알고리즘(수12-신찬수) 중간고사 18, October, 2017

학번:_____ **이름:** _____

1. [13] 다음의 물음에 답하시요.
   A. [1] python 소스 코드의 확장자는 _____이다.
   B. [2] $f(n) = 5n/\log n + 2n^{1.2}$를 Big-O 기호로 나타내면 $f(n)$ = _____이다.
   C. [2] 오름차순으로 정렬하는 insertion sort에 대한 최선의 경우(best case) 입력은 숫자들이 _____의 상태로 주어질 때이다.
   D. [2] n개의 수를 오름차순으로 정렬하는 insertion sort의 최악의 경우(worst case) 입력에 대한 비교 회수를 n에 관한 식으로 정확히 표현하면 _____번이다.
   E. [2] 1차원 배열에 저장된 n개의 수 중에서 k번째로 작은 수를 찾는 quick select 알고리즘의 최악의 경우의 수행시간 T(n)을 점화식으로 나타내면 T(n) = _____이다.
   F. [2] 위의 점화식 T(n)을 전개하여 Big-O 기호로 표기하면 O(_____)이다. 단, T(1)=1이며 $n = 2^k$라 가정한다.
   G. [2] Hanoi 탑 문제에서 n개의 원반을 한 막대에서 다른 막대로 모두 옮기는 데 필요한 최소 원반 이동횟수를 H(n)이라 하면, H(n)에 관한 점화식은 H(n) = _____이다. 그리고 H(1) = 1이다.

2. [13] 다음 python 코드를 실행한 결과는 각각 무엇인가? 출력 내용은 print 문장 오른쪽에 직접 쓰시오. (python version 3를 기준으로 작성한 코드이다.)

```
# 2점(1점씩)
print(3//2, 3%2)
print(int(3.14))
```

```
# 2점
s = 0
for x in range(2, 12, 3):
    s += x
print(s)
```

```
# 3점(1.5점씩)
a = [2, 3, 1]
print(a)  # [2, 3, 1]로 출력됨!
print("happy algorithm".split())
print("quickSelect".split('c'))
```

```
# 4점(2점씩)
a = []
for i in range(6):
    if i%2: a.append(2*i)
    else: a.append(2*i-1)
print(a)
a.pop()
a.pop(0)
a.insert(2, 9)
print(a)
```

```
# 2점
# max(a, b)는 a, b 중 최대값, min(a, b)는 a, b 중 최소값을 리턴.
def f(a, b):
    if a*b == 0: return a+b
    a, b = max(a, b), min(a, b)
    return f(a-b, b)
print(f(126, 54))
```

3. [5] 다음 식들을 Big-O로 표현했을 때, 가장 작은 식부터 가장 큰 식 순서로 재배열 하시요.

$$2n\log n, \quad 3^{\log_2 n}, \quad 8n^{1.1}, \quad \frac{7n^2}{\log n}, \quad n^2 - 31n + 6, \quad 2^{\sqrt{n}}, \quad n - 10^{10}$$

4. [9] 아래 코드를 보고 물음에 답하시요.

```
def g(n):
    if n == 1: return 0
    return 1 + g(n >> 1)
```

A. [2] g(100)의 값은 얼마인가?

B. [2] g(n)은 결국 어떤 값을 계산하는 함수인가?

C. [2] g(n)의 수행시간 T(n)을 T(1)의 값과 함께 점화식으로 표현하시요.

D. [3] 이 점화식을 전개한 후, Big-O 기호로 표시하시요.

5. [**8점 도전 문제**] 다음 점화식을 전개한 후, Big-O 기호로 표기하시요. (필요한 가정이 있다면 직접 언급하시요.) [hint: $n = 2^k$로 가정한 후, 치환하는 방법을 적용]

$$T(n) = 2T(\sqrt{n}) + \log n, \qquad T(1) = 1$$

6. [8] 아래 코드는 quickSelect 함수이다. 다음 물음에 답하시요.

```
def quickSelect(A, k):
    # n개의 A의 값 중에서 k번째로 작은 수 찾아 리턴
    print(A, k)  # 현재 A, k 값들을 출력함
    S, M, L = [], [], []
    p = A[0]
    for a in A:
        if _____________ : S.append(a)
        elif a > p: L.append(a)
        else: ___________________
    if _________________: return quickSelect(S, k)
    elif ______________________: return quickSelect(____, ____________________)
    else: return p
```

