a)

```
void f1(int n)
{
    int i=2;
    while(i < n){
        /* do something that takes O(1) time */
        i = i*i;
    }
}
```

- After 1st iteration of while loop: $i = 4 = 2^2$

  After 2nd iteration of while loop: $i = 16 = 2^4$

  After 3rd iteration of while loop: $i = 256 = 2^8$

  - $T(n) = \underbrace{\Theta(1) + \Theta(1) + \Theta(1) + \cdots}_{K \text{ times}}$

    where $n = 2^{2^K}$

    $T(n) = K \cdot \Theta(1)$

  - Solve for K: $n = 2^{2^K}$

    $\log(n) = \log(2^{2^K})$

    $\log n = 2^K$

    $\log(\log n) = \log(2^K)$

    $\log\log n = K$

  - $T(n) = \log(\log(n))\,\Theta(1)$

    $T(n) = \Theta(\log(\log n))$

b)

```
void f2(int n)
{
    for(int i=1; i <= n; i++){
        if( (i % (int)sqrt(n)) == 0){
            for(int k=0; k < pow(i,3); k++) {
              /* do something that takes O(1) time */
            }
        }
    }
}
```

first for loop: n^{th} iterations,

however, the succeding if statement limits the amount of operations that are run, out of the n^{th} iterations.

- Looking at a few examples:

$n = 9 \Rightarrow$ if statement evalues to true when $i \% 3 == 0$

$\therefore i = 3, 6, 9$ (3 iterations)

$n = 16 \Rightarrow$ if statement evaluates to true when $i \% 4 == 0$

$\therefore i = 4, 8, 12, 16$ (4 iterations)

Conclusion: The first for loop + if statement combined will run $\sqrt{n}$ times.

- $T(n) = \sum_{i=1}^{i=\sqrt{n}} \left(\sum_{i=1}^{?} \theta(1) \right)$

↑ inner for loop

- Continuing examples

$n = 9 \Rightarrow i = 3, 6, 9$

$T(n) = \left(3^3 + 3^{2^3} + 3^{3^3}\right) \theta(1)$

$n = 16 \Rightarrow i = 4, 8, 12, 16$

$T(n) = \left(4^3 + 4^{2^3} + 4^{3^3} + 4^{4^3}\right) \theta(1)$

- $T(n) = \sum_{i=1}^{\sqrt{n}} (n^3) \theta(1)$

$= \sum_{i=1}^{\sqrt{n}} \theta(n^3)$

$= \theta(n^3 \sqrt{n}) = \theta(n^{3.5})$

c)

Part (c)

```
for(int i=1; i <= n; i++){
  for(int k=1; k <= n; k++){
    if( A[k] == i){
      for(int m=1; m <= n; m=m+m){
        // do something that takes O(1) time
        // Assume the contents of the A[] array are not changed
      }
    }
  }
}
```

$$T(n) = \sum_{i=1}^{n} \sum_{k=1}^{n} \left(\theta(1) + O\left(\sum_{m=1}^{\log(n)} \theta(1) \right) \right)$$

if condition:

Worst case scenerio – all elements are same value of i

best case scenerio – no elements are i

Looking at only worst case scenerio:

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

↑ assume currently, i = the element in A

$$= \theta(n^2) + \theta(n \log n)$$

$$T(n) = \theta(n^2)$$

d)

```
int f (int n)
{
  int *a = new int [10];        ← Θ(10)
  int size = 10;                ← Θ(1)
  for (int i = 0; i < n; i ++)  ← Θ(n)
     {
        if (i == size)          ← only one n will evaluate to true
          {
             int newsize = 3*size/2;          ← Θ(1)
             int *b = new int [newsize];      ← Θ(15)
             for (int j = 0; j < size; j ++) b[j] = a[j];   ← Θ(15)
             delete [] a;                     ← Θ(10)
             a = b;                           ← Θ(1)
             size = newsize;                  ← Θ(1)
          }
        a[i] = i*i;             ← Θ(1)
     }
}
```

If Condition case test:

k	i
1	10
2	15 (×3/2)
3	22 (×3/2)
4	33 (×3/2)

$i \approx 6\left(\frac{3}{2}\right)^k$

Stops when $k = \log_{1.5}(n)$

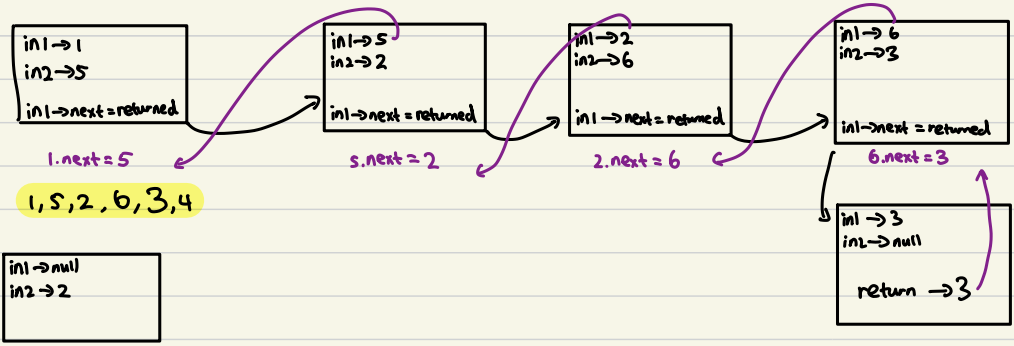
$$T(n) = \Theta(10) + \Theta(1) + \sum_{i=1}^{\log_{1.5}(n)} \left(\Theta(1) + \Theta\left(\left(\tfrac{3}{2}\right)^k\right)\right) + \sum_{i=1}^{n} \Theta(1)$$

$$= \Theta(10) + \Theta(1) + \Theta(\log n) + \Theta(n) + \Theta(n)$$

$$T(n) = \Theta(n)$$

#2

a)


in1 -> 1
in2 -> 5
in1->next = returned
1.next = 5
in1 -> 5
in2 -> 2
in1->next = returned
5.next = 2
in1 -> 2
in2 -> 6
in1 -> next = returned
2.next = 6
in1 -> 6
in2 -> 3
in1->next = returned
6.next = 3
in1 -> 3
in2 -> null
return -> 3

1, 5, 2, 6, 3, 4

b)

in1 -> null
in2 -> 2

2