

# Report 2

제어문과 배열

학번	2018212236
학부	전자정보통신공학
이름	김동주

제출일자	2018-09-16
담당교수	반상우

작업환경	Visual Studio Code
컴파일러	MinGW GCC

# 필수 문제

제어문과 배열 내용 학습에 가장 도움이 되는 프로그램 구현 5개 이상.

## 1. 배열의 선언 방법

내용	배열을 선언하는 다양한 방법에 대하여 알아보았다.
코드	<pre>#include &lt;stdio.h&gt;  int main() {     int arr1[4];     int arr2[] = { 1, 2, 3, 4 };     int arr3[4] = { 1, 2, 3, 4 };     int arr4[4] = { 0 };      int i;      for (i = 0; i &lt; 4; i ++){         printf("%d ", arr1[i]);         puts("");     }      for (i = 0; i &lt; 4; i ++){         printf("%d ", arr2[i]);         puts("");     }      for (i = 0; i &lt; 4; i ++){         printf("%d ", arr3[i]);         puts("");     }      for (i = 0; i &lt; 4; i ++){         printf("%d ", arr4[i]);         puts("");     }      return 0; }</pre>

## 실행결과

```

1  #include <stdio.h>
2
3  int main() {
4      int arr1[4];
5      int arr2[] = { 1, 2, 3, 4 };
6      int arr3[4] = { 1, 2, 3, 4 };
7      int arr4[4] = { 0 };
8
9
10     int i;
11
12     for (i = 0; i < 4; i++)
13         printf("%d ", arr1[i]);
14     puts("");
15
16     for (i = 0; i < 4; i++)
17         printf("%d ", arr2[i]);
18     puts("");
19     for (i = 0; i < 4; i++)
20         printf("%d ", arr3[i]);
21     puts("");
22
23     for (i = 0; i < 4; i++)
24         printf("%d ", arr4[i]);
25     puts("");
26
27     return 0;
28 }

```

```

e:\Git\clone_repository\Dongguk_Gyeongju\2018-2학기\프로그래밍실습2\Report2\원수>.f1.exe
4201163 4201072 0 3620864

```

## 고찰

4201163 4201072 0 3620864

1 2 3 4

1 2 3 4

0 0 0 0

의 출력중에서 첫 줄에 출력된 큰 수 들은 코드상에서 등장하지 않는다. 계산되어 나온 수 역시 아니다.

```
int arr1[4];
```

위의 arr1에서 출력된 값이 '4201163 4201072 0 3620864' 이고, 이는 선언되지 않은 변수로 부터 나왔다는 점, 배열 arr1의 네 원소의 값이 제 각각인 점을 보아, 선언되지 않은 변수의 값을 출력하면 알 수 없는 값이 나오는 것 같다.

왜 이런 현상이 일어나는 지를 찾아보았다.

우선 변수를 선언하게 되면, 그 변수에게 선언된 자료형의 크기에 해당하는 메모리를 할당하게 되고, 초기화를 통해 그 영역에 값을 지정해 주는 것이다. 초기화를 하지 않은 상태에선 그 변수가 가지는 메모리 영역이 이전에 사용하던 값이 남아있어, 그 잔재가 출력되는 것이다.

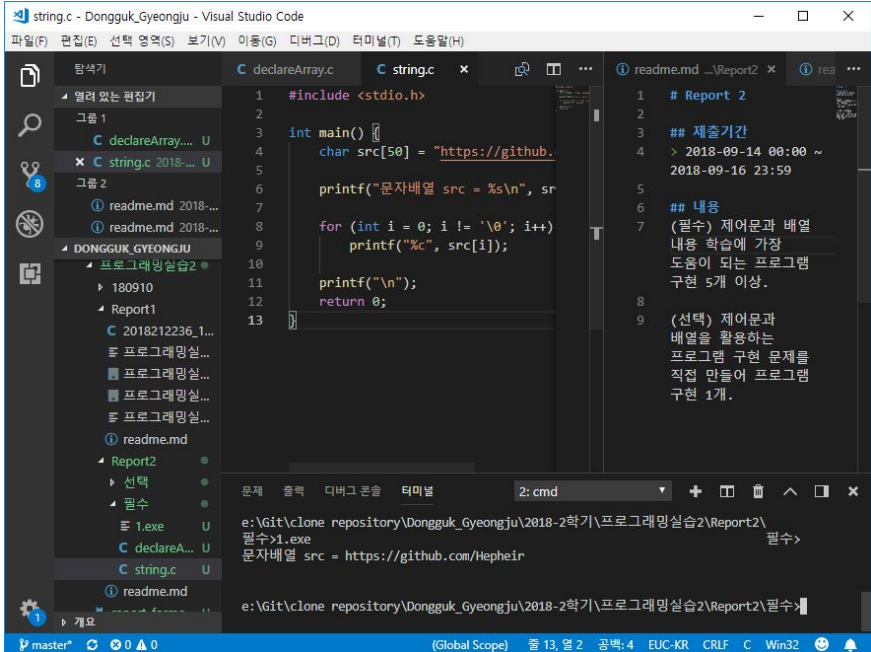
따라서 위 문제의 arr1이 가지는 메모리 영역이 이전에 사용하던 값을 계산해보면