A. [4] 위 코드의 빈 줄 5개를 채우시요. (0.5, 0.5, 1, 1, 1점씩)

B. [4] quickSelect([4, 5, 1, 3, 10, 4, 7, 1], 7))으로 호출한다. 그러면 여러 번 출력되는 그 값을 차례대로 한 줄에 하나씩 쓰세요.

7. [**10점 도전문제**] 리스트 A에 n개의 정수가 저장되어 있는 경우, n 개의 정수 중에서 3개의 수를 뽑아 곱한 값이 최대가 되도록 하고 싶다. 예를 들어, A = [2, -5, 2, -2, 3]이 저장되 있다면, 세 개의 수를 뽑는 경우는 총 20가지 이다. 만약, 2, 2, 3를 뽑아 곱하면 2x2x3 = 12가 된다. 그러나 2, -5, -2를 뽑아 곱하면 2x(-5)x(-2) = 20이 되어 더 큰 곱이 된다. 이 예제에서는 2, -5, -2를 선택하는 것이 그 곱이 최대가 되는 경우이다.

```
def threeMax(A):
    # n개의 A의 값 중에서 세 수를 선택해 곱했을 때 가장 큰 곱셈 값을 리턴
```

A. [7] 곱이 최대가 되는 3개의 수를 찾는 코드 threeMax를 짜보자. 이때, n은 3 이상이라고 가정한다. quickSelect 함수를 필요한 만큼 마음대로 호출해도 된다. C언어, Python, 가상언어 중 어떤 언어로 작성해도 된다.

B. [3] 작성한 알고리즘의 수행시간 T(n)을 분석한 후, Big-O 기호로 표기하시요.

8. [10] 아래 함수 h에 대해, 다음 물음에 답하시요.

```
def h(A, a, b):
    if a == b: return A[a]
    m = (a+b)//2
    return max(h(A, a, m), h(A, m+1, b))
```

A. [2] h([3, 2, 4, 9, 8, 12, 4, 0], 0, 7)라고 호출했다. 리턴 값은 얼마인가?

B. [2] 함수 h는 결국 무엇을 계산하는 함수인가?

C. [3] n개의 수가 저장된 리스트 A에 대해, 함수 h를 수행했을 때의 수행시간 T(n)이라 하자. T(n)의 점화식을 T(1)과 함께 표기하시요.

D. [3] 이 점화식을 전개한 후, Big-O 기호로 표기하시요.

9. [1] 지금까지의 알고리즘 강의에 대해 느낀 점, 아쉬운 점, 개선할 점 등을 솔직히 써 주면 이 후 강의에 반영하도록 노력하겠습니다.

