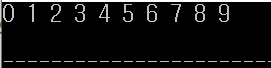
|  |
| --- |
| 2020년 1학기 |
| 프로그래밍 언어 레포트 |
| 11주차 학습활동 |

201721899 사회학과

송지민

**[과제#1] 실습#1, #2, #3 실습수행 및 실행화면 캡처**

**1. [실습#1] 선택정렬 코드 및 실행화면**

#include <stdio.h>

#define SIZE 10

int main(void)

{

int list[SIZE] = {3, 2, 9, 7, 1, 4, 8, 0, 6, 5};

int i, j, temp, least;

for(i = 0; i < SIZE-1; i++)

{

least = i;

for(j = i+1; j < SIZE; j++)

{

if(list[j] < list[least])

least = j;

}

temp = list[i];

list[i] = list[least];

list[least] = temp;

}

for(i = 0; i < SIZE; i++)

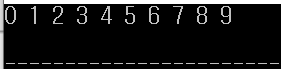
printf("%d ", list[i]);

printf("\n");

return 0;

}

=> 다른 방식으로 작성한 코드(반복문을 통해 index를 바꾸는 것이 아니라 배열값을 서로 바꿈)



#include <stdio.h>

#define SIZE 10

int main(void)

{

int list[SIZE] = {3, 2, 9, 7, 1, 4, 8, 0, 6, 5};

int i, j, temp, least;

for(i = 0; i < SIZE; i++)

{

for(j = i + 1; j < SIZE; j++)

{

if(list[i] > list[j])

{

temp = list[i];

list[i] = list[j];

list[j] = temp;

}

}

printf("%d ", list[i]);

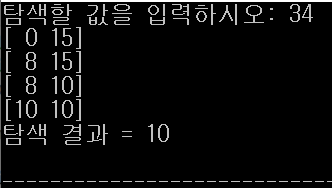
}

printf("\n");

return 0;

}

**2. [실습#2] 이진탐색 코드 및 실행화면**



#include <stdio.h>

#define SIZE 16

int binary\_search(int list[], int n, int key);

int main(void)

{

int key = 0;

int grade[SIZE] = {2, 6, 11, 13, 18, 20, 22, 27, 29, 30, 34, 38, 41, 42, 45, 47};

printf("탐색할 값을 입력하시오: ");

scanf("%d", &key);

printf("탐색 결과 = %d\n", binary\_search(grade, SIZE, key));

return 0;

}

int binary\_search(int list[], int n, int key)

{

int low, high, middle;

low = 0;

high = n-1;

while(low <= high){

printf("[%2d %2d]\n", low, high);

middle = (low + high)/2;

if(key == list[middle])

return middle;

else if(key > list[middle])

low = middle + 1;

else

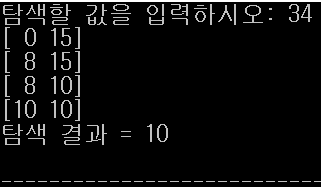
high = middle - 1;

}

return -1;

}

=> 정렬되지 않은 배열을 정렬하고 이진탐색 하는 코드

#include <stdio.h>

#define SIZE 16

int binary\_search(int list[], int n, int key);

int main(void)

{

int key = 0;

int i, j, temp;

int grade[SIZE] = {34, 11, 20, 41, 45, 13, 2, 6, 38, 29, 42, 47, 22, 27, 18, 30};

printf("탐색할 값을 입력하시오: ");

scanf("%d", &key);

for(i = 0; i < SIZE - 1; i++)

for(j = i+1; j < SIZE; j++)

{

if(grade[i] > grade[j])

{

temp = grade[i];

grade[i] = grade[j];

grade[j] = temp;

}

}

printf("탐색 결과 = %d\n", binary\_search(grade, SIZE, key));

return 0;

}

int binary\_search(int list[], int n, int key)

{

int low, high, middle;

low = 0;

high = n-1;

while(low <= high){

printf("[%2d %2d]\n", low, high);

middle = (low + high)/2;

if(key == list[middle])

return middle;

else if(key > list[middle])

low = middle + 1;

else

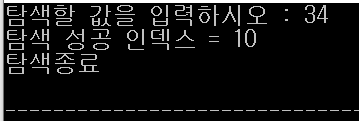
high = middle - 1;

}

return -1;

}

=> 순차 탐색으로 찾아보기

#include <stdio.h>

#define SIZE 16

int main(void)

{

int key, i;

int list[SIZE] = {2, 6, 11, 13, 18, 20, 22, 27, 29, 30, 34, 38, 41, 42, 45, 47};

printf("탐색할 값을 입력하시오 : ");

scanf("%d", &key);

for(i = 0; i < SIZE; i++)

{

if(key == list[i])

printf("탐색 성공 인덱스 = %d\n", i);