(int 자료형은 4byte = 32bit 이므로)

4201163	00000000 01000000 00011010 11001011
4201072	00000000 01000000 00011010 01110000
0	00000000 00000000 00000000 00000000

	3620864	00000000 00110111 01000000 00000000
위와 같은 값이라고 예상된다.		

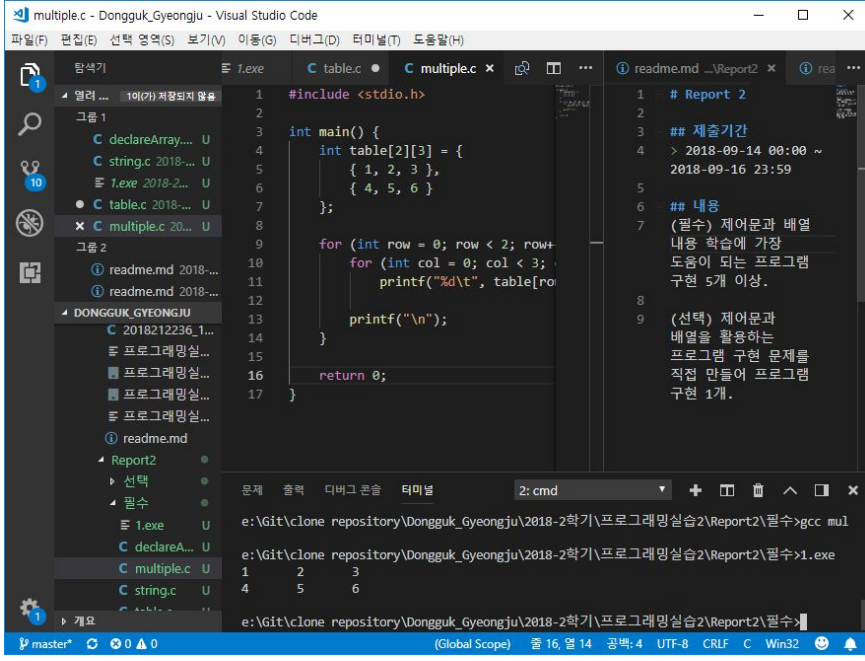
## 2. 문자의 배열

내용	문자 변수의 배열을 이용하여 문자열을 표현과 문자열의 출력
코드	<pre>#include &lt;stdio.h&gt;  int main() {     char src[50] = "https://github.com/Hepheir";      printf("문자배열 src = %s\n", src);      for (int i = 0; i != '\0'; i++)         printf("%c", src[i]);      printf("\n");     return 0; }</pre>
실행결과	
고찰	타 언어에는 문자열이라는 자료형이 존재하여 문장의 출력을 쉽게

	<p>할 수 있다. 같은 작업을 C언어에서는 어떻게 할 수 있을 지에 대하여 알아보았다.</p> <p>printf의 출력 포맷 중, 문자열을 출력하기 위한 형식문자 %s을 사용하거나, 일반 배열의 출력과 동일하게 문자열이 끝나는 부분인 NULL문자가 나오기 전까지 반복문을 통해 출력할 수 있었다.</p>
--	--