알고리즘설계와해석(금12-신찬수) **중간고사** **26, October, 2018**

학번:_____ **이름:** _____

1. [10.5] 다음의 물음에 답하시오.
   A. [1] A = list(range(1, 11, 2))에서 print(len(A))의 결과는? _____
   B. [1.5] $f(n) = 7n\log n + 3n^{1.2}$를 Big-O 기호로 나타내면 $f(n)$ = _____이다.
   C. [1] quick 정렬은 최악의 경우와 최선의 경우의 수행시간을 Big-O로 표기했을 때, 서로 같다. (O / X)
   D. [1.5] n개의 값을 오름차순으로 정렬하는 insertion sort의 최선의 경우(best case) 입력은 숫자들이 _____의 상태로 주어질 때이다. 이 때의 비교회수는 O( )번이다.
   E. [0.5] n개의 값이 저장된 힙(heap)의 높이를 Big-O 표기법으로 나타내면? O(_____)
   F. [1] 강의시간에 배운 정렬 알고리즘 중에서 in-place 알고리즘을 찾아 동그라미로 표시한다:
      selection, insertion, bubble, merge
   G. [1] n개의 수를 오름차순으로 정렬하는 quick sort 알고리즘의 최악의 경우(worst case)의 수행시간 T(n)을 점화식으로 나타내면 T(n) = _____이다.
   H. [1] 위의 점화식 T(n)을 전개하여 Big-O 기호로 표기하면 O(_____)이다. 단, n = 2^k로 가정한다.
   I. [1] n개의 값이 저장된 리스트 A에 대해, A.pop() 연산의 수행시간을 Big-O 기호로 표시하면 O(_____)이다.
   J. [1] n자리 수 A와 B를 곱셈하는 Karatsuba 알고리즘의 기본 덧셈과 곱셈 횟수를 T(n)이라고 하면, T(n)의 점화식은 T(n) = _____이다.

2. [4] 오른쪽 코드에 대해, 다음 질문에 답하시오.
   A. [1.5] f2(6)의 리턴 값은? _____
   B. [2.5] f2(n)의 수행시간 T(n)을 Big-O 표기법으로 표시하고, 그 이유를 설명하시오. (이유 없으면 0점)

```
def f2(n):
    c = 0
    for i in range(1, n):
        j = 1
        while j < n:
            c += 1
            j *= 2
    return c
```

3. [5] 오른쪽 코드를 보고 답하시오.
   A. [1.5] f3(A)는 무엇을 리턴하나?
   B. [1.5] len(A) = n일 때, f3(A)의 수행시간 T(n)을 점화식으로 표현하면? (처음 항도 표기해야 함.)
   C. [2] 이 점화식을 **전개한 후**, Big-O로 표현하시오.

```
def f3(A):
    if len(A) <= 1: return A
    return [A[-1]] + f3(A[:len(A)-1])
```

4. [**8점 도전 문제**] 다음 점화식을 전개한 후, Big-O 기호로 표기하시요. (필요한 가정이 있다면 직접 언급하시요.) [hint: $n = 2^k$로 가정한 후, 치환 이용]

$$T(n) = T\left(\frac{n}{2}\right) + n\log n, \qquad T(1) = 1$$

5. [7.5] 아래 코드는 quickSort 함수이다. 다음 물음에 답하시요.

```
def quickSort(A):
    if len(A) <= 1: return A
    print(A)  # 현재 A 값들을 출력함
    S, M, L = [], [], [] # S, M, L은 각각 pivot보다 작거나, 같거나, 큰 값들을 저장
    p = A[0]
    for a in A:
        if _____________ : S.append(a)
        elif a == p: _______________
        else: _________________
    return quicksort(L) + M + quicksort(S)   # 함수 호출 및 연산 순서 주의할 것!
```

A. [1.5] 위 코드의 빈 줄 3개를 채우시요. (0.5 점씩)

B. [4] A = quickSort([4, 5, 1, 3, 10, 4, 7, 1])으로 호출한다. 그러면 여러 번 출력되는데, 그 출력 값을 차례대로 한 줄에 하나씩 쓰세요.

C. [1] A = quicksort(A) 후, print(A)의 결과는? _____

D. [1] 위의 알고리즘은 stable한가? _____ in-place인가? _____

6. [6] 이진탐색 문제는 오름차순으로 정렬된 값들이 저장된 리스트 A에서 특정 값 x의 존재 여부를 결정하는 문제이다. 만약 A에 x가 존재한다면 True를 존재하지 않는다면 False를 리턴하는 함수 BS를 아래와 같이 작성해보자. 아래 물음에 답하시요.

```
def BS(A, a, b, x): # A[a] … A[b] 중에 x가 있는지 없는지 결정
    if a > b: return _________________
    m = ______________________
    if x < A[m]: return BS(_____________________________)
    elif x > A[m]: return BS(_____________________________)
    else: return _________________
```

