Time Complexity

1. `void func1 () {`

`int count=0;`	1
`for(int i=1; i<=n; i++){`	
`    for (int j=n/2 ; j>=1; j--){`	
`        count++;`	$n * n/2 \approx \frac{n^2}{2}$
`    }`	
`}`	
`for (int k=1; k<=2n; k++){`	
`    count--;`	$2n$
`}`	
`printf ("%d", count);`	1
`}`	

$$\therefore f(n) = \frac{n^2}{2} + 2n + 2 \approx \boxed{O(n^2)}$$

2. `void func2 () {`

`int count=0;`	1
`for (int i=1; i<=n; i++){`	
`    count++;`	n
`}`	
`for (int p=1; p<=n; p*=2){`	$\ldots\ldots \log_2 n$
`    for (int q=n; q>=1; q/=3){`	$\ldots\ldots \log_2 n \cdot \log_3 n$
`        for (int r=n; r>=1; r/=4){`	$\ldots \log_2 n \cdot \log_3 n \cdot \log_4 n$
`            count++;`	$\log_2 n * \log_3 n * \log_4 n$
`        }`	
`    }`	
`}`	
`printf ("%d", count);`	1
`}`	

$$\therefore f(n) = \log_2(n) \cdot \log_3(n) \cdot \log_4(n) + n + 2$$

$$\approx \boxed{O\left(\log_2(n) \cdot \log_3(n) \cdot \log_4(n)\right)}$$

③
```
void func3(){
    int count=0;  ———————— 1
    for(int i=1; i≤n; i++){
        for(int j=n; j≥1; j*=2){
            count++;  ———————— n*log2 n
        }
    }
                              → value of count = nlog2 n
    for(int k=1; k≤count; k++){
        printf("%d", count);  ———— n*log2 n
    }
    for(int p=1; p≤n; p++){
        break;  ———————— 1
    }
}
```

$\therefore f(n) = 2n\log_2(n) + 2 \approx \boxed{O(n\log_2 n)}$

④
```
void func4(){
    int count=0;  ———————— 1
    for(int i=n; i≥1; i/=2){ .... log2 n
        for(int j=i; j≥1; j/=3){ .... log3(i) ≈ log3(log2(n))
            for(int k=j; k≥1; k/=4){ .... log4(j)
                count++;  ———— log4(log3(log2(n)))
            }
        }
    }
}
```

$\therefore f(n) = \log_4(\log_3(\log_2(n))) + 1$

$\approx \boxed{O(\log_4(\log_3(\log_2(n))))}$