

자료구조응용

03. step count, 성능측정 (9점)

2022.3.14

1. 3행 4열의 행렬 2개(a.txt, b.txt)를 파일입력 받아 행렬 더하기 함수(Program 1.16)의 Program Step count를 계산하여 출력하는 프로그램을 작성하라.(3점)

```
void add(int a[][MAX_SIZE], int b[][MAX_SIZE],
         int c[][MAX_SIZE], int rows, int cols)
{
    int i, j;
    for (i = 0; i < rows; i++)
        for (j = 0; j < cols; j++)
            c[i][j] = a[i][j] + b[i][j];
}
```

Program 1.16: Matrix addition

▶▶입력 파일 형식

| a.txt | b.txt |
|------------|---------|
| 1 2 3 4 | 1 1 1 1 |
| 5 6 7 8 | 2 2 2 2 |
| 9 10 11 12 | 3 3 3 3 |

▶▶구현세부사항

- ① main의 지역변수로 3행 4열의 2차원 배열 3개를 선언하여 사용하라.
- ② Program 1.17 혹은 1.18로 add 함수의 스텝카운트를 구하여 화면 출력한다.

▶▶실행예

step count : ____

2. 다음 Program 1.24를 사용하여 선택정렬에 대한 성능측정을 하라. 그리고 프로그램의 실행결과를 이용하여 그래프를 작성하라. (3점)

```
#include <stdio.h>
#include <time.h>
#include "selectionSort.h"
#define MAX_SIZE 1001
void main(void)
{
    int i, n, step = 10;
    int a[MAX_SIZE];
    double duration;
    clock_t start;

    /* times for n = 0, 10, ..., 100, 200, ..., 1000 */
    printf("    n    time\n");
    for (n = 0; n <= 1000; n += step)
    /* get time for size n */

        /* initialize with worst-case data */
        for (i = 0; i < n; i++)
            a[i] = n - i;

        start = clock( );
        sort(a, n);
        duration = ((double) (clock() - start))
                    / CLOCKS_PER_SEC;
        printf("%6d    %f\n", n, duration);
        if (n == 100) step = 100;
    }
}
```

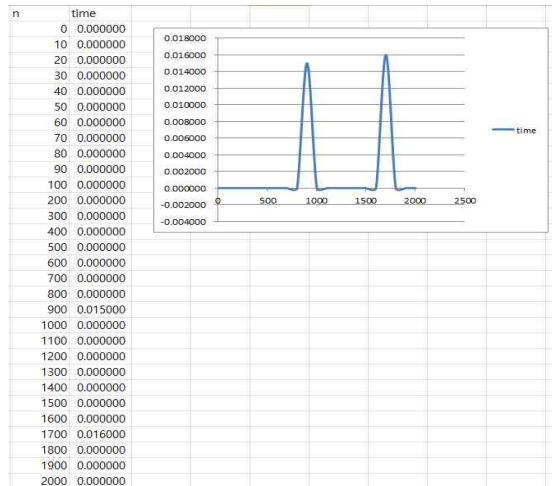
Program 1.24: First timing program for selection sort

▶▶ 구현세부사항

- ❶ Program 1.24 외에 “selectionSort.h”와 “selectionSort.c” 파일을 프로젝트에 추가
- ❷ “selectionSort.h”에는 SWAP 매크로 정의문, sort 함수원형을 작성
- ❸ “selectionSort.c”에는 매크로 SWAP을 사용하는 sort함수를 정의
- ❹ n을 2000까지 증가시켜 데이터를 생성하도록 소스를 수정하기
- ❺ 성능측정 결과는 매 측정 마다 화면 및 파일(out.txt)로 출력
- ❻ 출력결과를 엑셀 프로그램에 입력하여 그래프를 그려볼 것