A. [2.5] 위의 빈 칸 5 곳을 채우시요. (0.5점씩)

B. [1.5] n개의 값이 저장된 A에서의 수행시간 T(n)의 점화식을 T(1)의 값과 함께 나타내세요.

C. [2] 점화식 T(n)을 전개한 후, Big-O 기호로 표기하시요.

7. [5] 앞에서 작성한 이진탐색 BS 함수를 이용해, do_something(A, k) 함수를 아래와 같이 작성했다. 다음 물음에 답하시오.

```
def do_something(A, k):
    A.sort()
    for x in A:
        if BS(A, 0, len(A), k-x) == True: return True
    return False
```

A. [2] 위 함수는 결국 무엇을 계산하는 함수인가?

B. [3] 위 함수 do_something(A, k)의 수행시간을 T(n)을 자세히 분석하고, 이를 Big-O로 표기하시요. (분석에 대한 설명이 없으면 0점)

8. [6] 리스트 A, B에 각각 n개의 정수가 **오름차순으로 정렬**되어 저장되어 있다. 이 두 개의 리스트의 값들을 모두 모아 리스트 C에 오름차순으로 merge(합병)하고 싶다.

```
def merge_two_lists(A, B, C):
    n = len(A)
    C = []
    i, j = 0, 0          # A, B의 인덱스를 각각 i, j가 가르킴.









    return C
```

A. [4] 위의 빈 칸에 코드를 최대한 정갈하게 작성하시오.

B. [2] 작성한 함수의 수행시간을 수행시간 T(n)을 분석한 후, Big-O 기호로 표기하시요.

9. [1] 지금까지의 알고리즘 강의에 대해 느낀 점, 아쉬운 점, 개선할 점 등을 솔직히 써 주면 이 후 강의에 반영하도록 노력하겠습니다.

알고리즘설계와해석(금12)　　기말고사　　신찬수 15, Dec., 2017

학번:_____　　이름: _____

1. [13] 다음의 물음에 답하시오.

A. [1] 3개의 서로 다른 수를 정렬하는 데, _____번의 비교면 항상 충분하다.

B. [2] n개의 수를 정렬하는 insertion sort 알고리즘의 최선의 경우의 비교횟수를 Big-O 기호로 표기하면 _____이고, 최악의 경우의 비교횟수를 Big-O로 표기하면 _____이다.

C. [1] 두 문자열 X = "ABCDBAA"와 Y = "BADCADA"의 LCS(Longest Common Subsequence)의 길이는 _____ 이다.

D. [2] 1차원 배열에 저장된 n개의 정수에 대해, 가장 큰 수와 두 번째로 큰 수를 찾기 위해선 _____ 번의 비교면 충분하다. (단, n은 2^k의 형식이고, 비교횟수는 되도록 작아야 한다.)

E. [2] insertion, merge, quick, radix 정렬 알고리즘 중에서 최선의 경우와 최악의 경우의 Big-O 수행시간이 **다른** 알고리즘을 모두 나열하면? _____

F. [1] 수업시간에 배운 max-heap H의 H[k], H[2*k+1], H[2*k+2] 중 최대값은 _____이다.

G. [2] print([2*x for x in range(6) if x%2])의 결과는? _____

H. [1] 4-queens 문제에서 해(solution) 중 하나는 (2, 4, 1, 3)이다. 다른 해를 모두 적으면?

_____

I. [1] 행렬곱셈문제를 풀기 위한 다음 방법 중에서 가장 좋은 방법에 동그라미를 하시오.
(Divide-and-Conquer, Dynamic Programming, Greedy, Backtracking)

2. [15] 아래 Python 코드를 보고 물음에 답하시오.

```
def doSomething2(A, i, j):
    if A[i] > A[j]:
        A[i], A[j] = A[j], A[i]
    if (j-i+1) > 2:
        t = (j-i+1) // 3
        doSomething2(A, i, j-t)
        doSomething2(A, i+t, j)
        doSomething2(A, i, j-t)

A = [3, 2, 9, 1, -6, 8, 0]
doSomething2(A, 0, len(A)-1)
print(A)
```

