

## 17-2 겨울계절/중간/알고리즘/이현자

### 1. 빈칸 채우기

시간 복잡도는 알고리즘이 수행하는 (기본적인 연산) 횟수를 (입력크기)에 대해 함수로 표현한다.

2. 순차검색 알고리즘 →  $x$ 가 배열  $S$ 안에 없는 경우 단계별 반복 횟수 구하기 (코드 주어진고 표시된 코드들이 몇 번 수행되는지 구하기)

3.  $2^{n+1} \in O(2^n)$ 임을 보이고  $c$ 와  $N$ 을 쓰시오.

4.  $3^n \in o(5^n)$ 임을 보여라

5. 시간복잡도 카테고리 효율성 좋은 순서대로 나열하기

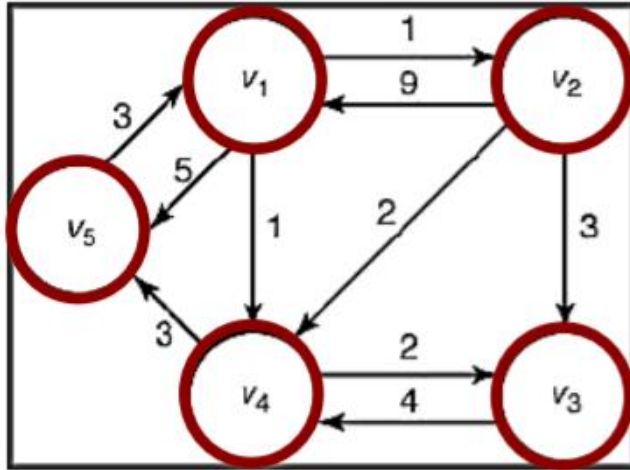
$$\theta(\lg n) < \theta(n) < \theta(n \lg n) < \theta(n^2) < \theta(n^j) < \theta(n^k) < \theta(a^n) < \theta(b^n) < \theta(n!)$$

(여기서  $k > j > 2$ 이고,  $b > a > 1$ 이다.)

6. 최소/최대값 찾기 알고리즘에서  $T(n) = 3n/2 - 2$  ( $n \geq 2$ ,  $n = 2^k$ )가 모든  $n$ 에 대해 성립함을 귀납적으로 증명하시오.

7. Dijkstra 단일출발점 최단경로 문제

주어진 그래프에 대한 인접행렬과 과정 보기



8. 이진검색 재현식 주어지고, 시간복잡도 구하기

$$n > 1 \text{ 이고, } n = 2^k (k \geq 1) \quad W(n) = \underbrace{W\left(\frac{n}{2}\right)}_{\substack{\text{재귀호출에서} \\ \text{비교횟수}}} + \underbrace{1}_{\substack{\text{최상위 레벨에서} \\ \text{비교횟수}}}, \quad W(1) = 1$$

9. QuickSort 계산 과정 보이기 (정렬할 숫자들과 퀵소트 코드 주어짐)  
(Partition 끝난 후의 배열 상황 보이기)


```
void quickSort(index low, index high){  
    index pivotPoint;  
    if(high>low){  
        pivotPoint = partition(low, high);  
        quickSort(low, pivotPoint-1);  
        quickSort(pivotPoint+1, high);  
    }  
}
```

```
index partition(index low, index high, index& pivotpoint){  
    index i, j=low;  
    keytype pivotitem = S[low];  
    for(i=low+1; i<=high; i++){  
        if(S[i]<pivotitem){  
            j++;  
            exchange S[i] and S[j];  
        }  
    }  
    pivotPoint = j;  
    exchange S[low] and S[pivotPoint];  
    Return pivotPoint;  
}
```

10. Floyd 최단 경로 알고리즘2 코드와 배열P 주어지고, 출력 결과와 과정 쓰기

```
void path(index q,r) {  
    if (P[q][r] != 0) {  
        path(q,P[q][r]);  
        cout << " v" << P[q][r];  
        path(P[q][r],r);  
    }  
}
```

11. 그리디 알고리즘의 동전 거스름돈 문제에서 while문 코드 작성하기

```
void minChange(int changes[], int n, int &cc[]){  
    cc[]={0};  
    for(i=1; i<=r;i++) //r 배열의 크기  
          
}
```