4일차 내용 정리

[1397] DO: [SCSA 기본C 위크샵 문제]
4명의 성적을 입력 받아 1등인 학생을 찾으시오

Language: C

```c
#include <stdio.h>

void main()
{
    int scores[4][3];
    int rows = sizeof(scores) / sizeof(scores[0]); // 4
    int colums = sizeof(scores[0]) / sizeof(int); // 3

    // 입력부
    for(int i = 0; i < rows; i++){
        for(int j = 0; j < colums; j++){
            scanf(" %d", &scores[i][j]);
        }
    }

    int idx = 0, sum, max = 0;
    // 1등 학생 검사 Logic
    for(int i = 0; i < rows; i++){
        sum = 0;
        for(int j = 0; j < colums; j++){
            sum += scores[i][j];
        }
        if(sum > max){
            max = sum;
            idx = i;
        }
    }

    printf("%d", idx);
```

배열의 기본적인 개념과
더 나아가 2차원 배열까지 다룰 수 있는 지 묻고 있다
2차원 배열에서 행과 열을 구하는 방법을 기억해두자
2차원 배열이 어렵다면 입력부와 출력부를 나눠서 연습하자
출력부는 다음장에서 확인할 수 있다
1등 학생 검사 Logic은 각 행의 누적합(sum)이 가장 큰 요소를
인덱스 정보를 나타내는 변수 idx에 저장하도록 하자
단, 성적이 같은 경우는 낮은 번호를 출력하도록 문제에서 제시하였으므로
if안에 (sum >= max)가 아닌 (sum > max)로 구현하자

4일차 내용 정리

[1397] DO: [SCSA 기본C 위크샵 문제]
4명의 성적을 입력 받아 1등인 학생을 찾으시오

```
30
31      // 출력부
32      // for(int i = 0; i < rows; i++){
33      //     for(int j = 0; j < colums; j++){
34      //         printf("%d ", scores[i][j]);
35      //     }
36      //     printf("\n");
37      // }
38
39      return 0;
40
41  }
```

2차원 배열의 행과 열을 다루는 연습을 해 두도록 하자
능숙하게 다룰 수 있다면 3차원을 넘어서도 두렵지 않다

4일차 내용 정리

[6179] D1: [SCSA 기본C 워크샵 문제]
1개의 정수를 추가하고 배열를 인쇄하시오

Language: C

```c
#include <stdio.h>

void main()
{
    int N;
    scanf("%d", &N);

    int array[N];

    for(int i = 0; i < N; i++){
        scanf(" %d", &array[i]);
    }

    int add_value, idx;
    scanf("%d", &add_value);

    for(int i = 0; i < N; i++){
        if(add_value <= array[i]){
            idx = i;
            break;
        }
    }
    for(int i = N; i >= idx; i--){
        array[i] = array[i - 1];
    }
    array[idx] = add_value;

    for(int i = 0; i < N + 1; i++){
        printf("%d ", array[i]);
    }
    return 0;
}
```

4일차 내용 정리

[6179] D1: [SCSA 기본C 워크샵 문제]
1개의 정수를 추가하고 배열를 인쇄하시오

드디어, 코드가 1페이지를 넘어갔다
위에서부터 차근차근 진행해보자
Line 15까지는 문제에서 주어진 값들을 입력받는 코드로 구성되어 있다

항상 단, <- 등의 조건이나 예외 상황을 주시하자
입력 값은 반드시 오름차순 정렬(Ascending Sort)되어 있으므로
추가 된 값의 위치를 1번의 반복문 O(n)을 통해 특정할 수 있다
특정된 인덱스를 변수 idx에 저장하자

배열에서 해당 인덱스 오른쪽 값들을 한 칸 오른쪽으로 밀고
idx에 add_value를 저장하자

출력하는 코드까지 작성해주면 완성!!