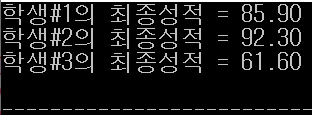
}

printf("탐색종료\n");

return 0;

}

**3. [실습#3] 2차원 배열 예제 코드 및 실행화면**

#include <stdio.h>

#define ROWS 3

#define COLS 5

int main(void)

{

int a[ROWS][COLS] = {{87, 98, 80, 76, 3},

{99, 89, 90, 90, 0},

{65, 68, 50, 49, 0}};

int i;

for(i = 0; i < ROWS; i++)

{

double final\_scores = a[i][0]\*0.3 + a[i][1]\*0.4 +

a[i][2]\*0.2 + a[i][3]\*0.1 -a[i][4];

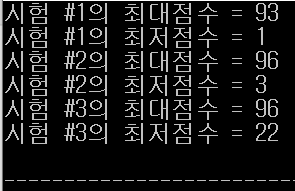
printf("학생#%d의 최종성적 = %5.2f\n", i+1, final\_scores);

}

return 0;

}

**[과제#2] 시험점수 통계처리 프로그램**

**1. 코드 및 실행화면**

// 시험점수 통계처리 프로그램

#include <stdio.h> // 표준입출력 헤더파일 선언

#include <stdlib.h> // 표준라이브러리 헤더파일 선언

// 랜덤함수 사용하여 난수 받기 위함

void get\_stat\_per\_test(int score[][3]) // 반환값이 없으며 매개변수로 배열을 받는 함수

/\* 2차원 배열은 선형구조인 1차원 배열과 달리 행과 열을 지닌 배열이다.

자료형 배열명[행의 개수][열의 개수] 형태로 선언하며 1차원 배열과 마찬가지로 초기화하지 않으면 기본값 0이 저장된다.

함수 호출 시 인수를 입력하면 사본이 전달되었던 변수와 달리 배열은 원본이 전달된다.

따라서 함수에서 처리를 통해 배열의 값이 바뀌면 함수 코드 실행이 끝나도 배열에 바뀐 값이 저장되어 있다.

또한 매개변수를 선언할 때 실제로 배열이 생성되는 것이 아니기 때문에,

배열을 받는 매개변수는 배열의 크기를 적어주지 않아도 된다.\*/

{

// 변수 선언

int i, k, min, max, sum; // 인덱스 변수, 최대 최저값 처리 변수, 값 교환을 위한 변수

// 최대점수, 최소점수 구하기

for(i = 0; i < 3; i++){ // 0부터 2까지 3번 반복(열의 개수만큼)

min = score[0][i]; // 최저점수를 제일 첫번째 행의 값이라고 가정

max = score[9][i]; // 최대점수를 제일 마지막 행의 값이라고 가정

// 최대점수와 최저점수는 각 시험마다 출력해야 하므로 외부 반복문에 작성

for(k = 0; k < 9; k++){ // 0부터 8까지 9번 반복(10개를 비교하므로)

if(score[k][i] < min){ // 만약 최저점수가 더 크면(작게 만들어주기)

sum = min; // 최저점수라고 가정한 값을 복사해 둠

min = score[k][i]; // 최저점수에 작은 값을 저장

score[k][i] = sum; // 복사해 둔 값을 더 커야 하는 배열인덱스에 저장

}

if(score[k][i] > max){ // 만약 최대점수가 더 작다면(크게 만들어주기)

sum = max; // 최대점수라고 가정한 값을 복사해 둠

max = score[k][i]; // 최대점수에 큰 값을 저장

score[k][i] = sum; // 복사해둔 값을 더 작아야 하는 배열인덱스에 저장

}

}

/\* swap 함수를 이용하여 배열의 값을 교환하였으며 이것을 반복하면

제일 작은 수는 min에 제일 큰 수는 max에 저장된다.\*/

printf("시험 #%d의 최대점수 = %d\n", i+1, max); // 최대점수 출력

printf("시험 #%d의 최저점수 = %d\n", i+1, min); // 최저점수 출력

/\* 마찬가지로 열마다 각각 한 번씩 출력해야 하므로 외부 반복문에 작성하며,

인덱스는 0부터 시작하기 때문에 +1을 해주어야 1부터 3까지로 출력된다.\*/

}

// 반환값이 없으므로 return 값을 주지 않아도 됨

}

int main(void) // 반환형이 정수형이며 매개변수가 없는 메인 함수 선언부

{

int i, k; // 2개의 인덱스 변수

int score[10][3]; // 행이 10개고 열이 3개인 2차원 배열 선언

// 반복문 통해 시험 3개에 대한 학생 10명의 점수 받기(30번 반복)

for(i = 0; i < 10; i++){ // 0부터 9까지 10번 반복

for(k = 0; k < 3; k++){ // 0부터 2까지 3번 반복

score[i][k] = rand()%100 + 1; // 2차원 배열에 1부터 100까지의 난수 저장

// 0행의 0열부터 차례대로 점수가 저장됨

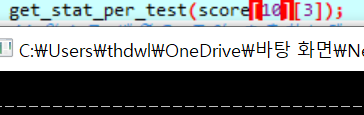
// 0행의 2열까지 저장을 마치면 외부 반복문으로 나가서 1행의 0열 저장 시작