▶▶ 실행결과에 대한 그래프 작성 예

| n | time |
|------|----------|
| 0 | 0.000000 |
| 10 | 0.000000 |
| 20 | 0.000000 |
| 30 | 0.000000 |
| 40 | 0.000000 |
| 50 | 0.000000 |
| 60 | 0.000000 |
| 70 | 0.000000 |
| 80 | 0.000000 |
| 90 | 0.000000 |
| 100 | 0.000000 |
| 200 | 0.000000 |
| 300 | 0.000000 |
| 400 | 0.000000 |
| 500 | 0.000000 |
| 600 | 0.000000 |
| 700 | 0.000000 |
| 800 | 0.000000 |
| 900 | 0.015000 |
| 1000 | 0.000000 |
| 1100 | 0.000000 |
| 1200 | 0.000000 |
| 1300 | 0.000000 |
| 1400 | 0.000000 |
| 1500 | 0.000000 |
| 1600 | 0.000000 |
| 1700 | 0.015000 |
| 1800 | 0.000000 |
| 1900 | 0.000000 |
| 2000 | 0.000000 |



3. 다음 Program 1.25는 Program 1.24의 성능측정 방법을 좀 더 정교하게 수정한 프로그램이다. 3번과 동일한 구현 조건으로 프로그램을 작성하고 그래프를 그려보시오.(3점)

```
#include <stdio.h>
#include <time.h>
#include "selectionSort.h"
#define MAX_SIZE 1001
void main(void)
{
    int i, n, step = 10;
    int a[MAX_SIZE];
    double duration;

    /* times for n = 0, 10, ..., 100, 200, ..., 1000 */
    printf("    n    repetitions    time\n");
    for (n = 0; n <= 1000; n += step)
    {
        /* get time for size n */
        long repetitions = 0;
        clock_t start = clock( );
        do
        {
            repetitions++;

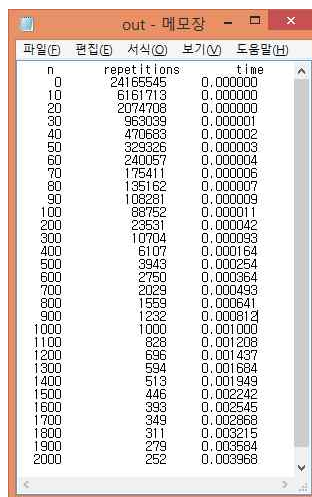
            /* initialize with worst-case data */
            for (i = 0; i < n; i++)
                a[i] = n - i;

            sort(a, n);
        } while (clock( ) - start < 1000);
        /* repeat until enough time has elapsed */

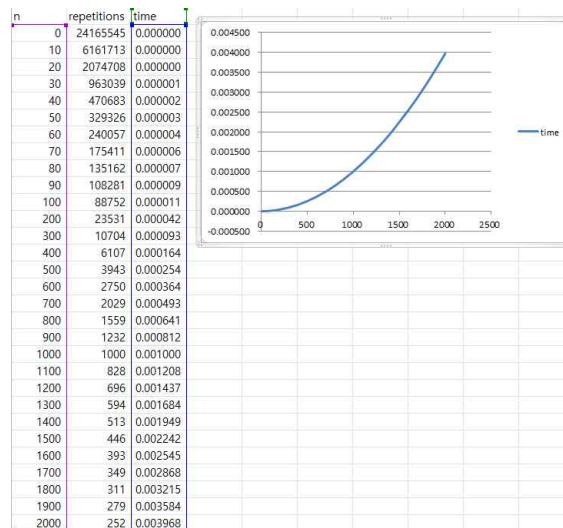
        duration = ((double) (clock() - start))
                    / CLOCKS_PER_SEC;
        duration /= repetitions;
        printf("%6d %9d %f\n", n, repetitions, duration);
        if (n == 100) step = 100;
    }
}
```