### 3. 2차원 배열

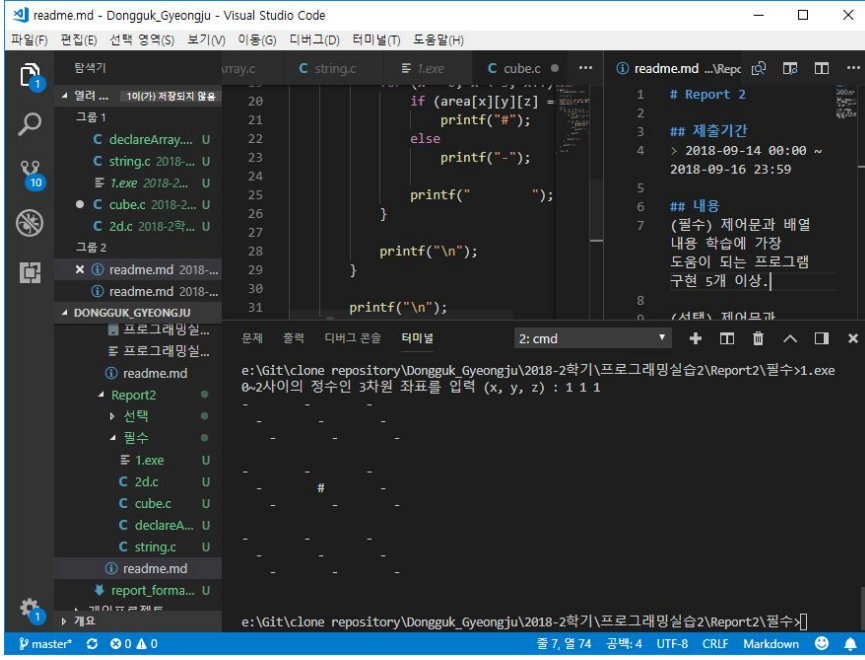
내용	2차원 배열을 선언하고 출력
코드	<pre>#include &lt;stdio.h&gt;  int main() {     int table[2][3] = {         { 1, 2, 3 },         { 4, 5, 6 }     };      for (int row = 0; row &lt; 2; row++) {         for (int col = 0; col &lt; 3; col++)             printf("%d\t", table[row][col]);          printf("\n");     }      return 0; }</pre>

<p><b>실행결과</b></p>	
<p><b>고찰</b></p>	<p>배열의 한 원소를 또 다른 배열을 사용하는 2차원 배열을 선언하였다. 자료의 저장에 매우 용이할 것이라고 생각하였다.</p> <p>잘 생각해 보면 컴퓨터에 저장되는 정보는 모두 1차원 디지털의 형태로 저장된다. 그렇기에 2차원의 평면, 3차원의 입체의 형태의 정보를 어떻게 저장하고 표현하는가에 대하여 의문이 생기곤 하였다. C언어 프로그래밍을 배우면서 배열안에 배열이라는 패러다임을 처음 접했을 때, 충격적이면서 어떻게 이런 사고의 전환을 했을까에 대해 감명을 받았다.</p> <p>차원의 한계를 뛰어넘어 1차원의 매체에 2차원, 3차원, ... n차원의 정보를 기록할 수 있게 되었다는 사실은 다시 생각해 보아도 매우 충격적이고 신선하며 창의적이라고 느낀다. 세삼 인류의 대단함을 다시금 떠올리게 된다.</p>

## 4. 다차원 배열의 활용

<p><b>내용</b></p>	<p>3차원 배열을 출력</p>
<p><b>코드</b></p>	<pre>#include &lt;stdio.h&gt;  int main() {     int area[3][3][3] = { 0 };      int x, y, z;      printf("0~2사이의 정수인 3차원 좌표를 입력 (x, y, z)</pre>

```
: ");  
    scanf("%d %d %d", &x, &y, &z);  
    area[x][y][z] = 1;  
  
    int i;  
    for (z = 3; z > 0; z--) {  
        for (y = 3; y > 0; y--) {  
            // Indent  
            for (i = 0; i < y; i++)  
                printf(" ");  
  
            for (x = 0; x < 3; x++) {  
                if (area[x][y][z] == 1)  
                    printf("#");  
                else  
                    printf("-");  
  
                printf(" ");  
            }  
  
            printf("\n");  
        }  
  
        printf("\n");  
    }  
  
    return 0;  
}
```

<p><b>실행결과</b></p>	
<p><b>고찰</b></p>	<p>1차원 매체에서 다양한 차원을 표현한다는 개념의 이해가 어려운 사람이 있을 수 있다. 그런 사람들의 이해를 도울 수 있지 않을까 하여 3차원 배열을 3D 입체의 형태와 비슷해 보이도록 행과 열, 그리고 입체감을 주기위한 인덴트(indent)를 사용하여 정사영의 형태로 화면에 출력하였다.</p>