}

}

get\_stat\_per\_test(score); // 각 시험의 최대, 최저 점수 판별하는 함수 호출

// 인수로 배열을 주어 호출하는데, 이 때 배열의 자료형이나 크기를 입력하면 오류가 발생함



return 0; // 0 반환하여 끝내기

}

**2. 학생의 시험 점수를 확인하기 위해 시험 점수까지 출력(코드 다른 부분만 주석처리)**

#include <stdio.h>

#include <stdlib.h>

void get\_stat\_per\_test(int score[10][3])

{

int i, k, min, max, sum;

for(i = 0; i < 3; i++){

min = score[0][i];

max = score[9][i];

for(k = 0; k < 9; k++){

if(score[k][i] < min){

sum = min;

min = score[k][i];

score[k][i] = sum;

}

if(score[k][i] > max){

sum = max;

max = score[k][i];

score[k][i] = sum;

}

}

printf("시험 #%d의 최대점수 = %d\n", i+1, max);

printf("시험 #%d의 최저점수 = %d\n", i+1, min);

}

return 0;

}

int main(void)

{

int i, k;

int score[10][3];

printf(" 시험1 시험2 시험3\n"); // 행에 해당하는 시험1,2,3 표시

for(i = 0; i < 10; i++){ // 0부터 9까지 10번 반복

printf("%2d번 학생 : ", i+1); // 행마다 학생번호 표시

for(k = 0; k < 3; k++){ // 0부터 2까지 3번 반복

score[i][k] = rand()%100 + 1; // 배열에 난수 저장

printf("%4d ", score[i][k]); // 저장한 난수를 출력

}

printf("\n"); // 시험 1,2,3에 대한 i번 학생 점수 출력하고 다음 학생 점수 출력할 때 줄 바꿈

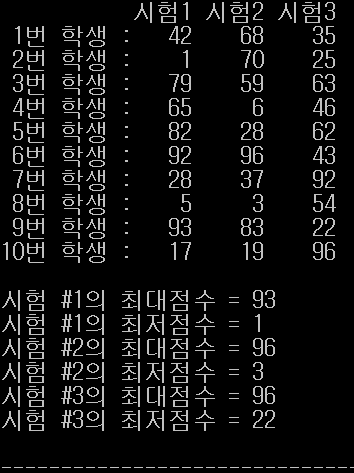
}

printf("\n"); // 각 시험의 최대, 최저 점수 출력 전 줄바꿈

get\_stat\_per\_test(score);

return 0;

}



(실행화면에서 표시된 값이 강의 노트의 출력 값과 달라 난수를 이용해 받은 시험 점수를 전부 출력해 보았습니다.

그러나 실행화면을 보면 시험1과 시험2의 최대점수가 93, 96으로 보이기에, 오류는 아닌 것으로 생각됩니다.)

**3. 최저, 최대 점수 처리 및 출력하는 함수 부분만 변경**

#include <stdio.h>

void get\_stat\_per\_test(int score[][3])

{

int i, k, min, max, sum;

for(k = 0; k < 3; k++)

{

min = max = score[0][k]; // 최솟값, 최댓값 모두 행이 0일 때라고 가정

sum = 0; // 합계를 0으로 초기화

for(i = 0; i < 10; i++) // 0부터 9까지 10번 반복

{

if(score[i][k] < min) // 만약 배열값이 최소라고 가정한 값보다 작으면

min = score[i][k]; // 최솟값에 배열값 넣어주기

if(score[i][k] > max) // 만약 배열값이 최대라고 가정한 값보다 크면

max = score[i][k]; // 최댓값에 배열값 넣어주기

sum += score[i][k]; // 누적하여 더하기

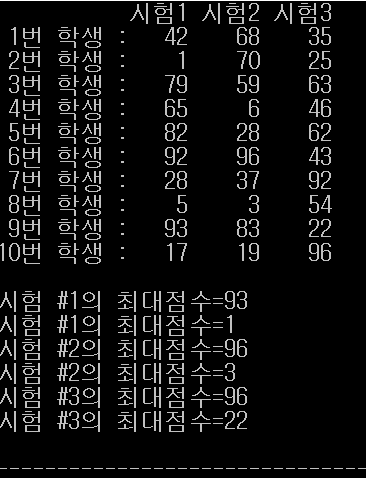
}

printf("시험 #%d의 최대점수=%d\n", k+1, max); // 최댓값 출력

printf("시험 #%d의 최대점수=%d\n", k+1, min); // 최솟값 출력

}

}

int main(void)