A. [1] 위의 코드에서 // 연산자는 어떤 연산을 수행하는가?

B. [3] 위의 print(A)의 결과는 무엇인가?

C. [3] 위의 함수 doSomething2는 A의 저장된 값을 어떻게 처리하는가? 이유도 함께 쓰시오.

알고리즘설계와해석(금12)　　기말고사　　신찬수 15, Dec., 2017

학번:_____　　**이름:** _____

1. [13] 다음의 물음에 답하시오.
   A. [1] 3개의 서로 다른 수를 정렬하는 데, _____번의 비교면 항상 충분하다.
   B. [2] n개의 수를 정렬하는 insertion sort 알고리즘의 최선의 경우의 비교횟수를 Big-O 기호로 표기하면 _____이고, 최악의 경우의 비교횟수를 Big-O로 표기하면 _____이다.
   C. [1] 두 문자열 X = "ABCDBAA"와 Y = "BADCADA"의 LCS(Longest Common Subsequence)의 길이는 _____ 이다.
   D. [2] 1차원 배열에 저장된 n개의 정수에 대해, 가장 큰 수와 두 번째로 큰 수를 찾기 위해선 _____ 번의 비교면 충분하다. (단, n은 2^k의 형식이고, 비교횟수는 되도록 작아야 한다.)
   E. [2] insertion, merge, quick, radix 정렬 알고리즘 중에서 최선의 경우와 최악의 경우의 Big-O 수행시간이 **다른** 알고리즘을 모두 나열하면? _____
   F. [1] 수업시간에 배운 max-heap H의 H[k], H[2*k+1], H[2*k+2] 중 최대값은 _____이다.
   G. [2] print([2*x for x in range(6) if x%2])의 결과는? _____
   H. [1] 4-queens 문제에서 해(solution) 중 하나는 (2, 4, 1, 3)이다. 다른 해를 모두 적으면?
   _____
   I. [1] 행렬곱셈문제를 풀기 위한 다음 방법 중에서 가장 좋은 방법에 동그라미를 하시오.
   (Divide-and-Conquer, Dynamic Programming, Greedy, Backtracking)

2. [15] 아래 Python 코드를 보고 물음에 답하시오.

```
def doSomething2(A, i, j):
    if A[i] > A[j]:
        A[i], A[j] = A[j], A[i]
    if (j-i+1) > 2:
        t = (j-i+1) // 3
        doSomething2(A, i, j-t)
        doSomething2(A, i+t, j)
        doSomething2(A, i, j-t)

A = [3, 2, 9, 1, -6, 8, 0]
doSomething2(A, 0, len(A)-1)
print(A)
```

   A. [1] 위의 코드에서 // 연산자는 어떤 연산을 수행하는가?

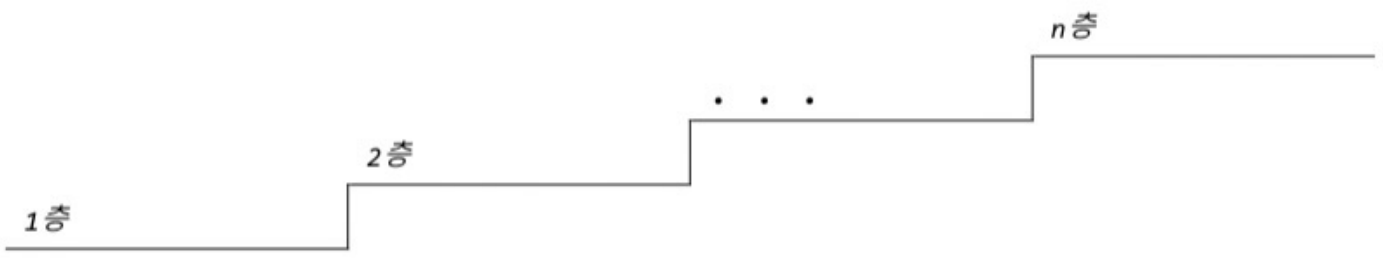
   B. [3] 위의 print(A)의 결과는 무엇인가?

