q3 _ answers

part a) $\theta(\log(\log(n)))$

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

The number of times the while loop runs varies depending on the value of n. For example, if $n=2$ the while runs zero times. If $n=4$, the while loop runs once. If $n=16$, the while loop runs twice. If $n=256$, the while loop runs three times. If we create a variable K, and set it equal to the number of times the while loop runs, we can determine a relationship between n and K. This relationship is $n = 2^{2^K}$. From, this we can determine that the loop runs $K = \log(\log n)$ times. Thus, my final runtime is $\theta(\log(\log n))$.

| K | 0 | 1 | 2 | 3 ... | Arbitrary K | stop when K = |
|---|---|---|---|---|---|---|
| i | 2 | 4 | 16 | 256 ... | $i = 2^{2^K}$ | $i = n$ |
| n | $n \geq 2$ | $n \geq 4$ | $n \geq 16$ | $n \geq 256$ ... | | |

Since the code segment stops when $i = n$, the worst case run time is $n = 2^{2^k}$, where $k = \log(\log n)$. thus, the final runtime is $\theta(\log(\log n))$.

part b) $\Theta(n^{7/2})$

```
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 */
            }
        }
    }
}
```

In the worst case scenario, the inner for loop runs $n^3$ times. This is the case in which $i=n$. The code inside the if statement itself will run $\sqrt{n}$ times. For example, if $n=100$, the code inside the if statement will run when $i=10, 20, 30, 40, 50, 60, 70, 80, 90$, and $100$. The outer for loop will run $n$ times. Thus, the total runtime for the code is:

$$\underbrace{\Theta(n+\sqrt{n}\cdot n^3)} + \underbrace{\Theta(n-\sqrt{n})} = \Theta(n^{7/2})$$

this accounts for the worst case assuming the if statement is entered

this accounts for the case where the if statement isn't entered

$$\sum_{i=1}^{n}\Theta(1) + \sum_{i=1}^{\sqrt{n}}\sum_{k=1}^{i^3}\Theta(1) + \sum_{i=1}^{n}\Theta(1) - \sum_{i=1}^{\sqrt{n}}\Theta(1)$$

$$= \Theta(n) + \sum_{i=1}^{\sqrt{n}}(n^3) + \Theta(n) = \Theta(n+\sqrt{n}\cdot n^3) + \Theta(n-\sqrt{n}) = \Theta(n^{7/2})$$

worst case when $i=n$

part c) $\Theta(n^2)$

```
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
      }
    }
  }
}
```

The inner for loop runs $\log(n)$ times since it runs until $m=n$ and $m$ is doubled after each iteration. The if statement will evaluate to true a maximum of $n$ times so the code inside the if statement will run a total of $n$ times. Thus the run time for the code including the if statement and the inner for loop is $\Theta(n\log n)$. The outer for loops will run $n$ times each so the runtime for the outer for loops is $\Theta(n^2)$. Thus the runtime for the code is : $\Theta(n^2 + n\log n) + \Theta(n^2 - n)$. Thus, my final runtime for the code is $\Theta(n^2)$.

$$\underbrace{\Theta(n^2 + n \cdot \log n)}_{\text{This accounts for the worst case assuming the if statement is entered}} + \underbrace{\Theta(n^2 - n)}_{\text{This accounts for the worst case assuming the if statement isn't entered}} = \Theta(n^2)$$

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

dummy variable $j$ for # of times if statement is true

dummy variable for $j$: for # of times if statement is true

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

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

part d) $\Theta(n)$

```
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 * size/2;
            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;
    }
}
```

The outer for loop runs n times. The number of times the inner for loop runs is dependent on the value of n. For example, if n is 20, the inner for loop will run 10 times when $i = 10$ and 15 times when i is equal to 15. If n is 30, the inner for loop will run 10 times when i is equal to 10, 15 times when i is equal to 15 and 22 times when i is equal to 22. This means that the inner for loop runs $\left(\left(\frac{3}{2}\right)^{x} \cdot 10\right)$ times each time the size is reached. Depending on the value of n, the value of x changes. The first time the if statement is entered, x is 0, and the inner for loop runs 10 times. Then, the second time the if statement is entered, x is 1, and the inner for loop runs 15 times. When n is 30, the first time the if statement is entered, x is 0 and the inner for loop runs 10 times. The second time the if statement is run, x is 1, and the inner for loop runs 15 times. The third time the if statement is entered, x is 2 and the inner for loop runs 22 times.

part d (continued)

Since the size variables are integers, we must floor the vale of $\left(\left(\frac{3}{2}\right)^x \cdot 10\right)$, which is why the inner for loop runs 22 times even though $\left(\frac{3}{2}\right)^2 \cdot 10$ evaluates to 22.5. From our previous calculations through example values of n, we can conclude that the if statement is entered $\log_{3/2}\left(\frac{n}{10}\right)$ times. This means the total runtime for the inner for loop is $\sum_{x=0}^{\log_{3/2}\left(\frac{n}{10}\right)} 10 \cdot \left(\frac{3}{2}\right)^x$. The total runtime for the outer for loop is $\Theta(n)$.

Thus, our worst case runtime for this code segment will evaluate to

$$\Theta\left(\underbrace{n + \sum_{x=0}^{\log_{\frac{3}{2}}\left(\frac{n}{10}\right)} 10 \cdot \left(\frac{3}{2}\right)^x}_{\text{outer and inner for loop and accounting for when the if statement does evaluate to true}}\right) + \Theta\left(\underbrace{n}_{\text{outer for loop}} - \log_{\frac{3}{2}}(n)\right)$$

while accounting for when the if statement doesn't evaluate to true

We can evaluate the summation using the geometric series formula from lecture:

$$\sum_{x=0}^{\log_{\frac{3}{2}}\left(\frac{n}{10}\right)} \left(\frac{3}{2}\right)^x \cdot 10 = \frac{10 \cdot \left(\frac{3}{2}\right)^{\log_{\frac{3}{2}}\left(\frac{n}{10}\right)} - 1}{\frac{3}{2} - 1} = \frac{10 \cdot \left(\frac{n}{10}\right) - 1}{\frac{3}{2} - 1} = \frac{n-10}{\frac{1}{2}} = \Theta(n)$$

Additionally, if we plug in the max possible value to the summation equation, we have $10 \cdot \left(\frac{3}{2}\right)^{\log_{\frac{3}{2}}\left(\frac{n}{10}\right)}$ which evaluates to n. Each number before n in the summation must be less than this value, since this is the max, so the other numbers in the summation can be represented as $n-c_1$, $n-c_2$ ... where $c_1$ and $c_2$ are constants. This means that the summation will be $c_3(n)$ where $c_3$ is some constant. Thus, for this reason and the geometric one above, the summation is $\Theta(n)$.

Thus the total runtime evaluates to $\Theta(n+n) + \Theta(n) = \Theta(n)$