Program 1.25: More accurate timing program for selection sort

▶▶실행결과에 대한 그래프 작성 예



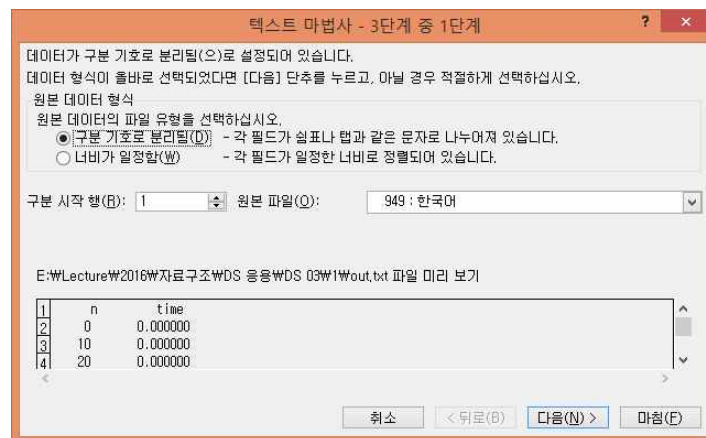
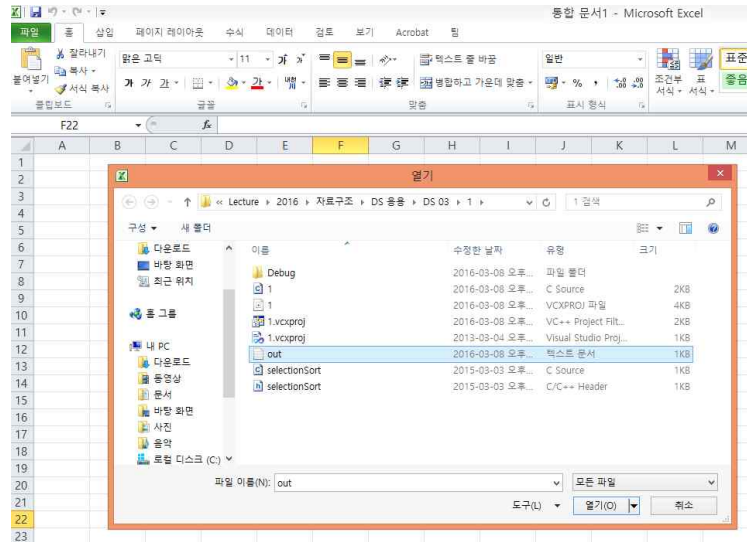
| n | repetitions | time |
|------|-------------|----------|
| 0 | 24165545 | 0.000000 |
| 10 | 6161713 | 0.000000 |
| 20 | 2074708 | 0.000000 |
| 30 | 963039 | 0.000001 |
| 40 | 470683 | 0.000002 |
| 50 | 329326 | 0.000003 |
| 60 | 240057 | 0.000004 |
| 70 | 175411 | 0.000006 |
| 80 | 135162 | 0.000007 |
| 90 | 108281 | 0.000009 |
| 100 | 88752 | 0.000011 |
| 200 | 23531 | 0.000042 |
| 300 | 10704 | 0.000083 |
| 400 | 6107 | 0.000164 |
| 500 | 3943 | 0.000254 |
| 600 | 2750 | 0.000364 |
| 700 | 2029 | 0.000493 |
| 800 | 1559 | 0.000641 |
| 900 | 1232 | 0.000812 |
| 1000 | 1000 | 0.001000 |
| 1100 | 828 | 0.001208 |
| 1200 | 696 | 0.001437 |
| 1300 | 594 | 0.001684 |
| 1400 | 513 | 0.001949 |
| 1500 | 446 | 0.002242 |
| 1600 | 393 | 0.002545 |
| 1700 | 349 | 0.002868 |
| 1800 | 311 | 0.003215 |
| 1900 | 279 | 0.003584 |
| 2000 | 252 | 0.003968 |