   C. [3] 위의 함수 doSomething2는 A의 저장된 값을 어떻게 처리하는가? 이유도 함께 쓰시오.

D. [3] A에 n개의 값이 저장된다고 하면, doSomething2(A, 0, len(A)-1)의 수행시간 T(n)을 점화식으로 표현하시오. (T(1) = c(상수)라고 가정한다.)

E. [5] 점화식 T(n)을 전개한 후, Big-O 기호로 표기하시오. (필요한 가정이 있으면 하시오.)

3. [7] 계단에서 한 아이가 놀고 있다. 아래 그림처럼 계단의 수가 n개라 하자 (n > 4라고 가정) 아이는 한번에 **1계단 또는 3계단씩** 뛰어 올라갈 수 있다. 이 아이가 1층에서 출발하여 n층에 도달할 수 있는 모든 경우의 수가 얼마인지 계산하고 싶다. 다음 물음에 답하시오.



A. [4] 이 문제를 해결하는 **동적계획법** 알고리즘을 python 코드로 작성하시오.

```
def jump(n):  # n층에 도달하는 경우의 수를 return 해야 함!
        A = [0]*(n+1)    # A[i]에는 i층에 도달하는 경우의 수를 저정하면 됨
```

B. [3] 위 알고리즘의 수행시간 T(n)이 얼마인지 설명하고, Big-O 기호로 표기하시오.

4. [7] 아래 코드를 보고 물음에 답하시오.

```
def f(n):
    if n <= 1: return 1
    return f(n//2) + f(n//3)
```

A. [2] f(100)의 출력 값은?

B. [2] f(n)의 수행시간 T(n)을 점화식으로 표현하시오. (T(1) = 1)

C. [3] 위의 점화식을 $T(n) \le 6n$ 임을 n 값에 대한 **귀납법**으로 증명하고 싶다. n = 1일 때, $T(1) = 1 \le 6$ 이므로 성립한다. n 미만의 값에 대해 성립한다고 가정하고, n일 때 성립함을 보이시오.

5. [7] 6개의 알파벳 a, b, c, d, e, f의 각 빈도수(frequency)가 3, 2, 16, 14, 9, 6라 하자. Huffman 알고리즘을 이용하여, 알파벳에 할당된 prefix-free 이진 코드의 길이에 빈도수를 곱한 값의 합(비용)이 가장 작아지도록 하고 싶다. 다음 물음에 답하시오.

A. [4] 최소 비용의 이진 prefix-free 코드 할당을 위한 트리를 에지들이 **서로 교차하지 않도록** 오른쪽 빈 곳에 그리세요.

B. [3] 최소비용은 얼마인가? (계산과정필요)

6. [16] n개의 아이템의 크기(size)는 s[1], ... , s[n]이고 가치(profit)는 p[1], ..., p[n]이라 하자. 크기가 K인 배낭이 주어졌을 때, n개의 아이템들 중에서 몇 개를 선택하여 배낭에 넣는데, 선택된 아이템들의 크기의 합이 K를 넘지 않으면서 가치의 합은 최대가 되도록 하는 문제가 knapsack 문제이다. **0/1 knapsack 문제**에서는 선택된 아이템이 나누어질 수 없다. 만약, 선택한 아이템을 나누어 넣을 수 있다면 그 문제를 **fractional knapsack 문제**라 한다.

