04장 연습문제  : 2번, 4번, 5번, 6번

프로그램 작성

거듭제곱 프로그램 power.py를 수정하여 2의 거듭제곱이 100에서 1000까지 변하며 그 시간을 측정 출력하도록 하세요.

ex)

N=100 억지기법 : xxxx초 축소정복기법 : xxx초

N=200 억지기법 : xxxx초 축소정복기법 : xxx초

....

N=1000 억지기법 : xxxx초 축소정복기법 : xxx초

**2. 리스트 A, L, G, O, R, I, T, H, M을 삽입 정렬을 이용해 오름차순으로 정렬하라.**

def insertion\_sort(arr):

    for a in range(1, len(arr)):

            for b in range(a, 0, -1):

                if arr[b] < arr[b-1]:

                    arr[b], arr[b-1] = arr[b-1], arr[b]

                else:

                    break

    return arr

a = ['A', 'L', 'G', 'O', 'R', 'I', 'T', 'H', 'M']

print(a)

print(insertion\_sort(a))

**4. 삽입 정렬이 안정성을 만족하는지를 설명하라.**

삽입 정렬은 안정성을 만족한다 = 동일한 두 원소 간의 순서가 정렬 후에 바뀌지 않는다

[5, 3-1, 2, 3-2] 라는 배열이 있다면

[3-1, 5, 2, 3-2]

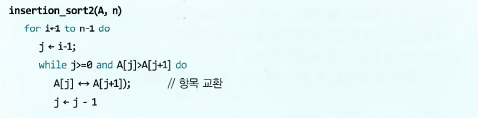
[3-1, 2, 5, 3-2]

[2, 3-1, 5, 3-2]

[2, 3-1, 3-2, 5]

이처럼 3이 들어온 순서대로 정렬되어 있다

**5. 다음은 삽입 정렬의 변형된 알고리즘이다.**



(1) 이 알고리즘을 파이썬으로 구현하라.

def insertion\_sort2(A, n):

    for i in range(1, n):

        j = i-1

        while j>=0 and A[j] > A[j+1]:

            A[j], A[j+1] = A[j+1], A[j]

            j -= 1

    return A

a = ['A', 'L', 'G', 'O', 'R', 'I', 'T', 'H', 'M']

print(a)

print(insertion\_sort2(a, len(a)))

(2) 이 알고리즘의 효율은 어떨까? 알고리즘 4.3과 비교하여 설명하라.

복사 횟수가 더 많아 비효율적이다

**6. 쉘 정렬(shell sort)은 삽입 정렬이 어느 정도 정렬된 배열에 대해서는 대단히 빠른 것에 착안한 정렬 방법이다. 이 정렬 방법을 찾아보고 삽입 정렬과의 차이를 설명해 보라.**

쉘 정렬은 일정 간격 떨어져 있는 원소들끼리 부분집합을 구성한 후 각 부분집합의 원소들에 대해서 삽입 정렬을 수행하되, 간격을 줄여가며 삽입 정렬을 반복하여 전체 원소들을 정렬하는 방식의 정렬 알고리즘이다

삽입 정렬은 자료가 역순으로 정렬되어 있을수록 비교 횟수가 늘어나는 치명적인 문제점이 존재했고, 이 문제점을 간격이라는 개념을 도입해 극복해낸 정렬 방법이 쉘 정렬이다

**거듭제곱 프로그램 power.py를 수정하여 2의 거듭제곱이 100에서 1000까지 변하며 그 시간을 측정 출력하도록 하세요.**

ex)

N=100 억지기법 : xxxx초 축소정복기법 : xxx초

N=200 억지기법 : xxxx초 축소정복기법 : xxx초

....

N=1000 억지기법 : xxxx초 축소정복기법 : xxx초

def slow\_power(x, n) :      # 반복으로 을 구하는 함수

    result = 1.0

    for i in range(n):          # 루브: n번 반복

        result = result \* x

    return result

def power(x, n) :

    if n == 0 :                     # 종료 조건

        return 1                        # 모든 수의 0승은 1

    elif (n % 2) == 0 :             # n이 짝수

        return power(x\*x, n//2)         # 정수의 나눗셈

    else :                          # n이 홀수

        return x \* power(x\*x, (n-1)//2)

import time     # time 모듈 포함

for i in range(1, 11):

    n = i\*100

    t1 = time.time()

    for i in range(100000) : power(2.0, n)          # 축소정복 10만회

    t2 = time.time()

    for i in range(100000) : slow\_power(2.0, n)     # 억지기법 10만회

    t3 = time.time()

    print("N=%d 억지기법 = %.3f초 축소정보기법 : %.3f초" % (n, t3-t2, t2-t1))