5장 연습문제

5번, 6번, 10번, 11번, 12번, 16번, 18번

**5. 리스트 A, L, G, 0, R, I, T, H, M을 병합 정렬을 이용해 오름차순으로 정렬하라.**

def merge\_sort(A, left, right) :        # A[left..right]를 오름차순으로 정렬

    if left<right :                     # 항목이 2개 이상인 경우

        mid = (left + right) // 2       # 리스트의 균등 분할

        merge\_sort(A, left, mid)        # 부분 리스트 정렬

        merge\_sort(A, mid + 1, right)   # 부분 리스트 정렬

        merge(A, left, mid, right)      # 병합

    # else: 항목이 1개 인 경우. 자동으로 정복되었음(하나이므로)

def merge(A, left, mid, right) :

    k = left                    # 병합을 위한 임시 리스트의 인덱스

    i = left                    # 왼쪽 리스트의 인덱스

    j = mid + 1             # 오른쪽 리스트의 인덱스

    while i<=mid and j<=right :

        if A[i] <= A[j] :       # 기본 연산

            sorted[k] = A[i]

            i, k = i+1, k+1

        else:

            sorted[k] = A[j]

            j, k = j+1, k+1

    if i > mid :                # 한쪽에 남아 있는 레코드의 일괄 복사

        sorted[k:k+right-j+1] = A[j:right+1]    # 리스트의 슬라이싱 이용

    else :

        sorted[k:k+mid-i+1] = A[i:mid+1]        # 리스트의 슬라이싱 이용

    A[left:right+1] = sorted[left:right+1]      # sorted를 A에 다시 복사

MergeSort : ['A', 'G', 'H', 'I', 'L', 'M', 'O', 'R', 'T']

**6. 다음 리스트를 병합 정렬을 이용해 오름차순으로 정렬하라. 각 단계에서의 배열의 내용을**

**나타내어라.**



분할

74963875

7496 3875

74 96 38 75

정복

7 4 9 6 3 8 7 5

병합

47 69 38 57

4679 3578

34567789

**10. 리스트 A, L, G, 0, R, I, T, H, M을 퀵 정렬을 이용해 오름차순으로 정렬하라.**

def quick\_sort(A, left, right) :        # A[left..right]를 오름차순으로 정렬

    if left<right :                     # 정렬 범위가 2개 이상인 경우

        mid = partition(A, left, right) # 좌우로 분할

        quick\_sort(A, left, mid-1)      # 왼쪽 부분리스트를 퀵 정렬

        quick\_sort(A, mid+1, right)     # 오른쪽 부분리스트를 퀵 정렬

def partition(A, left, right) :

    low = left + 1              # 왼쪽 부분 리스트의 인덱스 (증가방향)

    high = right                    # 오른쪽 부분 리스트의 인덱스 (감소방향)

    pivot = A[left]                 # 피벗 설정

    while (low <= high) :           # low와 high가 역전되지 않는 한 반복

        while low <= right and A[low] < pivot : low += 1

        while high >= left and A[high]> pivot : high-= 1

        if low < high :         # 선택된 두 레코드 교환

            A[low], A[high] = A[high], A[low]

    A[left], A[high] = A[high], A[left] # 마지막으로 high와 피벗 항목 교환

    return high                 # 피벗의 위치 반환

QuickSort : ['A', 'G', 'H', 'I', 'L', 'M', 'O', 'R', 'T']

**11. 다음의 정수 배열을 퀵 정렬을 사용하여 오름차순으로 정렬하려고 한다. 각 단계에서의 배**

**열의 내용을 나타내어라. 단, 배열의 첫 번째 요소를 피벗으로 선택한다.**



71 49 92 55 38 82 72 53

49 55 38 53 [71] 92 82 72

38 [49] 55 53 [71] 82 72 [92]

[38] [49] 53 [55] [71] 72 [82] [92]

[38] [49] [53] [55] [71] [72] [82] [92]

**12. 퀵 정렬에서 불균형 분할을 완화하기 위해 리스트의 왼쪽, 오른쪽, 중간의 3개의 항목 중**

**에서 중간값을 피벗으로 선택하는 방법(median of three)을 구현하라. 이를 위해 알고리즘**

**4.12의 partition()을 수정하라.**

def median\_of\_three(A, low, high):

    mid = (low + high - 1) // 2

    if ((A[low] < A[mid] and A[mid] < A[high]) or (A[high] < A[mid] and A[mid] < A[low])):

        return A[mid]

    elif ((A[mid] < A[low] and A[low] < A[high]) or (A[high] < A[mid] and A[mid] < A[low])):

        return A[low]

    else:

        return A[high]

def partition(A, low, high):

    pivot = median\_of\_three(A, low, high)

    i = low - 1

    j = high + 1

    while True:

        i += 1

        while A[i] < pivot:

            i += 1

        j -= 1

        while A[j] > pivot:

            j -= 1

        if i >= j:

            return j

        A[i], A[j] = A[j], A[i]

**16. 주어진 이진트리에서 모든 노드의 수를 계산하는 알고리즘을 분할 정복 기법으로 설계하라.**

class TNode:                        # 이진트리를 위한 노드 클래스

    def \_\_init\_\_ (self, data, left, right): # 생성자

        self.data = data            # 노드의 데이터

        self.left = left            # 왼쪽 자식을 위한 링크

        self.right = right          # 오른쪽 자식을 위한 링크

def count\_node(root):

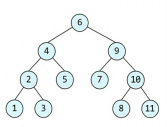
    if root is None:

        return 0

    else:

        return 1 + count\_node(root.left) + count\_node(root.right)

**18. 다음의 이진트리에 대하여 질문에 답하라.**



1. 이 트리를 전위 순회한 결과를 적어라.

6-4-2-1-3-5-9-7-10-8-11

1. 이 트리를 중위 순회한 결과를 적어라.

1-2-3-4-5-6-7-9-8-10-11

1. 이 트리를 후위 순회한 결과를 적어라.

1-3-2-5-4-7-8-11-10-9-6