A. [2] fractional knapsack 문제의 최적해는 어떻게 구할 수 있는가? 그 방법을 구체적으로 설명하시오.

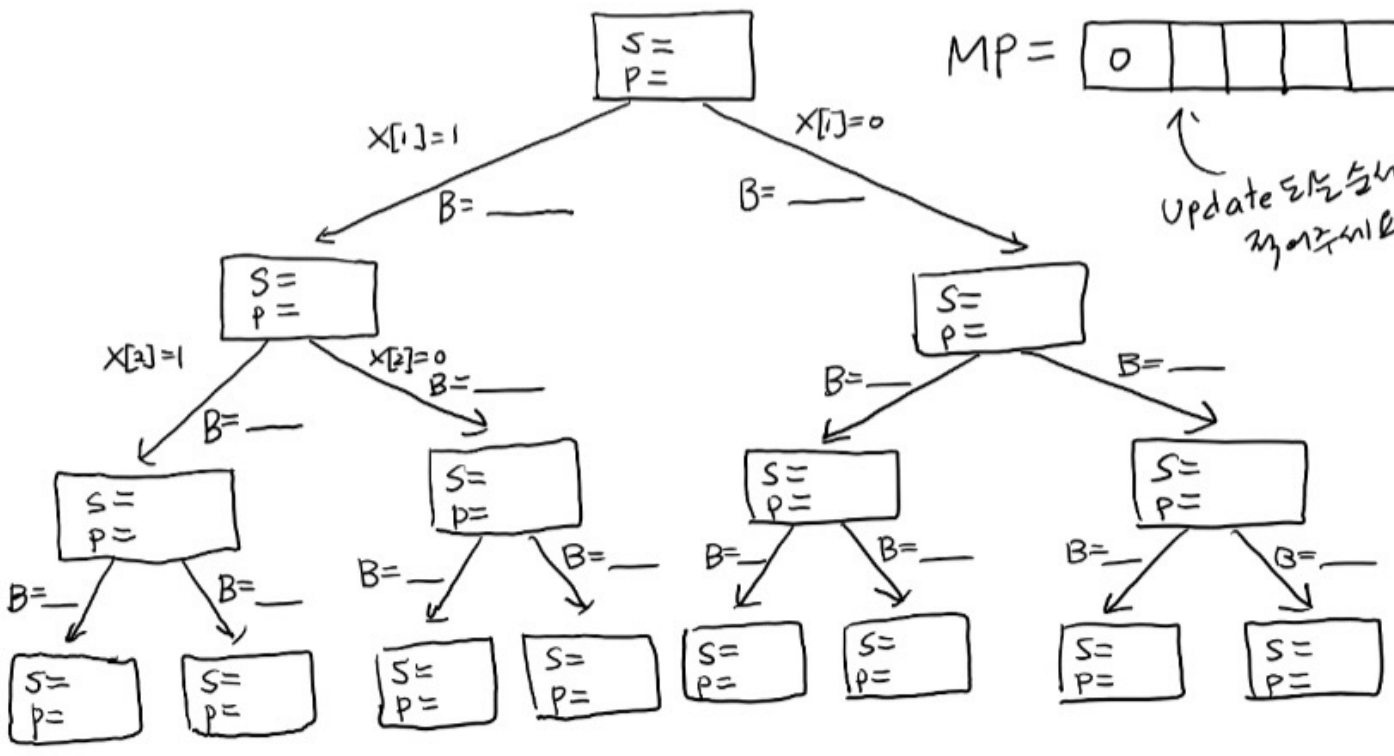
B. [2] 0/1 knapsack 문제에 위의 greedy 알고리즘을 적용하면 항상 최적의 해를 찾는가? 증명하거나 반례를 찾아라.