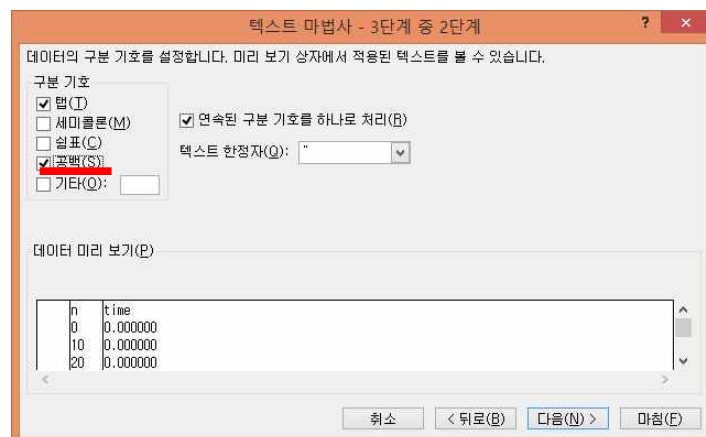
■ 참고 : 실행결과 파일로부터 표와 그래프 만들기

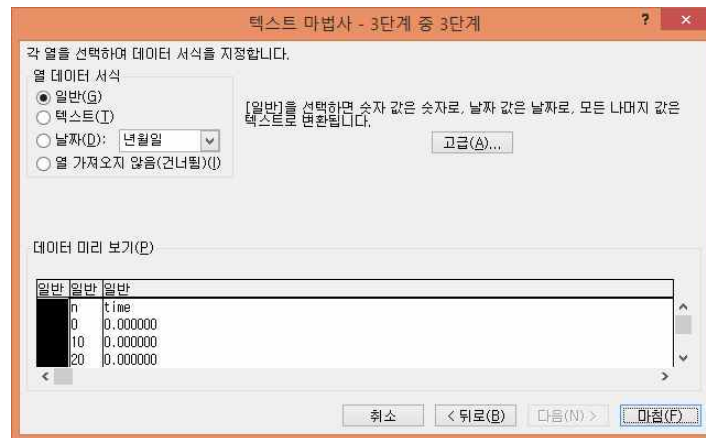
① 엑셀의 파일>열기로 출력파일(out.txt)을 연다.

(주의: 엑셀 실행 후 출력파일(out.txt)을 마우스 드래그 & 드랍으로 오픈하면 안 됨)



(※ 경우에 따라 “너비가 일정함”으로 진행되는 경우도 있음)