{

int i, k;

int score[10][3];

printf(" 시험1 시험2 시험3\n");

for(i = 0; i < 10; i++){

printf("%2d번 학생 : ", i+1);

for(k = 0; k < 3; k++){

score[i][k] = rand()%100 + 1;

printf("%4d ", score[i][k]);

}

printf("\n");

}

printf("\n");

get\_stat\_per\_test(score);

return 0;

}

**[과제#3] 배열과 정렬을 이용한 성적처리 프로그램**

**1. 코드 및 실행화면**

// 배열과 정렬을 이용한 성적처리 프로그램

#include <stdio.h> // 표준입출력 헤더파일 선언

#include <stdlib.h> // 표준라이브러리 헤더파일 선언(난수 받기 위해 선언)

#define STUDENT 5 // 배열의 크기 선언(학생의 수)

#define COLS 6 // 배열의 크기 선언(평가항목의 수+총점)

/\* 배열의 크기를 기호상수로 선언하면 반복문에서 반복하는 횟수 등,

배열의 크기가 할당되는 곳의 값이 기호상수를 바꾸는 것만으로 변경되므로 코드 수정이 용이하다.\*/

int main(void) // 반환값이 정수형이며 매개변수가 없는 메인 함수의 선언부

{

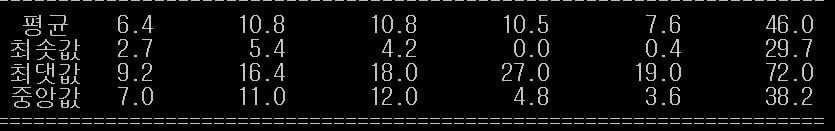
// 변수 선언

int i, j, k; // 정수형 변수 선언(인덱스 변수)

double avg, sum, tmp; // 실수형 변수 선언

// tmp는 실수끼리의 swap 함수의 변수이므로 실수로 선언해야 함

// 정수로 선언했을 경우 값을 바꾸는 과정에서 소수점이 소실(실행화면첨부)



// 배열 선언

int score[STUDENT][COLS]; // 환산하지 않은 점수 저장할 배열

double score2[STUDENT][COLS]; // 환산한 점수 저장할 배열

int rank[STUDENT]; // 순위 매기기 위한 배열

/\* 평가항목 당 점수를 받는 배열은 2차원 배열로, 총점, 열마다 정렬(하나의 열에 한하여 계산하므로), 순위(1부터5까지 선형적이므로)를 정하는 배열은 1차원 배열로 선언하였다.\*/

// 평가항목 점수와 총점을 학생별로 출력

printf("\n=====================================================================\n");

printf(" 출석점수 중간고사 기말고사 과제점수 수시평가 총점 \n");

printf("---------------------------------------------------------------------\n");

// 평가항목 5가지와 총점이 열이 됨

// 학생별로 평가항목 5가지와 총점 계산 및 출력하는 반복문

for(i = 0; i < STUDENT; i++) // 0부터 4까지, 학생수만큼 반복

{

sum = 0; // 총점 0으로 만들어 줌

// 학생별로 총점을 구하고 다른 학생 총점을 구할 때는 0으로 만들어야 하므로 외부 반복문에서 실시

printf(" %d번 ", i+1); // 학번 출력(학번은 인덱스+1 해야 함)

for(k = 0; k < COLS; k++) // 0부터 5까지 6번 반복

{

score[i][k] = rand()%100; // 랜덤함수를 이용하여 난수를 받아 저장하기

// 점수가 0부터 99점까지이므로 100으로 나눈 나머지를 넣어 줌

// 환산점수 구하기

// 평가항목별로 반영비율이 다르므로 경우를 고려해야 함

if(k >= 0 && k < COLS-1)

{

if(k == 0) // 0열일 때, 즉 출석점수(비율10%)일 때

score2[i][k] = score[i][k]\*0.1; // 난수로 받은 점수에 0.1을 곱하여 환산점수 배열에 저장

else if(k == 1 || k == 2 || k == 4) // 중간, 기말고사, 수시평가(비율20%)일 때

score2[i][k] = score[i][k]\*0.2; // 난수로 받은 점수에 0.2를 곱하여 환산점수 배열에 저장

else // 과제점수(30%)일 때

score2[i][k] = score[i][k]\*0.3; // 난수로 받은 점수에 0.3을 곱하여 환산점수 배열에 저장

// 학생별로 환산점수 총점 구하기

sum += score2[i][k]; // 환산점수를 횡으로 더해줌

}

else // 열이 5일 때(총점)

score2[i][k] = sum; // 앞서 누적하여 합한 값을 저장

printf(" %4.1lf ", score2[i][k]);