아래 C번부터는 0/1 knapsack 문제에 대한 물음이다. 해(solution)는 아이템 i를 선택할지 말지를 결정하는 것이다. 따라서 배열 x에 대해, 아이템 i가 선택되면 x[i] = 1, 아니면 x[i] = 0을 결정하면 된다. State space tree에서 현재 탐색 중인 (상태)노드 v에서의 가치의 합을 p, 크기의 합을 s, 현재까지 방문한 노드의 p 값 중에서 최대 값을 MP라 한다. 이 때, x[i] = 1 또는 x[i] = 0인 에지를 따라 탐색할 때, 이 에지를 따라 내려가 얻을 수 있는 최대 총 가치의 정확한 값을 R이라 하자. 이 R 값은 알 수 없기 때문에, 최대 가치의 **예상** 값 (한계함수 값) B를 정의하여 사용한다.

C. [3] 모든 에지에 대해, $R \le B$ 이 성립해야 한다. 그 이유는 무엇인가?

D. [3] fractional knapsack 문제의 해를 이용하여 B를 정의하면 위의 C번의 부등식 조건이 성립하는가? 그 이유는?

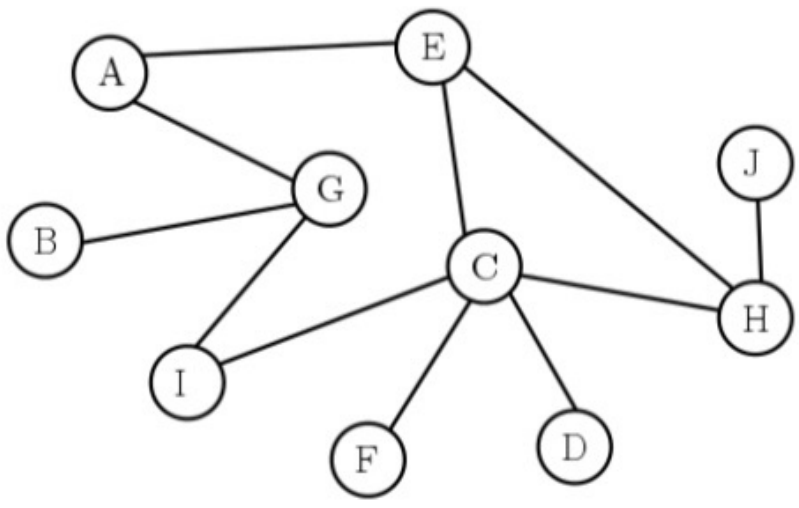
E. [6] 만약, 아이템의 크기 s = [3, 4, 2], 이익 p = [12, 20, 6]이고, 배낭 크기가 K = 5일 때, 아래 그림에 나타난 state space tree의 빈 칸을 채우시오. 아이템 번호 i는 1번부터 시작하고 (즉, 아이템 1은 크기가 s[1] = 3, 가치가 p[1] = 12), 주어진 아이템 번호는 변경하지 않는다. 트리를 DFS(깊이우선탐색)에 의해 방문한다. 즉, x[i] = 1인 경우를 먼저 방문한 후, x[i] = 0인 경우를 방문한다. **주의할 점**: 위의 D번에서의 B 값을 이용하여, 현재 방문 노드의 자손 노드를 방문할 필요가 없다고 판단될 때에는 자손 노드를 모두 X자로 그려 방문하지 않아도 됨을 표시하라. 방문하지 않는 노드와 에지에 대해선 s, p, B, MP등을 계산할 필요는 없다



7. [5] 아래 그래프를 노드 A부터 DFS 탐색법으로 방문하려고 한다. 한 노드에서 갈 수 있는 노드가 두 개 이상이라면, 알파벳 순서가 빠른 노드부터 방문한다고 가정한다. [time = 1부터 시작한다.]

   A. [3] 노드의 방문 순서를 차례대로 나열하시오. 즉, pre[] 값이 커지는 순서로 노드 이름을 중복없이 나열하시오. (부분 점수 없음)

   B. [2] 노드 I의 pre[I] 값 _____, post[I] 값 _____



한 학기 동안 수고했습니다. 앞으로도 자료구조/알고리즘에 관심을 많이 갖길 바랍니다. 즐겁고 건강한 방학 보내기 바랍니다.