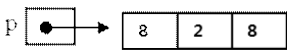
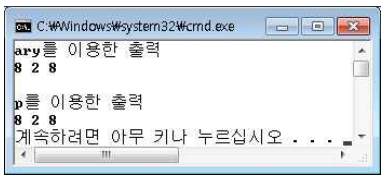
자료구조응용
01. 배열과 포인터 (11점)

2022.3.2.

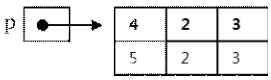
1. 다음 각 경우에 대해 배열을 선언하고 배열이름 ary 및 배열 포인터 p를 이용하여 배열요소를 출력하는 프로그램을 작성하라. 이때, ary와 p를 사용한 여러 가지 주소표현 및 역참조를 테스트하라.

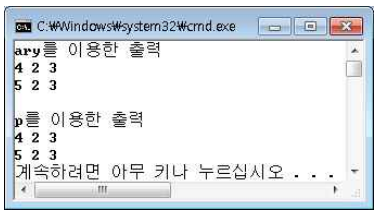
(1) 1차원 배열과 배열포인터



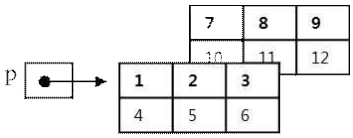


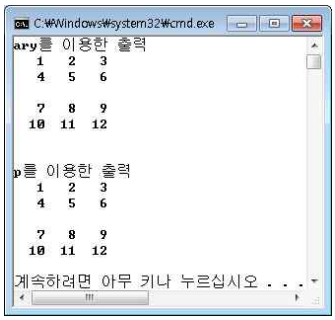
(2) 2차원 배열과 배열포인터



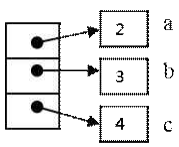


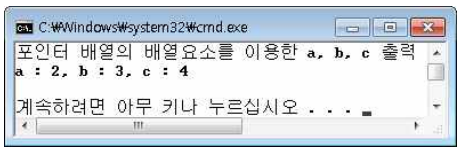
(3) 3차원 배열과 배열포인터





2. 다음 그림과 같은 구조를 가지도록 코드를 작성하고, 배열원소의 역참조를 이용해서 a, b, c의 값을 출력하는 문장을 추가하라. 단, 변수 a, b, c는 int형이며 **포인터 배열**을 사용하라.





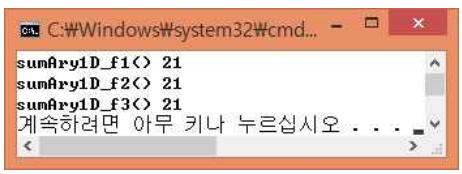
3. 다음은 1차원 배열에 대해 배열원소의 합을 구하는 프로그램의 일부이다. 형식매개변수가 다른 세 가지 버전의 함수를 각각 정의하고 실행되도록 작성하라. 각 함수는 배열 파라미터 혹은 배열 포인터를 매개변수에 사용하여야 하고 인자의 개수는 2개이다.

```
int main(void)
{
      int ary1D[ ] = {1, 2, 3, 4, 5, 6};

      printf("sumAry1D_f1() %d\n",   sumAry1D_f1(_____________));
      printf("sumAry1D_f2() %d\n\n", sumAry1D_f2(_____________));
      printf("sumAry1D_f3() %d\n\n", sumAry1D_f3(_____________));

      return 0;
}
```

▶▷함수원형

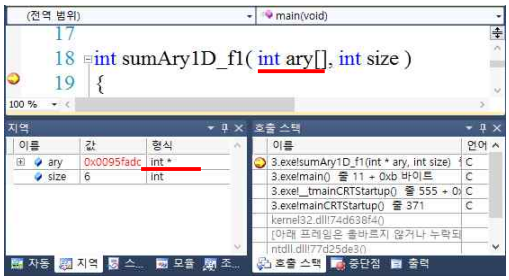
```
int sumAry1D_f1( int ary[], int size ); // 배열파라미터, 배열크기  (권장)
int sumAry1D_f2( int *ary,  int size ); // 배열포인터, 배열크기
int sumAry1D_f3( int ary[6],  int size ); //
```

▶▷실행예



▶▷셀프체크

디버거를 통해 3가지 배열전달 방법(int ary[], int *ary, 혹은 int[6])이 같은 포인터 타입(int *)의 함수인자를 사용하는 것을 확인하라.