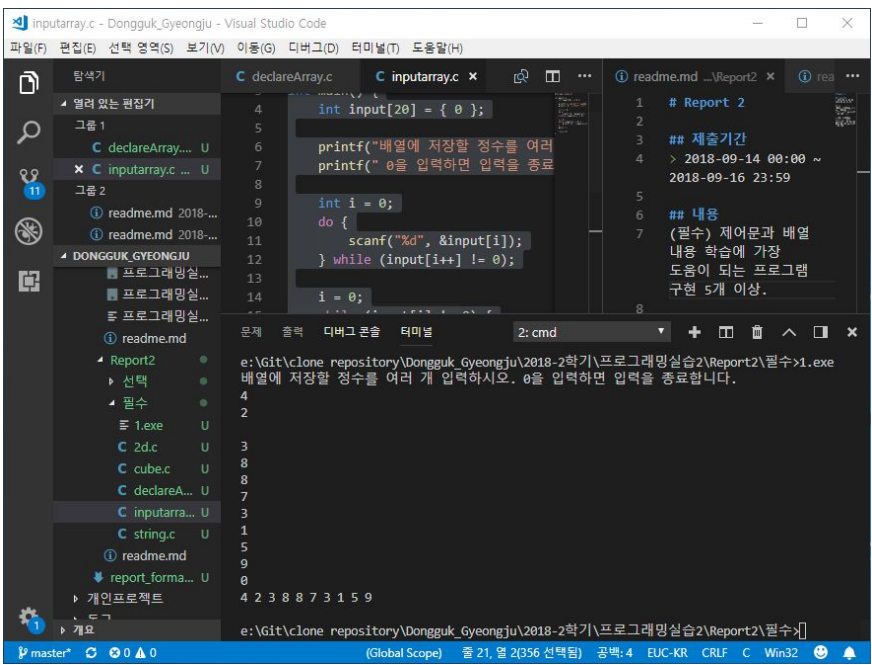
## 5. [교재 p.401] 배열에 정수 입출력

<p><b>내용</b></p>	<p>정수 int형 배열에 표준입력으로 받은 정수를 저장하여 출력</p>
<p><b>코드</b></p>	<pre>#include &lt;stdio.h&gt;  int main() {     int input[20] = { 0 };      printf("배열에 저장할 정수를 여러 개 입력하십시오.");     printf(" 0을 입력하면 입력을 종료합니다.\n");      int i = 0;     do {         scanf("%d", &amp;input[i]);     } while (input[i++] != 0);      i = 0;     while (input[i] != 0) {</pre>



```
printf("%d ", input[i++]);  
}  
puts("");  
  
return 0;  
}
```

**실행결과**



**고찰**

반복문과 표준입출력(stdio)를 이용하여 배열에 정수를 입력 받고, 0이 입력되었을 시, 입력을 멈추고 저장된 값들을 출력하였다.

배열의 특정 인덱스에 반복문을 이용하여 편리하게 값을 입력할 수 있다는 것과, 역으로 출력 하는 것까지 익히기에 도움이 되는 예제였다.

다만 예제를 풀면서... 위 예제는 배열보다는 반복문의 흐름을 이해하는 것이 조금 더 힘든 것 같다고 생각하였다. 가독성을 생각하면, for문과 같은 깔끔한 표현들이 있고

```
int i = 0;  
do {  
    scanf("%d", &input[i]);  
    i++;  
} while (input[i] != 0);
```

위와 같이 i++과 같은 증감문을 블록 맨 하단에 넣는 방법이 있음에도 불구하고

	<pre>while (input[i++] != 0); ... printf("%d ", input[i++] );</pre> <p>12, 15줄의 위와 같은 표현으로 코드를 작성하면, 코드의 줄은 조금 줄겠지만, 사람의 입장에선 좀 더 자세히 코드를 봐야하고 길고 복잡한 코드에선 저런 부분을 놓칠 수도 있겠다고 생각하였다.</p> <p>의외의 부분인 가독성에 관한 부분에서 깨달음을 얻은 풀이였다.</p>
--	---

## 선택 문제

제어문과 배열을 활용하는 프로그램 구현 문제를 직접 만들어 프로그램 구현 1개.

필수 문제의 1, 2, 3, 4번 문제 <== 직접 만든 프로그램