Big-O-Notation Analysis

1. ignore constants
2. certain terms dominate others

name: Kip Gorey

Part A:

```
void f1(int n)
{
    int i=2;
    while(i<n){
        //Θ(1)
        i=i×i;
    }
}
```

-> exponential growth

$n = i_0^{i_0^{x}}$ where x is the # of iterations through the loops

$i_0 = 2$, so, ... $\log_{i_0}(n) = i_0^{x}$

$\log(\log(n)) = x$

O runtime = $\Theta(\log(\log(n)))$

Ω runtime = $\Omega(\log(\log(n)))$

thus $\log(\log(n))$

$\sum_{x=0} \Theta(1) = \Theta(1 \cdot \log(\log(n)))$ $= \Theta(\log(\log(n)))$ for the while loop

final answer: $\boxed{\Theta(\log(\log(n)))}$ for the while loop

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);k++){
                    //Θ(1) run-time
                }
            }
}
```

outer loop runtime: $i<n$;

if statement run-time: only runs if i is divisible by sqrt n. so combined

the run time for the outer loop & if statement will be $\Theta\left(\frac{n}{\sqrt{n}}\right)$ which is really just $\Theta(\sqrt{n})$

inner loop: $k < i^3$; $\Theta(n^3)$

Final Answer

$\Theta(\sqrt{n} \cdot n^3) = \Theta(n^{\frac{1}{2}} \cdot n^{\frac{6}{2}}) = \boxed{\Theta(n^{\frac{7}{2}})}$

Part C:

Solution: $\sum \theta_{\text{runtime}} = \theta(n \cdot n \cdot 1 \cdot \log(n) \cdot 1) = \boxed{\theta(n^2 \cdot \log(n))}$

```
for (int i=1; i<=n; i++){                  --> θ(n)
    for (int k=1; k<=n; k++){              --> θ(n)
        if (A[k] == i){                    --> θ(1) *because its an if statement
            for (int m=1; m<=n; m=m+m){    --> θ(log(n))
                // do something for θ(1) runtime   --> θ(1)
            }
        }
    }
}
```

essentially doubling n every time

Outer Loop: $i<=n$, $i++$. So $\theta(n)$

2^{nd} outmost loop: $k<=n$, $k++$. So $\theta(n)$

if-statement: Only runs if $A[k]=i$

inner-loop: $m<=n$; $m+m$; so $\theta(\log(n))$

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

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

Thus, Final Answer: θ run-time $\Rightarrow$ $\theta(n^2)$

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*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;
     }
}
```

outer loop: $\sum_{i=0}^{n} \theta(1) = \theta(n)$

if statement: runs if $i=$size ; (size) $(\frac{3}{2})^n$, so if $\theta(\log(n))$ → takes precedence for $i=\log(n)$

inner-loop: occurs for $\theta(n\,(\frac{3}{2})^{\log(n)})$

run-time

$\Omega(n)$ ← run-time for $n<10$

$O(n \cdot \log(n)) = (n \log(n))$

↳ might be $O(n^2\log(n))$

→ remove $\log(n)$

Final Answer: θ run-time $= \theta(n)$

?? $\theta(n\log(n))$??