※입력 값이 오름차순이 아닐 경우에
C언어 정렬 알고리즘을 하나 이상 숙지하자

구현하기 쉽지만 느린 버블 정렬(Bubble Sort)과 선택 정렬(Selection Sort)
정도는 익혀두자

능숙해진다면, 퀵정렬(Quick Sort)에 도전해보자

4일차 내용 정리

[3527] D2:
[SCSA 7-14] 배열의 회전

Language: C

```c
#include <stdio.h>

void main()
{
    int N;
    scanf("%d", &N);

    int a[N][N];

    for(int i = 0; i < N; i++){
        for(int j = 0; j < N; j++){
            scanf(" %d", &a[i][j]);
        }
    }

    int b[N][N];

    for(int i = 0; i < N; i++){
        for(int j = 0; j < N; j++){
            b[i][j] = a[N - 1 - j][N - 1 - i];
        }
    }

    for(int i = 0; i < N; i++){
        for(int j = 0; j < N; j++){
            printf("%d ", b[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

4일차 내용 정리

[3527] D2:
[SCSA 7-14] 배열의 회전

또다시, 코드가 1페이지를 넘어갔다
2페이지가 아닌 것에 감사하자

위에서부터 차근차근 진행해 보자
Line 14까지는 문제에서 주어진 값들을 입력받는 코드로 구성되어 있다

배열의 회전에서 중요한 것은
어떤 회전 문제가 오든지 간에 적용할 수 있는 방법을 습득하는 것이다

해당 문제의 출제 의도는 2차원 배열의 index를
자유자재로 다룰 수 있는지 묻고 있다

```
b[0][0] = a[3][3];
b[0][1] = a[2][3];
b[0][2] = a[1][3];
b[0][3] = a[0][3];

b[1][0] = a[3][2];
b[1][1] = a[2][2];
b[1][2] = a[1][2];
b[1][3] = a[0][2];
```

등으로 풀어서 전개하고 반복문으로 만드는 작업을 하자
x -> y
0 -> 3
1 -> 2
2 -> 1
3 -> 0
해당 Logic은 x + y = 3을 떠올리자
앞으로 굉장히 많이 사용된다

4일차 내용 정리

[1396] D3: [SCSA 기본C 워크샵 문제]
4명의 성적을 입력 받아 합계를 계산하여 인쇄하시오

Language: C

```c
#include <stdio.h>

void main()
{
    int scores[4][3];

    int rows = sizeof(scores) / sizeof(scores[0]);
    int colums = sizeof(scores[0]) / sizeof(int);

    for(int i = 0; i < rows; i++){
        for(int j = 0; j < colums; j++){
            scanf(" %d", &scores[i][j]);
        }
    }

    int sum;
    for(int i = 0; i < rows; i++){
        sum = 0;
        for(int j = 0; j < colums; j++){
            sum += scores[i][j];
            printf("%4d ", scores[i][j]);
        }
        printf("%4d\n", sum);
    }
    return 0;
}
```

입력부는 이전 문제들에서 충분히 훈련되었을거라 생각한다
출력부에서 한 개의 행의 작업마다
해당 값들을 sum에 누적 합하는 Logic을 설계하자

배열에 저장하는 것이 아닌 출력시에만 나오는 값임을 기억하자

4일차 내용 정리

[1391] D5: [SCSA 기본C 워크샵 문제]
3*4배열의 행과 열 단위 합계를 구하시오

Language: C

```c
#include <stdio.h>

void main()
{
    int array[3][4];
    int rows = sizeof(array) / sizeof(array[0]);
    int colums = sizeof(array[0]) / sizeof(int);

    for(int i = 0; i < rows; i++){
        for(int j = 0; j < colums; j++){
            scanf(" %d", &array[i][j]);
        }
    }

    int sum;
    for(int i = 0; i < rows; i++){
        sum = 0;
        for(int j = 0; j < colums; j++){
            sum += array[i][j];
        }
        printf("%d ", sum);
    }
    printf("\n");

    for(int i = 0; i < colums; i++){
        sum = 0;
        for(int j = 0; j < rows; j++){
            sum += array[j][i];
        }
        printf("%d ", sum);
    }
    return 0;
}
```

4일차 내용 정리

[1391] D5: [SCSA 기본C 위크샵 문제]
3*4배열의 행과 열 단위 합계를 구하시오

행 단위 합계는 이전 D3 문제에서 연습해보았다
열 단위 합계에 대해 고민해보자

첫 번째 열의 합계는
array[0][0];
array[1][0];
array[2][0];

두 번째 열의 합계는
array[0][1];
array[1][1];
array[2][1];
이다

규칙을 찾아 각 열의 합계를 반복문으로 만들고
해당 반복문들을 다시 이중 반복문으로 감싸는 작업을 하자

이러한 작업들이 머릿속에서 완성될 수 있도록 연습하자

4일차 내용 정리

[6141] D6: [SCSA 기본C 워크샵 문제]
가장 큰 숫자와 가장 큰 숫자가 있는 행과 열 변호 인쇄

Language: C

