

17-2 겨울계절/중간/알고리즘/이현자

1. 빈칸 채우기

시간 복잡도는 알고리즘이 수행하는 (단위연산) 횟수를 (입력크기)에 대해 함수로 표현한다.
(기본적인연산)

← 입력값이 중요!

2. 순차검색 알고리즘 → x가 배열 S안에 없는 경우 단계별 반복 횟수 구하기 (코드 주어진 고 표시된 코드들이 몇 번 수행되는지 구하기)

$$\textcircled{1} W(n) = n \quad \textcircled{2} A(n) \quad \left. \begin{array}{l} \text{1가 배열안에 있을 확률} = p \text{ 일때} \\ \text{1가 배열의 k번째에 있을 확률} = \frac{p}{n} \end{array} \right\} \rightarrow A(n) = \sum_{k=1}^n k \cdot \frac{p}{n} + n \cdot (1-p) = \frac{p}{n} \cdot \frac{n(n+1)}{2} + n(1-p) = n(1-\frac{p}{2}) + \frac{p}{2}$$

3. $2^{n+1} \in O(2^n)$ 임을 보이고 c와 N을 쓰시오.

$f(n)$ 에 대해 $O(f(n))$ 은 $n \geq N$ 을 만족하는 모든 n 에 대해
 $g(n) \leq c \cdot f(n)$ 을 만족하는 양의 실수 c 와 음이 아닌 정수 N 이 존재하는 모든 $g(n)$ 의 집합이다.
 $\Rightarrow 2^{n+1} \leq 2 \cdot 2^n$, $n \geq 0$ 이므로 $c=2, N=0$

4. $3^n \in o(5^n)$ 임을 보여라

$f(n)$ 에 대해 $o(f(n))$ 은 $n \geq N$ 을 만족하는 모든 n 에 대해
 $g(n) \leq c \cdot f(n)$ 을 만족하는 모든 양의 실수 c 와 음이 아닌 정수 N 이 존재하는 모든 $g(n)$ 의 집합이다.
little o는 $g(n) \neq 0$ 일때 $\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = 0$ 와 같다. $\Rightarrow \lim_{n \rightarrow \infty} \frac{3^n}{5^n} = \lim_{n \rightarrow \infty} (\frac{3}{5})^n = 0$

5. 시간복잡도 카테고리 효율성 좋은 순서대로 나열하기

$$\theta(n^{\frac{1}{2}}), \theta(n^k), \theta(n), \theta(\lg n), \theta(n!), \theta(a^n), \theta(b^n), \theta(n^2), \theta(n \lg n)$$

(여기서 $k > 2$ 이고, $b > a > 1$ 이다.)

$$\theta(\lg n) < \theta(n) < \theta(n \lg n) < \theta(n^2) < \theta(n^{\frac{1}{2}}) < \theta(n^k) < \theta(a^n) < \theta(b^n) < \theta(n!)$$

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

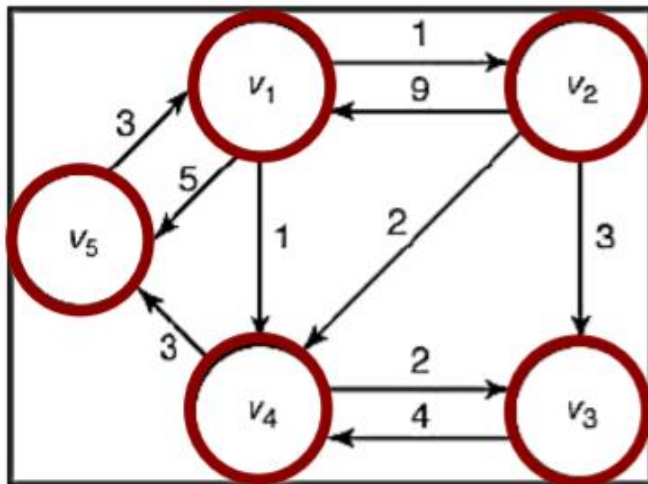
$$T(n) = \frac{3n}{2} - 2 \quad (n \leq 2, n = 2^k)$$

$$\textcircled{1} \text{ basic : } T(1) = -\frac{1}{2}$$

$\textcircled{2} \text{ hypothesis :}$

7. Dijkstra 단일출발점 최단경로 문제

주어진 그래프에 대한 인접행렬과 과정 보이기



8. 이진검색 재현식 주어지고, 시간복잡도 구하기

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

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
        
    }
```