Part a)

```
void f1(int n)
{
    int i = 2
    while (i < n){
        O(1)  i = i * i
    }
}
```

x	0	1	2	3
i	2	→ 4	→ 16	→ 256

growth of function 2^{2^x}

$2^{2^{(0)}} = 2^1 = 2$

$2^{2^{(1)}} = 2^2 = 4$

$n = 2^{2^x}$

$\log n = 2^x$

$\log(\log n) = x$

$O(\log(\log n))$

code runs $O(1)$ code $\log(\log(n))$ times, so $1 \cdot \log(\log(n)) = \log(\log(n))$

part 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); i++){
                O(1)
            }
        }
    }
}
```

$\sqrt{n}$ times

i^3

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

ex. $n = 9$ $\sqrt{9} = 3$ $i = 1-9$

$3\%3$, $6\%3$, $9\%3$ $\} = 0$ 3

$$\sum_{i=1}^{\sqrt{n}} \left(\sum_{k=0}^{i^3} 1 \right)$$

$1 + 1 + \ldots = i^3 + 1$

$$\sum_{i=1}^{\sqrt{n}} i^3 = (\sqrt{n}^4) = n^2$$

$O(n^2)$

Part c)

```
for (int i=1; i<=n; i++) {                          // runs n times
    for (int k=1; k<=n; k++) {                      // runs n times
        if (A[k] == i) {                            // worst case scenario all elements match i so code runs n times
            for (int m=1; m<=n; m=m+m) {            // how many times do have to add m together to get n
                // Do something that takes O(1) time
                // Assume the contents of the A[] array are not changed
            }
        }
    }
}
```

x = amount of iterations

x	1	2	3	4	5	6
m	1	2	4	8	16	32

$n = 2^{x-1}$

$m = 2^{i-1}$

$\log n = x - 1$

$2^{i-1} = 2^0 = 1$

$\log n + 1 = x \rightarrow \log n$

runtime = $n \cdot n \cdot \log n$ $\boxed{\Theta(n^2 \log n)}$

$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ arithmetic series

$\sum_{i=1}^{n} \Theta(i^p) = \Theta(n^{p+1})$

$\sum_{i=0}^{n} c^i = \frac{c^{n+1} - 1}{c-1} = \Theta(c^n)$ geometric series

$\sum_{i=1}^{n} \frac{1}{i} = \Theta(\log n)$

$\sum_{i=0}^{\log(n)} \sum_{j=0}^{2^i - 1} \Theta(1)$ because

$\sum_{i=0}^{\log(n)} \Theta(2^i) = \frac{2^{\log(n)+1} - 1}{2-1} =$

$= 2^{\log_2(n)} (2^1) - 1$

$= 2n - 1$

$= 4$

Part d)

```
int f(int n)
{
  int *a = new int[10];
  int size = 10;
  for(int i=0; i<n; i++)
  {
    if(i==size)
    {
      int newsize = 3/2(size);
      int *b = new int[newsize];
      for(int j=0; j<size; j++){ b[j] = a[j] }
      delete[] a;
      a=b;
      size = newsize
    }
    a[i] = i*i;
  }
}
```

O(1) — int *a = new int[10]; int size = 10;

for loop runs n times

if(i==size) — executes $\log_{\frac{3}{2}}(n)$ times

int newsize = $\frac{3}{2}$(size); — O(1)

int *b = new int[newsize]; — O(1)

for(int j=0; j<size; j++){ b[j] = a[j] } — runs size times

delete[] a; a=b; size = newsize — O(1)

a[i] = i*i; — O(1)

would be fractions, but integer math floors it

(1) size = 10, 15, 22, 33

times it resizes: 1, 2, 3, 4

(2) $15 \approx 10\left(\frac{3}{2}\right)$

$22 \approx 10\left(\frac{3}{2}\right)^2$

$33 \approx 10\left(\frac{3}{2}\right)^3$

(3) $10 \cdot \left(\frac{3}{2}\right)^x = n$

$\left(\frac{3}{2}\right)^x = \frac{n}{10}$

$x = \log_{\frac{3}{2}}\left(\frac{n}{10}\right)$

$= \log_{\frac{3}{2}}(n) - \log_{\frac{3}{2}}(10)$

$= \log_{\frac{3}{2}}(n)$

(4)

$$\sum_{i=0}^{\log_{3/2}(n)} \sum_{j=0}^{10\left(\frac{3}{2}\right)^i} 1$$

$$\sum_{i=0}^{\log_{3/2}(n)} 10\left(\frac{3}{2}\right)^i = 10 \sum_{i=0}^{\log_{3/2}(n)} \left(\frac{3}{2}\right)^i$$

$$= 10\left(\frac{\frac{3}{2}^{\log_{\frac{3}{2}}(n)+1} - 1}{\frac{3}{2} - 1}\right) \quad \text{(geometric series)}$$

$$= 10\left(\frac{\frac{3}{2}^{\log_{\frac{3}{2}}(n)}\left(\frac{3}{2}\right) - 1}{\frac{1}{2}}\right)$$

$$= 10\left(\frac{\frac{3}{2}n - 1}{\frac{1}{2}}\right) = 20\left(\frac{3}{2}n - 1\right) = 30n - 1 = O(n)$$