2022년 1학기 알고리즘분석 과제1

- 과제의 프로그램 소스와 보고서를 작성하여 e-campus에 업로드
  - ■ 프로그램 소스: 알고리즘 A,B를 한 번에 수행될 수 있도록 하나의 파이썬 프로그램으로 만들어 이름+학번+hw1.py 저장. 저장된 프로그램 소스는 n=5,000, 10,000 에 대해 수행되도록 작성
  - ■ 보고서: 이름+학번+hw1.pdf 로 저장

- 보고서에는 과제 내용의 알고리즘A,B 및 (2),(3),(4),(5)의 답변을 작성. 시간 및 문제 크 기의 추정 근거를 서술
- 두 파일을 e-campus에 기한 내에 업로드
- 동일한 과제를 제출한 모든 학생들에게 페널티 부과

n개의 데이타 (키값은 1~1,000 사이의 자연수를 random으로 생성)를 비내림차순으로 정렬하는 문제에 대해

(1) $O(n^2)$ 알고리즘인 insertion sort(알고리즘 A)와 평균적으로 $O(n \log_2 n)$ 알고리즘 quick sort(알고리즘 B)를 python으로 구현한다. quick sort의 시간복잡도 분석은 본 강의 4주차 1차시에 설명되어 있다.

(2) 다음의 문제 크기 n에 대해 알고리즘 A, B가 종료될 때까지의 시간을 측정하여 다음 테이블에 채워 넣으시오.

```python
import time

N = [5000, 10000, 20000, 30000, 40000, 80000]

print("================== Insertion Sort ==================")
for n in N:
    if (n > 40000):
        break;

    print(f"n = {n}₩n")
    A = [randrange(1, 1001) for i in range(0, n)]
    start = time.time()
    insertion_sort(A)
    end = time.time()
    print(f"{end - start:.6f} sec")
    print()

print("================== Quick Sort ==================")
for n in N:
    print(f"n = {n}₩n")
    A = [randrange(1, 1001) for i in range(0, n)]
    start = time.time()
    quick_sort(A, 0, len(A)-1)
    end = time.time()
    print(f"{end - start:.6f} sec")
    print()

```

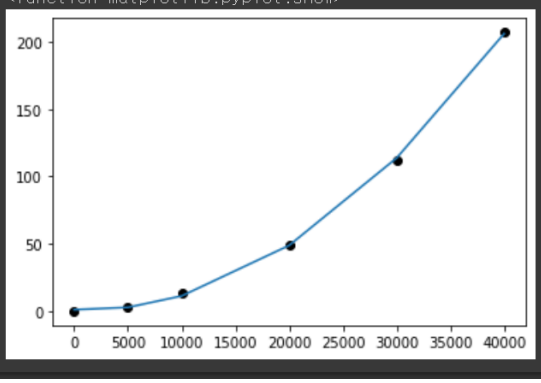
n	알고리즘 A	알고리즘 B
5,000	3.129395	0.016636
10,000	13.204421	0.034139
20,000	48.890387	0.066713
30,000	111.556953	0.110518
40,000	206.995246	0.148968
80,000		0.324701

단위 : 초

(3) 알고리즘 A는 n개의 입력에 대해 수행시간을 $f_A(n) = an^2 + bn$, 알고리즘 B는 n개의 입력에 대해 수행시간을 $f_B(n) = cn\log_2 n$ 로 표현한다. (2)에서 측정된 시간을 이용하여 a,b,c의 값을 구하라.