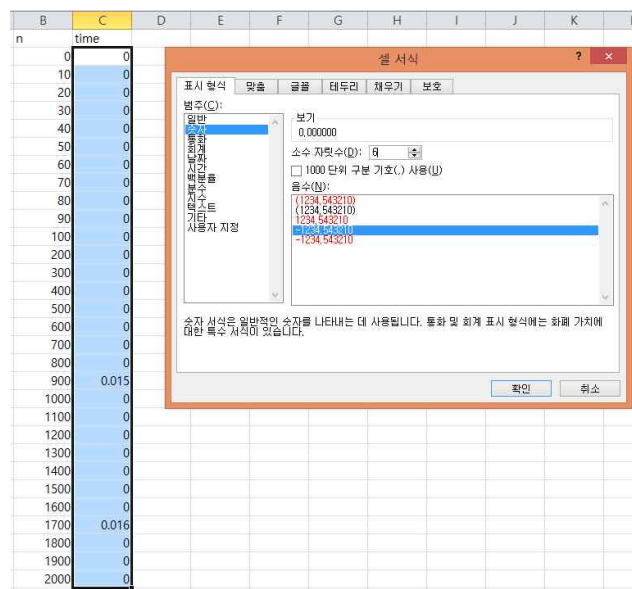




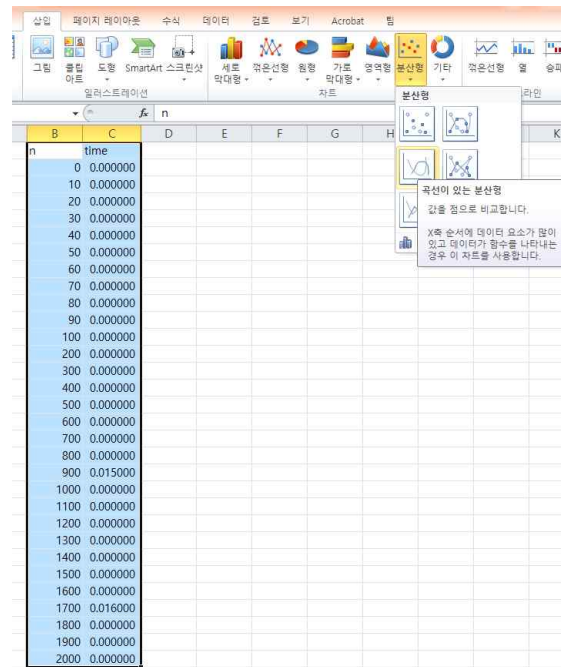
| | A | B | C |
|----|---|------|-------|
| 1 | | n | time |
| 2 | | 0 | 0 |
| 3 | | 10 | 0 |
| 4 | | 20 | 0 |
| 5 | | 30 | 0 |
| 6 | | 40 | 0 |
| 7 | | 50 | 0 |
| 8 | | 60 | 0 |
| 9 | | 70 | 0 |
| 10 | | 80 | 0 |
| 11 | | 90 | 0 |
| 12 | | 100 | 0 |
| 13 | | 200 | 0 |
| 14 | | 300 | 0 |
| 15 | | 400 | 0 |
| 16 | | 500 | 0 |
| 17 | | 600 | 0 |
| 18 | | 700 | 0 |
| 19 | | 800 | 0 |
| 20 | | 900 | 0.015 |
| 21 | | 1000 | 0 |
| 22 | | 1100 | 0 |
| 23 | | 1200 | 0 |
| 24 | | 1300 | 0 |
| 25 | | 1400 | 0 |
| 26 | | 1500 | 0 |
| 27 | | 1600 | 0 |
| 28 | | 1700 | 0.016 |
| 29 | | 1800 | 0 |
| 30 | | 1900 | 0 |
| 31 | | 2000 | 0 |

② 파일 > 다른이름으로 저장을 실행 후, Excel 통합문서 형식으로 저장한다.

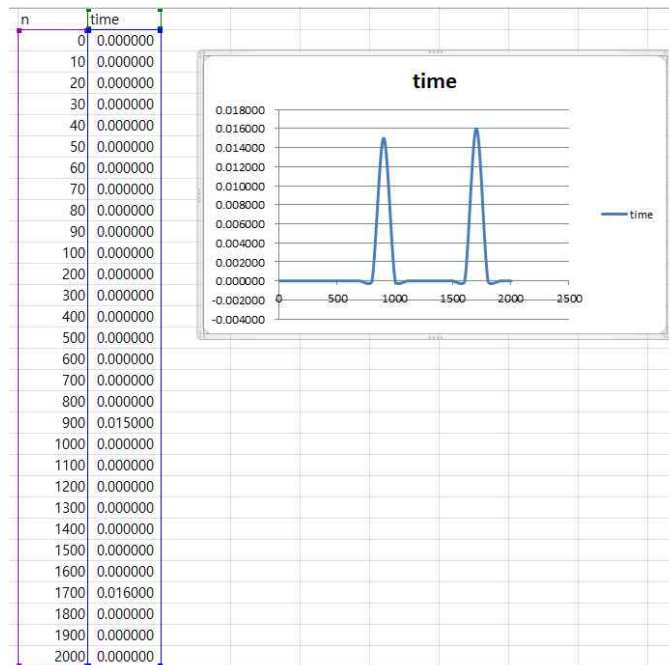
③ time 데이터의 셀서식을 “숫자, 소수 자리수 6”으로 변경한다.



④ 아래와 같이 블록을 지정한 후 “삽입 > 분산형 > 곡선이 있는 분산형” 아이콘을 선택하기



⑤ 그래프 생성 결과



■ 제출 형식

- 솔루션 이름 : DS 03
- 프로젝트 이름 : 1, 2, 3
- 각 소스파일에 주석처리
“학번 이름”

“본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다.”

- 실행화면을 캡처한 보고서를 작성 후 pdf 파일로 변환하여 솔루션 폴더에 포함
- 솔루션 정리 메뉴를 수행 후 전체 솔루션을 “학번.zip”으로 압축하여 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!
- 1차 마감 (LMS 과제 마감일) : 수업일 자정
- 2차 마감 (LMS 과제 이용 종료일) : 수업 익일 자정(만점의 50%, 반올림)
- 1, 2차 마감 이외의 제출은 허용하지 않습니다. (이메일 제출 불가!)