자료구조응용

04. 정적할당 배열, 동적할당 배열, 구조체 (14점)

2022.3.16.

1. 다음은 정적으로 할당받은 2차원 배열과 동적으로 할당받은 2차원배열을 함수로 전달하는 방법에 대해 비교하는 프로그램이다. 구현 조건에 맞게 각 함수를 정의한 후 실행하고 테스트해 보시오. (4점)

```
int main(void)
{
    // 정적할당의 2차원 배열(2행3열)
    int ary2D[2][3]= { {1, 2, 3}, {4, 5, 6}};

    // 동적할당의 2차원 배열(2행3열)
    int r, c;
    int **ary = (int **) malloc(sizeof(int*) * 2);
    for ( r = 0; r < 2; r++)
        ary[r] = (int *) malloc( sizeof(int) * 3);

    for ( r = 0; r < 2; r++)
        for ( c = 0; c < 3; c++)
            ary[r][c] = r+c;

    // 정적할당 배열
    printf("sumAry2D_f1() %d\n",  sumAry2D_f1(ary2D, 2, 3)); // 배열파라미터(권장)
    printf("sumAry2D_f2() %d\n", sumAry2D_f2(ary2D, 2, 3));  // 배열포인터
    printf("sumAry2D_f3() %d\n", sumAry2D_f3(ary2D, 2, 3));

    // 동적할당 배열
    printf("sumAry2D_f4() %d\n", sumAry2D_f4(ary, 2, 3));
    printf("sumAry2D_f5() %d\n", sumAry2D_f5(&ary, 2, 3));

    // 동적할당 배열을 sumAry2D_f1, f2, f3로 전달할 수 있을까? 테스트해보라!
    //printf("sumAry2D_f1~f3() %d\n", sumAry2D_f1(ary, 2, 3));

    // 정적할당 배열을 sumAry2D_f4, f5로 전달할 수 있을까? 테스트해보라!
    //printf("sumAry2D_f4~f5() %d\n", sumAry2D_f4(ary2D, 2, 3));

    freeAry2D(ary, 2); // 동적할당 배열의 메모리 해제

    return 0;
}
```

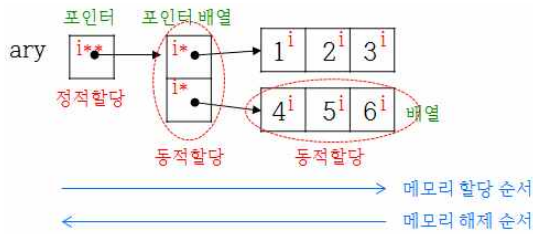
프로그램 1. 2차원 배열의 함수 전달 (정적할당 배열 vs. 동적할당 배열)

▶▷구현세부사항

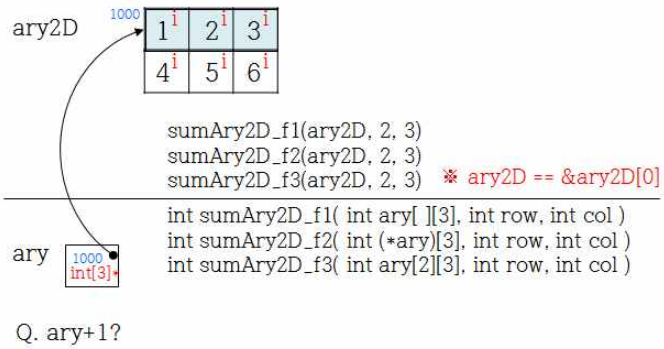
❶ 함수 sumAry2D_f1 ~ f3은 정적할당 배열을 전달하는 3가지 표현을 각각 사용
❷ 함수 sumAry2D_f4 ~ f5는 동적할당 배열을 전달하는 2가지 표현을 각각 사용

