2019202050 이강현

#include <stdio.h>

int main()

{

int arr2d[3][3];//2차원배열 선언

printf("%d \n", arr2d);// 1행 1열값 출력

printf("%d \n", arr2d[0]);// 1행 1열값 출력

printf("%d \n\n", &arr2d[0][0]);// 1행 1열값 출력

printf("%d \n", arr2d[1]);// 2행 1열값 출력

printf("%d \n\n", arr2d[1][0]);// 2행 1열값 출력

printf("%d \n", arr2d[2]);// 3행 1열값 출력

printf("%d \n\n", arr2d[2][0]);// 3행 1열값 출력

printf("sizeof(arr2d): %d \n", sizeof(arr2d));// 배열 전체크기 출력

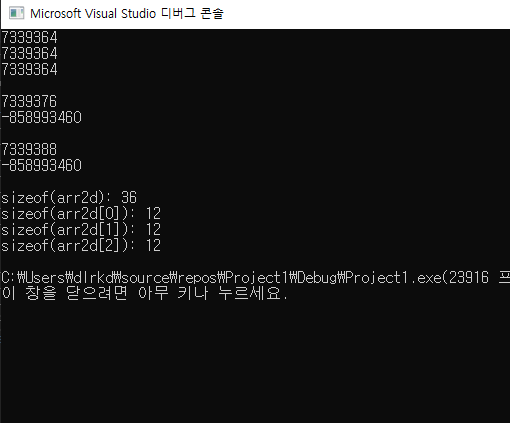
printf("sizeof(arr2d[0]): %d \n", sizeof(arr2d[0]));//배열 첫행 크기 출력

printf("sizeof(arr2d[1]): %d \n", sizeof(arr2d[1]));//배열 두번째 행 크기 출력

printf("sizeof(arr2d[2]): %d \n", sizeof(arr2d[2]));//배열 세번째 행 크기 출력

return 0;//종료

}



한줄 토의:arr2d와 arr2d[0]의 차이를 크기출력을 통해 알았다.

#include <stdio.h>

int main()

{

int arr1[3][2];//행3 열2인 배열 선언

int arr2[2][3];//행2 열3인 배열 선언

printf("arr1: %p \n", arr1);//arr1의 행1열1 주소 출력

printf("arr1+1: %p \n", arr1 + 1);//arr1의 행2열1 주소 출력

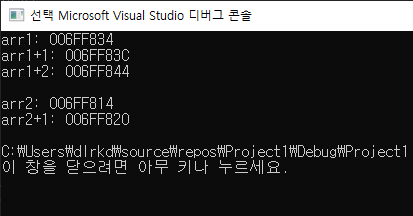
printf("arr1+2: %p \n\n", arr1 + 2);//arr1의 행3열1 주소 출력

printf("arr2: %p \n", arr2);//arr2의 행1열1 주소 출력

printf("arr2+1: %p \n", arr2 + 1);//arr2의 행2열1 주소 출력

return 0;//종료

}



한줄 토의: 배열마다의 포인터 값 증가의 차이는 각 행의 길이에 따라 차이가 발생하기 때문에 포인터 형과 행의 길이가 중요함을 알수 있다.

#include <stdio.h>

int main()

{

int arr1[2][2] = {

{1,2},{3,4}

};

/\*행2열2배열 선언및 초기화\*/

int arr2[3][2] = {

{1,2},{3,4},{5,6}

};

/\*행2열2배열 선언및 초기화\*/

int arr3[4][2] = {

{1,2},{3,4},{5,6},{7,8}

};

/\*행2열2배열 선언및 초기화\*/

int(\*ptr)[2];//가로 2에 int형 배열 포인터

int i;

ptr = arr1;//ptr이 arr1을 가리킴

printf("\*\*Show 2,2 arr1\*\*\n");

for (i = 0; i < 2; i++)

printf("d %d \n", ptr[i][0], ptr[i][1]);//arr1 출력 반복문

ptr = arr2; //ptr이 arr1을 가리킴

printf("\*\*Show 3,2 arr2\*\*\n");

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

printf("d %d \n", ptr[i][0], ptr[i][1]);//arr2 출력 반복문

ptr = arr3;//ptr이 arr1을 가리킴

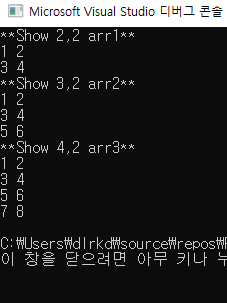
printf("\*\*Show 4,2 arr3\*\*\n");

for (i = 0; i < 4; i++)

printf("d %d \n", ptr[i][0], ptr[i][1]);//arr3 출력 반복문

return 0;

}



한줄 토의:가로 길이가 같은 배열끼리는 포인터 기본형을 설정하여 대입하면서 활용할 수 있다.