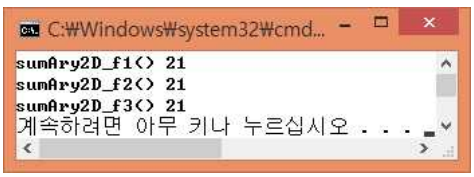
4. 다음은 2차원 배열에 대해 배열원소의 합을 구하는 프로그램의 일부이다. 형식매개변수가 다른 세 가지 버전의 함수를 각각 정의하고 실행되도록 작성하라. 각 함수는 배열 파라미터 혹은 배열 포인터를 매개변수에 사용하여야 하고 인자의 개수는 3개이다.

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

      printf("sumAry2D_f1() %d\n",   sumAry2D_f1(_____________));
      printf("sumAry2D_f2() %d\n\n", sumAry2D_f2(_____________));
      printf("sumAry2D_f3() %d\n\n", sumAry2D_f3(_____________));

    return 0;
}
```

▶▷함수원형

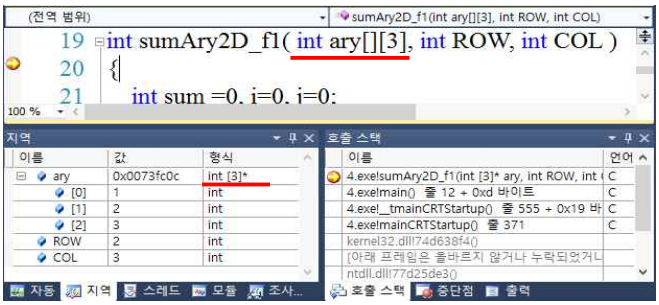
int sumAry2D_f1(int ary[][3], int ROW, int COL); // 배열파라미터 (권장)
int sumAry2D_f2(int (*ary)[3], int ROW, int COL); // 배열포인터
int sumAry2D_f3(int ary[2][3], int ROW, int COL);

▶▷실행예



▶▷셀프체크

디버거를 통해 3가지 배열전달 방법(int ary[][3], int (*ary)[3], 혹은 int ary[2][3])이 같은 포인터 타입(int [3]*)의 함수인자를 사용하는 것을 확인하시오.



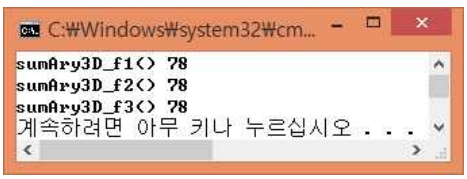
5. 다음은 3차원 배열에 대해 배열원소의 합을 구하는 프로그램의 일부이다. 형식매개변수가 다른 세 가지 버전의 함수를 각각 정의하고 실행되도록 작성하라. 각 함수는 배열 파라미터 혹은 배열 포인터를 매개변수에 사용하여야 하고 인자의 개수는 4개이다.

```
int main(void)
{
      int ary3D[ ][2][3] = { {{1, 2, 3}, {4, 5, 6}},
                             {{7, 8, 9}, {10, 11, 12} };

      printf("sumAry3D_f1() %d\n",   sumAry3D_f1(_____________));
      printf("sumAry3D_f2() %d\n\n", sumAry3D_f2(_____________));
      printf("sumAry3D_f3() %d\n\n", sumAry3D_f3(_____________));

    return 0;
}
```

▶▷함수원형

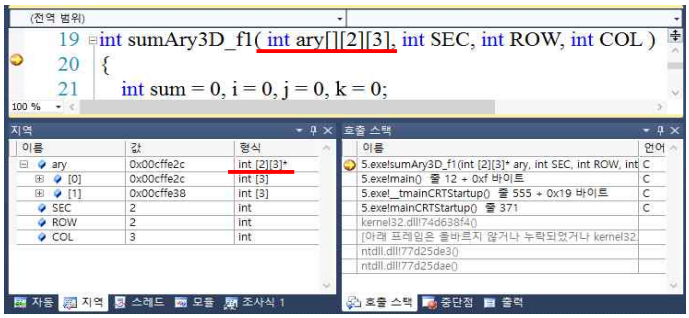
int sumAry3D_f1(int ary[][2][3], int SEC, int ROW, int COL);// 배열파라미터(권장)
int sumAry3D_f2(int (*ary)[2][3], int SEC, int ROW, int COL); // 배열포인터
int sumAry3D_f3(int ary[2][2][3], int SEC, int ROW, int COL);

▶▷실행예



▶▷셀프체크

디버거를 통해 3가지 배열전달 방법(int ary[][2][3], int (*ary)[2][3], 혹은 int ary[2][2][3])이 같은 포인터 타입(int [2][3]*)의 함수인자를 사용하는 것을 확인하시오.



6. 메모리의 동적 할당과 해제에 대해 다음 두 가지 버전으로 프로그램을 작성하라. 모두 제대로 main 함수를 구현하여 실행되도록 하여야 한다. 또한, 변수 i, f에 적당한 값을 저장하여 출력하는 문장도 추가하라.