▶▷2차원 배열을 함수로 전달하기

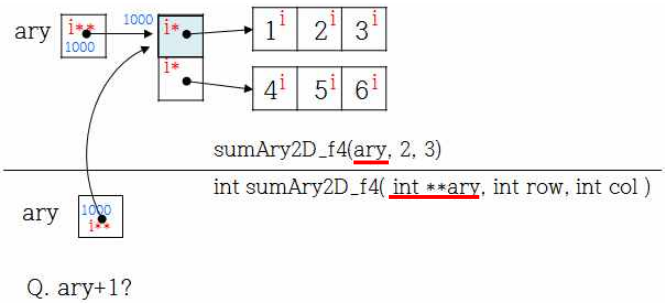
❶ 정적으로 할당되는 메모리의 크기는 컴파일 시점에, 동적으로 할당되는 메모리의 크기는 실행 중에 결정된다. **정적할당 변수**의 메모리 할당 시기는 프로그램 시작이나 함수호출 시이며, 메모리 해제 시기는 프로그램 종료 혹은 함수 반환 시점이다. **동적할당 변수**의 메모리 할당은 **malloc, calloc 혹은 realloc** 함수 등을 호출할 때 일어나며, 메모리 해제는 **free** 함수 호출을 통해 이루어진다. 이 문제에서 구현한 **동적할당 2차원 배열의 메모리 할당과 해제 순서**는 다음 그림과 같다.

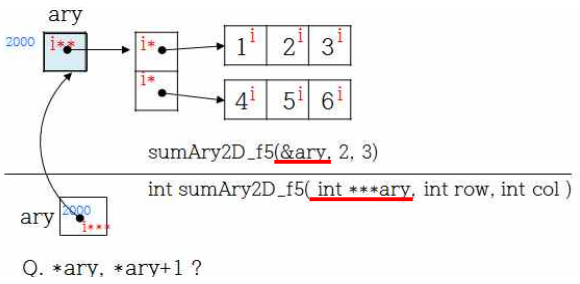


❷ 정적할당 배열을 함수로 전달하기

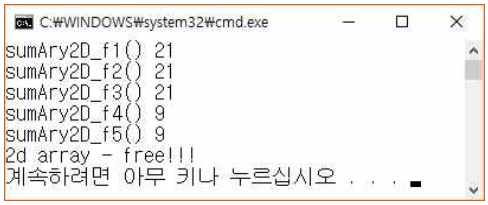


❸ 동적할당 배열을 함수로 전달하기





▶▷실행예



2. 프로그램 2는 정적으로 할당된 2차원 배열 3개를 전달하여 더하기를 수행하는 함수이다. 다음 조건으로 2차원 배열을 사용한 행렬 더하기 프로그램을 작성하라.(3점)

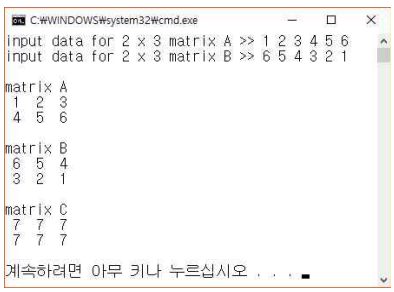
▶▷함수원형 및 프로그램

```
void input2dArray(int ary[][COLS], int rows, int cols); // 사용자 데이터 입력
void print2dArray(int ary[][COLS], int rows, int cols); // 행렬 출력
void add(int a[][COLS], int b[][COLS], int c[COLS], int rows, int cols);
```

```
void add(int a[][COLS], int b[][COLS], int c[][COLS], 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];
}
```

프로그램 2. 정적할당 배열을 사용한 행렬 덧셈

▶▷실행예



▶▷셀프체크

add 함수 호출 후 반환 직전/직후의 메모리 상태(스택)를 그림으로 그려보고 디버거의 지역창 및 호출스택 등을 통해 확인해 보시오.