```c
#include <stdio.h>

int a[5][4] = {{10,2,-3,4},
               {5,-6,7,-8},
               {-9,10,-19,12},
               {15,-8,7,-8},
               {-3,10,9,17}};

void main(void)
{
    int i, j;
    int rows = sizeof(a) / sizeof(a[0]);
    int colums = sizeof(a[0]) / sizeof(int);

    int max = a[0][0];
    int max_row = 0, max_col = 0;

    for(i = 0; i < rows; i++){
        for(int j = 0; j < colums; j++){
            if(a[i][j] > max){
                max = a[i][j];
                max_row = i;
                max_col = j;
            }
        }
    }
    printf("%d %d %d", max_row, max_col, max);
    return 0;
}
```

모든 요소를 탐색하며, 해당 값이 기존에 저장된 최댓값 max보다 크다면, 해당 행을 max_row, 해당 열을 max_col, 해당 값을 max에 저장하는 Logic을 설계해보자

4일차 내용 정리

[6140] D7: [SCSA 기본C 워크샵 문제]
같은 모양 찾기 simple

Language: C

```c
#include <stdio.h>

int main(void)
{
    // 입력부
    int M;
    scanf("%d", &M);
    int graph[M][M];

    for(int i = 0; i < M; i++){
        for(int j = 0; j < M; j++){
            scanf("%1d", &graph[i][j]);
        }
    }

    int P;
    scanf("%d", &P);
    int Pattern[P][P];

    for(int i = 0; i < P; i++){
        for(int j = 0; j < P; j++){
            scanf("%1d", &Pattern[i][j]);
        }
    }
```

모눈종이의 크기를 M, 모눈종이 정보를 graph에
패턴의 크기를 P, 패턴의 정보를 pattern에 저장하는 입력부 Logic을 구현하자

4일차 내용 정리

[6140] D7: [SCSA 기본C 워크샵 문제]
같은 모양 찾기 simple

```
24      }
25
26      // 판단 Logic
27      int count = 0;
28      for(int i = 0; i < M - P + 1; i++){
29          for(int j = 0; j < M - P + 1; j++){
30              if(graph[i][j] == pattern[0][0]){
31
32                  int check = 0;
33                  for(int k = 0; k < P; k++){
34                      if(check == 1){
35                          break;
36                      }
37                      for(int l = 0; l < P; l++){
38                          if(graph[i + k][j + l] == pattern[k][l]){
39                              continue;
40                          }else{
41                              check = 1;
42                              break;
43                          }
44                      }
45                  }
46                  if(check == 0){
47                      count++;
48                  }
49              }
50          }
51      }
52      printf("%d", count);
53      return 0;
54  }
```

우리는 할 수 있다
먼저 검사해야 하는 범위 설정을 해주자
모눈종이의 크기가 10칸이고 패턴이 3칸이면
1칸부터 8칸까지만 검사하면 된다(9, 10칸은 검사해도 크기가 안 나오므로)
따라서 M - P + 1번째 요소까지만 확인하자

이후, 모눈종이 모든 요소에 대해
패턴 첫 번째 칸(pattern[0][0])과 일치하는지 확인하자

일치한다면, 모든 패턴의 값과 확인하는 절차를 거치자
일치하지 않다면, check값을 1로 바꿔주는 작업 등을 통해 개수를 정확히 세어보자

4일차 내용 정리

[6143] D8: [SCSA 9-15]

음료수 자판기

Language: C

```c
#include <stdio.h>

char* Vending_Machine(int num)  // 함수 리턴 프로토타입 완성
{
    static char drink[4][10] = { "cola", "milk", "coffee", "wine" };
    return drink[num];
}

void main(void)
{
    // 코드 작성
    int num;
    scanf("%d", &num);
    printf("%s", Vending_Machine(num));
    return 0;
}
```

문자열을 리턴 타입으로 지정할 수 있는지 묻고 있다
정수를 입력 받는 scanf함수를 완성하도록 하자

4일차 내용 정리

[6145] D9: [SCSA 9-18]
문자열 길이 측정

Language: C

```c
#include <stdio.h>

unsigned int str_length(const char* d)
{
    int cnt = 0;
    while(*d){
        *d++;
        cnt++;
    }
    return cnt;
}

void main(void)
{
    char a[] = "Willtek";

    printf("%d\n", sizeof(a));
    printf("%d\n", str_length(a));
}
```

포인터의 개념에 대해 질문하고 있다
문제에 있는 a[] 형식처럼 문자 배열로 저장된 경우
문자열의 길이는 단순히 sizeof값에서 1을 빼면 된다
(※널 문자 ₩00이 포함되어 있으므로)

하지만, 포인터나 동적 메모리 할당 방식으로 저장된 문자열은
포인터의 크기를 반환하기 때문에 1을 빼는 방식으로 크기를 구할 수 없다
따라서 반복 횟수를 알 수 없는 반복문을 통하여 문자열을 크기를 구해보자