// 환산점수를 출력(소수점을 포함하여 4칸, 소수점 첫째자리까지 출력)

}

printf("\n"); // 행마다 줄바꿈하여 다음 행에서 다음 학생의 점수를 출력하게 함

}

printf("---------------------------------------------------------------------\n");

// 환산점수 열마다 평균 구하는 함수

printf(" 평균 "); // 평균 출력 행임을 표시

avg = 0; // 평균 0으로 초기화

// 평균을 구하고 출력하는 반복문

for(i = 0; i < COLS; i++) // 0부터 5까지 6개의 평가항목에 대하여 계산 및 출력

{

sum = 0; // 총점에 0입력(각 항목의 총점 구할 때마다 0으로 초기화해줌)

// 이 과정을 거치지 않으면 이전 과정에서 저장된 값에 추가하여 계산하므로 오류가 발생

for(k = 0; k < STUDENT; k++) // 0부터 4까지 학생수만큼 반복

{

sum += score2[k][i]; // 열의 행을 누적하여 더함

}

avg = sum/5.0; // 누적된 열별 환산점수를 5.0으로 나눔

// 실수/실수 형태로 나누지 않으면 정수가 나오므로 반드시 5.0처럼 실수표기 해주어야 함

printf(" %4.1lf ", avg); // 각 열의 총점 평균을 출력(4칸, 소수점 첫째자리까지)

}

printf("\n"); // 줄바꿈

// 순위 정하기

for(i = 0; i < STUDENT; i++) // 0부터 4까지 총 5번 반복

{

rank[i] = 1; // 순위에 1 저장

// 각 학생마다 다른 학생의 총점과 비교하여 순위가 결정되므로 1로 초기화

for(k = 0; k < STUDENT; k++) // 0부터 4까지 5번 반복

{

if(score2[i][5] < score2[k][5]) // 5열에 대해서 i행의 값이 더 작으면

rank[i]++; // 순위 1 증가

}

} /\* 순위 정하기는 각 학번 배열에 담긴 값은 유지한 상태에서 진행해야 하므로 정렬하기 전에 실시해야 한다.

다만, 출력은 따로 이루어지기 때문에 우선적으로 배열을 이용하여 환산점수를 정렬하기 전에 순위를 저장해 놓는다.\*/

// 최솟값 출력 안내하기

printf(" 최솟값 ");

// 2차원 배열 열의 행을 정렬하기 위한 반복문

for(i = 0; i < COLS; i++) // 0부터 5까지 총 6번 반복

{

for(k = 0; k < STUDENT; k++) // 0부터 4까지 5번 반복

{

// i번째 열의 k번째 행과 k+1번째 행을 비교하여 정렬

for(j = k+1; j < STUDENT; j++) // k+1부터 4까지, k보다 인덱스가 높은 값에 대해 반복

{

if(score2[k][i] > score2[j][i]) // 만약 인덱스가 작은 배열값이 더 크다면

{

tmp = score2[k][i]; // 큰 값을 저장

score2[k][i] = score2[j][i]; // 인덱스가 작은 배열에 작은 값을 저장

score2[j][i] = tmp; // 인덱스가 큰 배열에 큰 값을 저장

// swap함수를 이용하여 오름차순으로 정렬함

// 부등호를 반대로 하면 내림차순으로 정렬이 가능

}

}

}

printf(" %4.1lf ", score2[0][i]); // 최솟값 출력하기

} // 작은 수부터 정렬했으므로 첫번째 인덱스 배열값이 최소가 됨

printf("\n"); // 줄바꿈

printf(" 최댓값 "); // 최댓값 출력 안내

for(i = 0; i < COLS; i++) // 0부터 5까지 6번 반복

{

printf(" %4.1lf ", score2[STUDENT-1][i]); // 최대 환산점수를 출력

// 최대 환산점수는 인덱스가 가장 높은 배열값이므로 배열 크기에 -1해줌

}

printf("\n 중앙값 "); // 중앙값 출력 안내

for(i = 0; i < COLS; i++) // 0부터 5까지 6번 반복

{

printf(" %4.1lf ", score2[STUDENT/2][i]);

} // 중앙값이므로 행에 학생수/2 인덱스를 주고 열은 i

printf("\n=====================================================================\n");

// 학생별로 총점, 각 열의 평균 출력 완료

printf("\n"); // 줄바꿈

printf("=============\n"); // 총점별로 순위 매기기

printf(" 순위\n"); // 순위 출력 안내

printf("-------------\n"); // 구분선

// 순위 출력하는 반복문

// 순위의 경우 총점까지 다 받은 다음에 처리가 이루어져야 하므로 처리 및 출력을 따로 구성함

for(i = 0; i < STUDENT; i++)

