[프로그래밍 과제]

무작위로 n개의 2차원 좌표를 발생시키고 closest\_pair\_dist()알고리즘을 수행하여 결과를

확인해 보라. 점의 개수를 증가시킴에 따라 실제 처리시간이 어떻게 늘어나는지 측정하고

이를 선 그래프로 그려보라.

소스와 실행결과를 제출하세요.

**소스코드)**

import math                 # math 모듈 포함. 제곱근(sqrt) 함수 사용을 위해

import random

import time

import matplotlib.pyplot as pyplot

def distance(p1, p2):       # Euclidean 거리 계산 함수

    return math.sqrt((p1[0]-p2[0])\*\*2 + (p1[1]-p2[1])\*\*2)

def closest\_pair(p):

    n = len(p)                          # 점의 전체 개수

    mindist = float("inf")              # 최근접 거리 초기화(무한대)

    for i in range(n-1):

        for j in range(i+1, n):

            dist = distance(p[i], p[j]) # 유클리드 거리 계산

            if dist < mindist:

                mindist = dist

    return mindist

def strip\_closest(P, d):

    n = len(P)                  # 리스트내의 점의 수

    d\_min = d

    P.sort(key = lambda point: point[1])  # y축을 따라 정렬

    for i in range(n):          # y가 최소인 점부터 순서대로

        j = i + 1

        # P[i].y와 P[j].y의 차이가 d\_min 이내일 때 까지만 처리

        while j < n and (P[j][1] - P[i][1]) < d\_min:

            dij = distance(P[i], P[j])

            if dij < d\_min :

               d\_min = dij

            j += 1

    return d\_min                # d\_min 반환

def closest\_pair\_dist(P, n):

    if n <= 3:                      # 점이 3개 이하이면, brute force로 바로 계산

        return closest\_pair(P)      # 억지기법 알고리즘(알고리즘 3.4)

    mid = n // 2            # 중앙점을 찾음. P는 현재 x로 정렬되어 있음

    mid\_x = P[mid][0]       # 중앙점의 x좌표

    dl = closest\_pair\_dist(P[:mid], mid)    # Pl에서 dl 계산

    dr = closest\_pair\_dist(P[mid:], n-mid)  # Pr에서 dr 계산

    d = min(dl, dr)     # d는 둘 중에서 더 짧은 거리

    Pm = []             # 중앙에서 x좌표가 d이내인 점들의 집합 Pm을 만듦

    for i in range(n):  # Pm도 x에 대해 정렬되어 있음

        if abs(P[i][0] - mid\_x) < d:

            Pm.append(P[i])

    ds = strip\_closest(Pm, d)   # Pm내에서 d보다 작은 최근접쌍 거리 찾기

    return min(d, ds)

a = [] # x축 (좌표의 개수)

b = [] # y축 (처리 시간)

for i in range(1, 11):

    n = random.randrange(10000\*i, 10100\*i)

    a.append(n)

    p = [0 for i in range(n)]

    for j in range(n):

        p[j] = (random.randrange(1, 1000000), random.randrange(1, 1000000))

    p.sort(key = lambda point: point[0])    # 전처리: 점들을 x순으로 정렬

    starttime = time.time()

    print("가장 가까운 두 점의 거리", closest\_pair\_dist(p, len(p)))

    endtime = time.time()

    print("좌표의 개수 = %d, 처리 시간 = %f\n" % (n, (endtime - starttime)))

    b.append(endtime - starttime)

pyplot.plot(a, b)

pyplot.show()

**실행결과)**