3. **프로그램 3은 동적으로 2차원 배열을 생성하는 함수이며, 프로그램 4는 동적 배열을 사용하여 행렬 더하기를 수행하는 함수이다. 다음 조건으로 동적할당의 2차원 배열을 사용한 행렬 더하기 프로그램을 작성하시오. (3점)**

▶▷함수원형 및 프로그램

```
void inputMatrix(int **ary, int rows, int cols); // 사용자 데이터 입력
void printMatrix(int **ary, int rows, int cols); // 행렬 출력
void addMatrix(int **a, int **b, int **c, int rows, int cols);
int** make2dArray(int rows, int cols);
void free2dArray(int **ary, int rows);
```

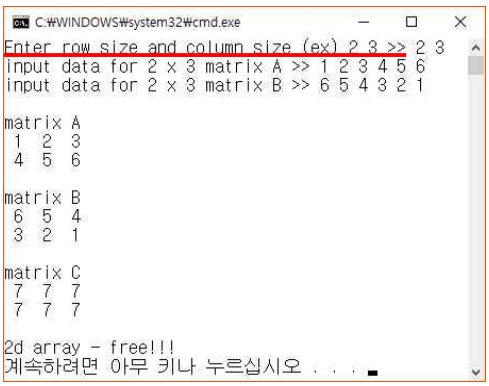
```
int** make2dArray(int rows, int cols)
{
    int **ary, i;

    MALLOC(ary, rows * sizeof( *ary));      // 매크로함수 MALLOC을 사용
    for( i = 0; i < rows; i++)
        MALLOC(ary[i], cols * sizeof( **ary));

    return ary;
}
```

프로그램 3. 동적할당으로 2차원 배열 생성하기

```
void addMatrix(int **a, int **b, int **c, 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];
}
```

프로그램 4. 동적할당 배열을 사용한 행렬 덧셈

▶▷실행예



▶▷셀프체크

❶ addMatrix 함수 호출 후 반환 직전/직후의 메모리 상태(스택)를 그림으로 그려보고 디버거의 지역창, 조사식 및 호출스택 등을 통해 확인해 보시오.

❷ inputMatrix 함수의 파라미터 int **a를 int a[][열의크기]로 수정 후 제대로 실행이 되는지 확인해 보시오.

4. 다음은 두 구조체에 대한 동등성을 체크하는 프로그램의 일부이다. 주어진 구현조건에 맞는 실행결과를 보이도록 프로그램을 작성하시오. (4점)

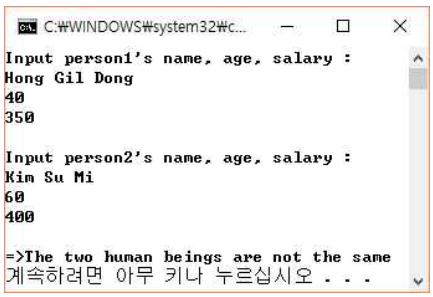
```
#define FALSE 0
#define TRUE 1

typedef struct{
    char name[20];
    int age;
    float salary;
} humanBeing;

int humansEqual( humanBeing *person1, humanBeing *person2)
{
    if(_____)
        return FALSE;

    if( person1->age != person2->age )
        return FALSE;

    if( person1->salary != person2->salary )
        return FALSE;

    return TRUE;
}
```

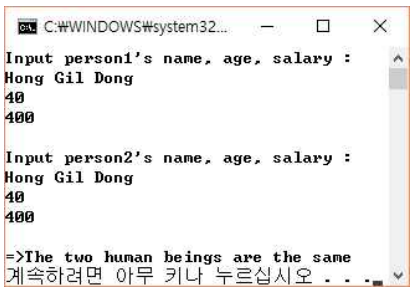
프로그램 5. 두 구조체 변수의 동등성 체크

▶▷구현세부사항

공백을 포함한 이름을 입력하려면, scanf 대신 gets 함수를 사용해야 한다.

▶▷실행예





■ 제출 형식

- 솔루션 이름 : DS 04
- 프로젝트 이름 : 1, 2, 3, 4
- 각 소스파일에 주석처리

  "학번 이름"

  "본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다."
- 실행화면을 캡쳐한 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 열람기한) : 수업 익일 자정(만점의 50%, 반올림)
- 1, 2차 마감 이외의 제출은 허용하지 않습니다. (이메일 제출 불가!)