{

printf("%d번 ", i+1); // 학번 출력, 인덱스는 0부터이므로 +1함

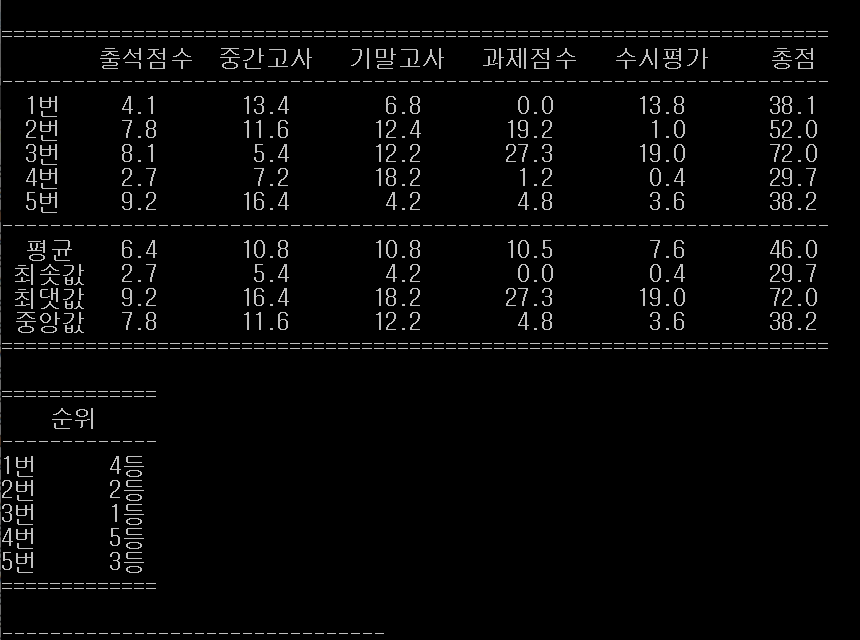
printf(" %2d등\n", rank[i]); // 등수 출력하기

}

printf("=============\n"); // 순위 출력 완료

return 0; // 0을 반환하여 끝내기

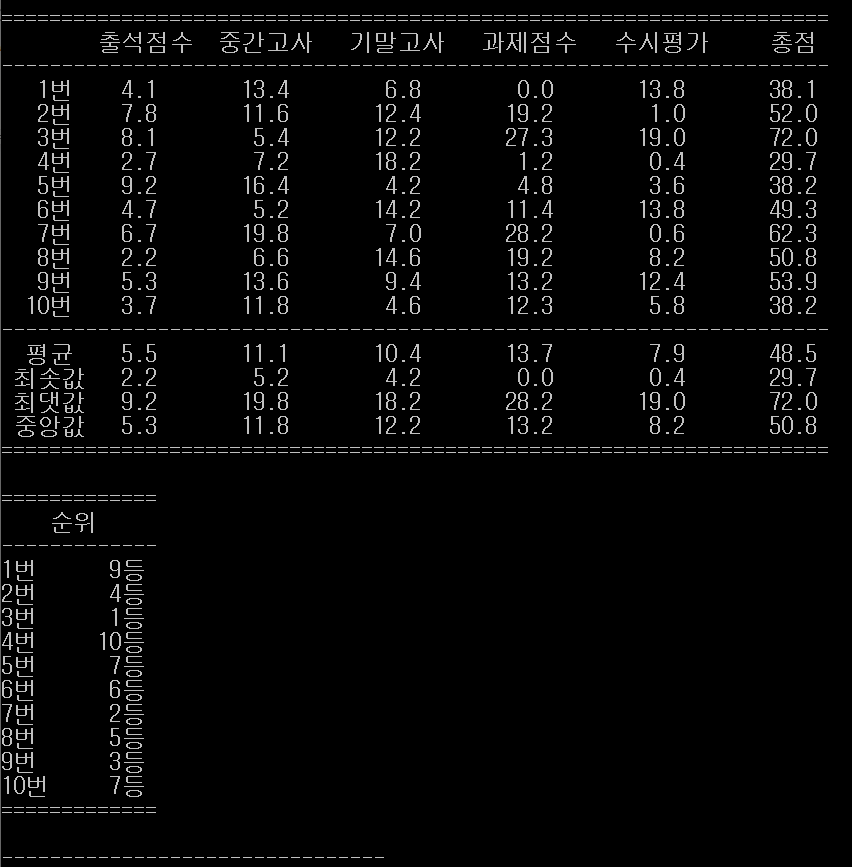
}



**2. 학생 수를 10명으로 바꾸어 출력함**

- #define STUDENT 5 -> #define STUDENT 10으로 수정

- 평균 부분에서 5.0으로 나누었던 것을 10.0으로 수정

- 학번 출력 칸 수정

**3. 출력형태를 다르게 하여 나타내기(총점을 따로 표시, 코드 다른 부분 위주로 주석처리)**

// 배열과 정렬을 이용한 성적처리 프로그램2

#include <stdio.h>

#include <stdlib.h>

#define STUDENT 5

#define COLS 5 // 배열의 크기 선언(평가항목의 수)

int main(void)

