Problem 3

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

Plug in n value of 25

n = 25 i=2 → i=4 → i=16 → i > n so stop while loop

i = i * i so i is being squared so

O(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++) {
                */ O(1) time thing */
            }
        }
    }
}
```

First for loop runs n times

if statement constant time (1)

2nd for loop: has pow(i,3) → runs n^3 times

$n \cdot (1+n^3) =$ **$O(n^4)$**

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){
                // O(1) time thing
                // array contents not changed
            }
        }
    }
}
```

First for loop and second for loop run n times

If statement constant time

2nd for loop $n/2 = 1/2 * n = n$

m = m+m

$1 \to 1+1=2 \to 2+2=4 \to 4+4=8 \to 8+8=16 \to 16+16=32$

$\log n$

$\boxed{O(\log n)}$

d) First for loop for(int i=0; i<n; i++) runs n times

int newSize = 3 * size / 2

size = newSize $\to$ size = 1.5 * size

10 (1.5ish)

$\boxed{O(n^2)}$