int [10];
   int size = 10;
   for (int i=0; i<n; i++) Θ(n) {
     if (i == size) {
        int newSize = 4* size;
        int *b = new int [newsize];
        for (int j=0; j<size; j++) b[j] = a[j]; Θ(newsize)
        delete []a; Θ(log n)
        a = b;
        size = newsize;
     }
   a[i] = i*i;
}
```

$\Theta(\text{size}) = \Theta(4^k)$ value of new size & size is power 4

$$T(n) = \sum_{k=0}^{\log_4 n} \left(\Theta(4^k) + (n - \log_4 n)\right) = \Theta(4^{\log_4 4^n}) = \Theta(n)$$