#include <stdio.h>

int main()

{

int num1 = 10, num2 = 20, num3 = 30, num4 = 40;//int형 변수 선언및 초기화

int arr2d[2][4] = { 1,2,3,4,5,6,7,8 };//int형 배열 선언 및 초기화

int i, j;

int\*whoA[4] = { &num1,&num2,&num3,&num4 };//포인터 배열

int(\*whoB)[4] = arr2d;//배열 포인터

printf("%d %d %d %d\n", \*whoA[0], \*whoA[1], \*whoA[2], \*whoA[3]);//포인터 배열 값 출력

for (i = 0; i < 2; i++)

{

for (j = 0; j < 4; j++)

printf("%d", whoB[i][j]);

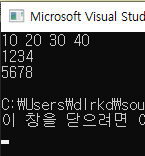
printf("\n");

}

/\*배열 포인터 출력문\*/

return 0;

}



한줄 토의: 배열 포인터와 포인터 배열의 차이를 알 수 있다.

#include <stdio.h>

void ShowArr2Style(int(\*arr)[4], int column)//배열의 이름과 세로길이를 인자로 받는다

{

/\*배열 출력 반복문\*/

int i, j;

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

{

for (j = 0; j < 4; j++)

printf("%d", arr[i][j]);

printf("\n");

}

printf("\n");

}

int Sum2DArr(int arr[][4], int column)//배열의 이름과 세로길이를 인자로 받는다

{

/\*배열 값의 총 합을 구하는 반복문\*/

int i, j, sum = 0;

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

for (j = 0; j < 4; j++)

sum += arr[i][j];

return sum;// 총합 반환

}

int main()

{

int arr1[2][4] = { 1,2,3,4,5,6,7,8 };// 행2 열 4인 배열 선언 및 초기화

int arr2[3][4] = { 1,1,1,1,3,3,3,3,5,5,5,5 };// 행3 열 4인 배열 선언 및 초기화

ShowArr2Style(arr1, sizeof(arr1) / sizeof(arr1[0]));//함수에 값 대입

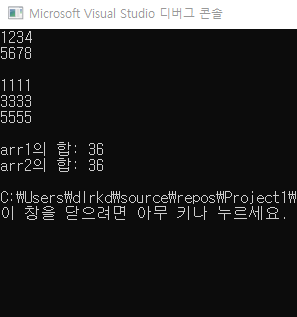
ShowArr2Style(arr2, sizeof(arr2) / sizeof(arr2[0]));//함수에 값 대입

printf("arr1의 합: %d \n", Sum2DArr(arr1, sizeof(arr1) / sizeof(arr1[0])));//함수에 값 대입

printf("arr2의 합: %d \n", Sum2DArr(arr2, sizeof(arr2) / sizeof(arr2[0])));//함수에 값 대입

return 0;

}



한줄 토의: int (\*arr)[4]와 int \*arr[][4]는 가로길이 4인 배열을 나타내는 동일한 뜻

#include <stdio.h>

int main()

{

int a[3][2] = { {1,2},{3,4},{5,6} };

printf("a[0]: %p \n", a[0]);//행1 열1 주소 출력

printf("\*(a+0): %p \n", \*(a + 0));//행1 열1 주소 출력

printf("a[1]: %p \n", a[1]);//행2 열1 주소 출력

printf("\*(a+1): %p \n", \*(a + 1));//행2 열1 주소 출력

printf("a[2]: %p \n", a[2]);//행3 열1 주소 출력

printf("\*(a+2): %p \n", \*(a + 2));//행3 열1 주소 출력

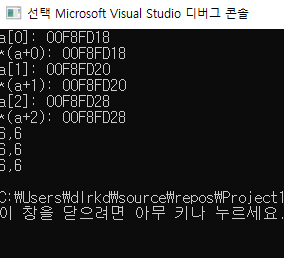
printf("%d,%d \n", a[2][1], (\*(a + 2))[1]);//행3 열2 값 출력

printf("%d,%d \n", a[2][1], \*(a[2] + 1));//행3 열2 값 출력

printf("%d,%d \n", a[2][1], \*(\*(a + 2) + 1));//행3 열2 값 출력

return 0;

}



한줄 토의:Arr[i]와 \*(Arr+i)는 같은 말이라는 것 을 알수 있다.

#include <stdio.h>

int main()

{

int arr[2][2][2] = { 1,2,3,4,5,6,7,8 };

printf("%d\n", arr[1][0][1]);

printf("%d\n", \*(arr[1][0] + 1));

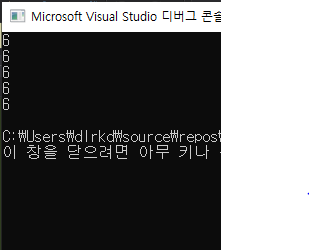
printf("%d\n", (\*(arr + 1))[0][1]);

printf("%d\n", \*((\*(arr + 1))[0] + 1));

printf("%d\n", \*(\*(\*(arr+1) + 0)+1));

return 0;

}



한줄 토의: Arr[i]와 \*(Arr+i)을 이용하여 배열을 다양하게 표현 가능함을 알 수 있다.