{

// 변수 선언

int i, j, k;

double avg, sum, tmp;

// 배열 선언

int score[STUDENT][COLS];

double score2[STUDENT][COLS];

double arr\_sum[STUDENT]; // 총점 저장할 배열

// 열이 평가항목까지만 포함하도록 크기 5로 선언하여 총점을 따로 저장함

int rank[STUDENT];

// 평가항목 점수와 총점을 학생별로 출력

printf("\n=====================================================================\n");

printf(" 출석점수 중간고사 기말고사 과제점수 수시평가 총점 \n");

printf("---------------------------------------------------------------------\n");

// 학생별로 평가항목 5가지와 총점 계산 및 출력하는 반복문

for(i = 0; i < STUDENT; i++)

{

sum = 0;

printf(" %d번 ", i+1);

for(k = 0; k < COLS; k++) // 0부터 4까지 5번 반복

{

score[i][k] = rand()%100;

// 환산점수 구하기

if(k == 0) // 출석점수(비율10%)일 때

score2[i][k] = score[i][k]\*0.1;

else if(k == 1 || k == 2 || k == 4)

// 중간, 기말고사, 수시평가(비율20%)일 때

score2[i][k] = score[i][k]\*0.2;

else // 과제점수(30%)일 때

score2[i][k] = score[i][k]\*0.3;

printf(" %4.1lf ", score2[i][k]);

// 환산점수를 출력(소수점을 포함하여 4칸, 소수점 첫째자리까지 출력)

// 학생별로 환산점수 총점 구하기

sum += score2[i][k]; // 환산점수를 횡으로 더해줌

}

arr\_sum[i] = sum; // 환산점수 총점 배열에 저장

printf(" %4.1lf ", arr\_sum[i]); // 총점 출력(소수점을 포함하여 4칸, 소수점 첫째자리까지 출력)

printf("\n");

}

printf("---------------------------------------------------------------------\n");

// 환산점수 열마다 평균 구하는 함수

printf(" 평균 "); // 평균 출력 행임을 표시

avg = 0; // 평균 0으로 초기화

// 평균을 구하고 출력하는 반복문

for(i = 0; i < COLS; i++) // 0부터 4까지 5개의 평가항목에 대하여 계산 및 출력

{

sum = 0;

for(k = 0; k < STUDENT; k++)

{

sum += score2[k][i];

}

avg = sum/5.0;

printf(" %4.1lf ", avg);

}

printf("\n");

printf(" 최솟값 ");

// 열의 행값 정렬하기

for(i = 0; i < COLS; i++) // 0부터 4까지 5번 반복

{

for(k = 0; k < STUDENT; k++) // 0부터 4까지 5번 반복

{

for(j = k+1; j < STUDENT; j++) // 정렬 위한 반복문

// 행의 인덱스가 k보다 큰수부터 4까지인 배열값을 인덱스가 k인 배열값과 비교

{

if(score2[k][i] > score2[j][i]) // 행의 인덱스가 작은 배열의 배열값이 더 클 때

{

tmp = score2[k][i];

score2[k][i] = score2[j][i];

score2[j][i] = tmp;

// 배열값을 바꿔 줌

}

}

}

printf(" %4.1lf ", score2[0][i]); // 최솟값 출력

}

printf("\n");

printf(" 최댓값 ");

for(i = 0; i < STUDENT; i++)

{

printf(" %4.1lf ", score2[STUDENT-1][i]);

}

printf("\n 중앙값 ");

for(i = 0; i < STUDENT; i++)

{

printf(" %4.1lf ", score2[STUDENT/2][i]);

}

printf("\n=====================================================================\n");

printf("\n");

printf("=============\n"); // 총점별로 순위 매기기

printf(" 순위\n"); // 순위 출력 안내

printf("-------------\n"); // 구분선

// 순위 처리하고 출력하는 반복문

// 환산점수와 총점의 배열을 분리했으므로 환산점수 정렬 이후 총점 순위 매기기 가능

for(i = 0; i < STUDENT; i++)

{

printf("%d번 ", i+1);

rank[i] = 1;

for(k = 0; k < STUDENT; k++)

{

if(arr\_sum[i] < arr\_sum[k]) // 총점 배열을 비교하여 순위매김

rank[i]++;

}

printf(" %2d등\n", rank[i]);