```
int i, *pi;
float f, *pf;
pi = (int *) malloc(sizeof(int));
pf = (float *) malloc(sizeof(float));
*pi = 1024;
*pf = 3.14;
printf("an integer = %d, a float = %f\n", *pi, *pf);
free(pi);
free(pf);
```

Program 1.1: Allocation and deallocation of memory

(1) 다음 코드를 이용하여 Program1.1을 수정

```
if ((pi = (int *) malloc(sizeof(int))) == NULL ||
    (pf = (float *) malloc(sizeof(float))) == NULL)
{fprintf(stderr, "Insufficient memory");
 exit(EXIT_FAILURE);
}
```

or by the equivalent code

```
if (!(pi = malloc(sizeof(int))) ||
    !(pf = malloc(sizeof(float))))
{fprintf(stderr, "Insufficient memory");
 exit(EXIT_FAILURE);
}
```

(2) 다음 매크로를 이용하여 Program1.1을 수정

```
#define MALLOC(p,s) \
   if (!((p) = malloc(s))) {\
      fprintf(stderr, "Insufficient memory"); \
      exit(EXIT_FAILURE);\
   }
```

※ MALLOC 사용 시 에러표시(붉은 밑줄)는 무시할 것.

7. 다음과 같은 프로그램을 두 가지 버전으로 작성하시오.

▶▷실행순서

① 사용자로부터 난수생성 개수(n)를 입력받는다.
② 정수 난수를 n개 발생시켜 1차원 배열에 저장한다.
③ 1차원 배열에 대해 선택정렬(selection sort)을 수행한다.
④ 사용자로부터 임의의 정수를 입력받는다.
⑤ 입력받은 정수가 배열에 있는지 이진탐색(binary search)을 수행하여 그 결과를 출력한다.

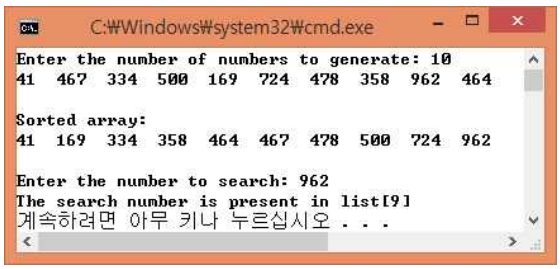
▶▷난수생성조건

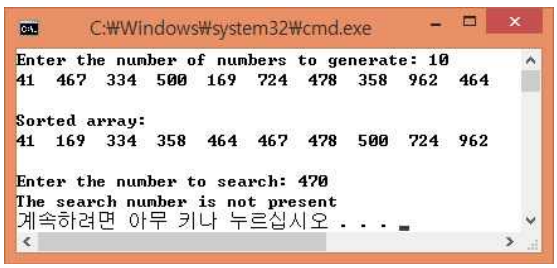
난수생성 개수는 최대 100개, 난수 범위는 0~999, 난수 중복허용, SEED를 지정하지 않음

▶▷교재 참고프로그램

Program 1.3, 1.4, 1.6, 1.7, 1.8

▶▷실행예





(1) 함수 swap, compare와 반복문을 사용한 이진탐색을 구현한 버전

(2) 매크로 SWAP, COMPARE와 재귀호출을 사용한 이진탐색을 구현한 버전

■ 제출 형식

- 솔루션 이름 : DS 01
- 프로젝트 이름 : 1-1, 1-2, 1-3, 2, 3, 4, 5, 6-1, 6-2, 7-1, 7-2
- 각 소스파일에 주석처리
  "학번 이름"
  "본인은 이 소스파일을 다른 사람의 소스를 복사하지 않고 직접 작성하였습니다."
- 실행화면을 캡쳐하여 한글파일에 추가 후 솔루션 폴더에 포함
- 솔루션 정리 메뉴를 수행 후 전체 솔루션을 "학번.zip"으로 압축하여 제출

■ 주의

- 소스 복사로는 실력향상을 기대할 수 없습니다!!!

- 1차 마감 : 수업일 자정 (프로젝트 당 1점씩, 11점 만점)

- 2차 마감 : 수업 익일 자정(만점의 50%, 반올림)