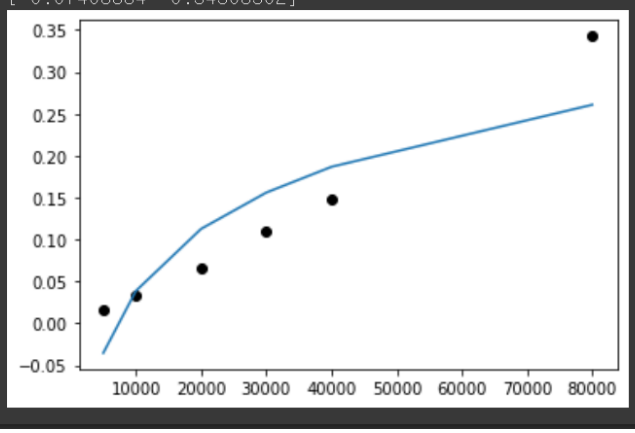
```
#insertion sort
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

N = np.array([0, 5000, 10000, 20000, 30000, 40000])
T = np.array([0, 3.129395, 13.204421, 48.890387, 111.556953, 206.995246])
num = len(N)

fit = np.polyfit(N, T, 2)
print(fit)


fit2 = fit[0]*(N**2) +fit[1]*N + fit[2]

plt.scatter(N, T, c='black')
plt.plot(N, fit2)
plt.show
```

```
[ 1.36802874e-07 -3.53225521e-04  1.17273121e+00]
<function matplotlib.pyplot.show>
```



```
N = np.array([5000, 10000, 20000, 30000, 40000, 80000])
T = np.array([0.016636, 0.034139, 0.066713, 0.110518, 0.148968, 0.342701])
num = len(N)
y = np.zeros(num)

for i in range(num):
    j = N[i]
    y[i] = math.log(j, 2)

fit = np.polyfit(np.log2(N), T, 1)
print(fit)


fit2 = fit[0]*np.log2(N)+fit[1]

plt.scatter(N, T, c='black')
plt.plot(N, fit2)
plt.show()
```

```
[ 0.07403334 -0.94503902]
```



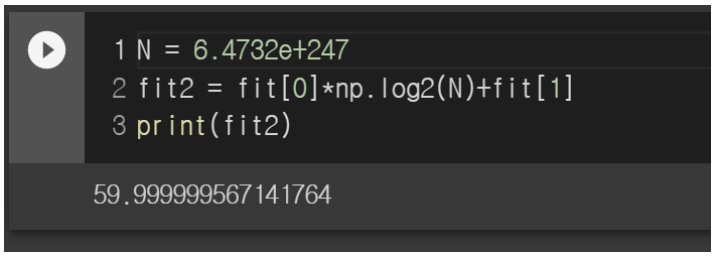
회귀분석을 이용하여 f(n)을 계산하였습니다. 이때, f_A(n), f_B(n) 모두 상수항이 없으므로 f_A(n)을 구할 때는 N = 0, T = 0, f_B(n)을 구할 때는 N = 1, T = 0인 case도 추가하여 회귀분석을 진행하였습니다. 다만, 조금 더 정확한 결과를 얻고자 $f\ (n) = an^2 + bn + \mathrm{u}$, $f_B(n) = cn\log_2 n +$ v 의 꼴로 수행시간을 계산하였습니다.

f_A(n): a = 1.3680e-7, b = -3.5323e-4, u = 1.1727e+0
f_B(n): c = 0.0740e+0, v = -0.9450e+0

(4) 우리나라 인구수는 5,000만명이 넘는다. n=5,000만일 때의 알고리즘 A의 수행시간을 (3)의 결과를 이용하여 추정한다. 추정 결과를 year 단위로 표시하라.

1년은 31,536,000 = 3.1536e+7sec이다.
따라서 위에서 구한 f_A(n)에 n = 5e+7을 대입하고, 그 결과를 3.1536e+7로 나눠주면
n=5000만일 때 알고리즘 A의 수행시간을 약 10.8444 year의 시간이 걸림을 추정할 수 있다.

(5) 알고리즘 B를 컴퓨터로 1분간 수행할 때 해결할 수 있는 문제의 크기 n' 를 (3)의 결과를 이용하여 추정한다.



f_B(n') = 0.047e+0log2(n')-0.945e+0 = 60을 만족하는 n'은 약 6.4732e+247이다.