}

printf("=============\n");

printf("\n");

sum = 0; // 총점 값 계산하기 위해 합계 0으로 초기화

// 총점 정렬하기 위한 반복문

// 총점은 앞서 1차원 배열로 저장하였기 때문에 1차원 배열 정렬을 이용

for(i = 0; i < STUDENT; i++) // 0부터 4까지 학생 수만큼 반복

{

for(k = i+1; k < STUDENT; k++) // i보다 1큰 수부터 4까지 반복

{

if(arr\_sum[i] > arr\_sum[k]) // 만약 인덱스가 작은 배열값이 더 작으면

{

tmp = arr\_sum[i]; // tmp에 큰 값을 저장

arr\_sum[i] = arr\_sum[k]; // 인덱스가 작은 배열에 작은 값을 저장

arr\_sum[k] = tmp; // 인덱스가 큰 배열에 큰 값을 저장

// swap함수를 이용하여 값을 바꿔줌

}

}

sum += arr\_sum[i]; // 총점의 평균 구하기 위해 총점을 누적하여 합함

}

avg = sum/5.0; // 총점의 평균 구하기 위해 누적합계를 5.0으로 나눔

// sum/5의 형태로 계산하면 정수형이 되므로 5.0으로 나눠야 함

// 총점의 평균, 최솟값, 최댓값, 중앙값 출력하기

printf("=============\n"); // 총점 값 출력 구분

printf(" 총점\n"); // 총점 출력 안내

printf("-------------\n"); // 총점 값 출력 시작 구분

printf("평균 : %4.1lf\n", avg); // 총점 평균 출력하기

printf("최소 : %4.1lf\n", arr\_sum[0]); // 인덱스 0 배열값 출력하여 최솟값 나타내기

printf("최대 : %4.1lf\n", arr\_sum[4]); // 인덱스 4 배열값 출력하여 최댓값 나타내기

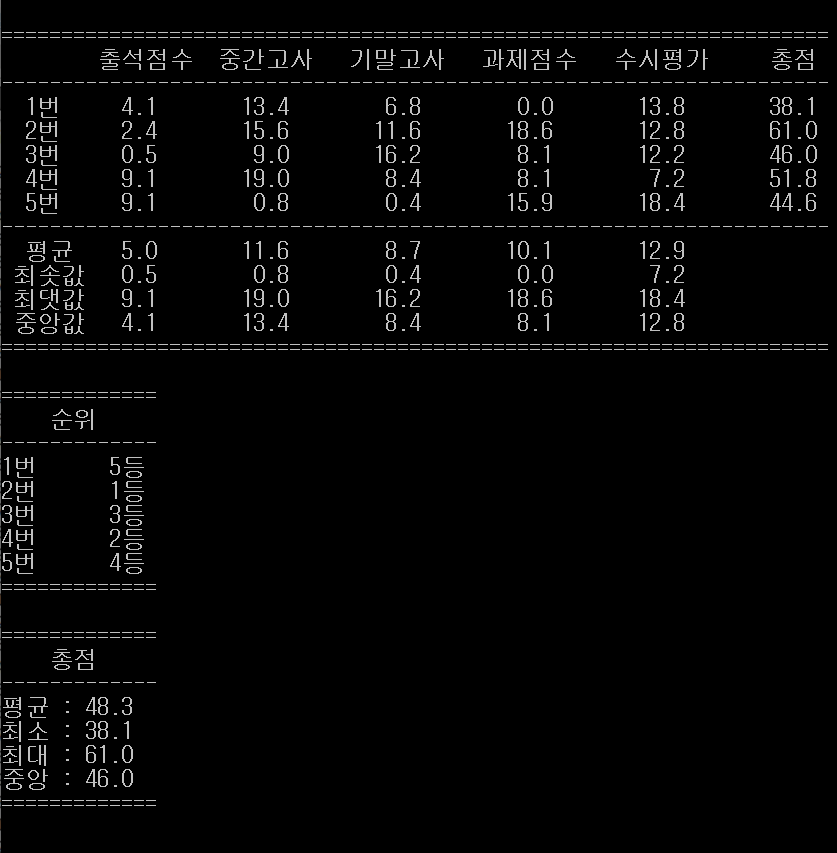
printf("중앙 : %4.1lf\n", arr\_sum[2]); // 인덱스 2 배열값 출력하여 중앙값 나타내기

// 모두 실수이므로 lf 변환명세 사용, 4칸으로 출력되며 소수점 첫째자리까지 나타남

printf("=============\n"); // 총점 값 출력 완료

return 0; // 0을 반환하여 끝내기

}



* srand() 함수를 사용하지 않았으므로 실행할 때마다 같은 값이 나오는데, 이유는 알 수 없으나 출력 형태를 바꾸었더니 다른 값을 받는 것으로 나타남