



C 언어 프로그래밍

Part 02. C 언어 기본 학습

Chapter 07. 배열

목차

1. 배열이란?
 2. 배열의 초기화
 3. 다차원 배열
 4. 배열의 크기와 길이
- [실전예제] 문자 배열 합치기

학습목표

- 배열의 개념과 인덱스의 개념을 배우고 적용할 수 있다.
- 배열을 초기화하는 형식을 배우고 문자열도 초기화할 수 있다.
- 다차원 배열의 개념을 이해하고 2차원 배열을 정의해본다.
- 배열의 크기와 길이의 용어를 정리하고 배열의 길이를 구할 수 있다.

01

배열이란?

01. 배열이란?

I. 배열의 개념

- 배열

배열은 같은 타입의 객체(변수)들이 일렬로 연속되게 모여서 구성된 객체이다.

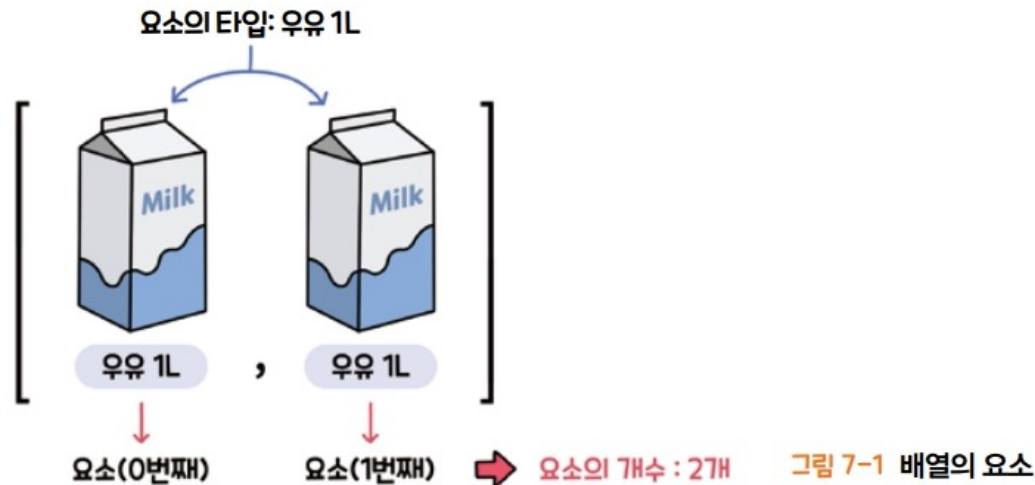
- '일렬로 연속되게'란 메모리 영역 관점에서 나온 개념
- 변수들이 각각 순서를 가지고 있으며, 각 변수가 차지한 메모리 영역이 빈틈없이 이어져 있음을 의미함

01. 배열이란?

I. 배열의 개념

• 배열에 관한 용어

- 요소 : 배열을 구성하는 낱개의 객체(변수)를 의미함
- 요소의 개수(배열의 길이) : 배열이 몇 개의 요소로 구성되는지를 나타냄
- 요소의 타입 : 각 요소의 타입을 의미함
- 요소의 순서 : 각 요소가 배열에서 몇 번째 요소인지를 나타냄



01. 배열이란?

II. 배열의 타입

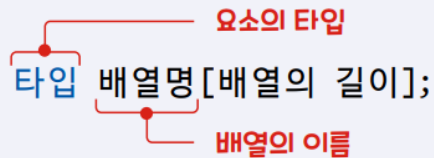
- 배열

- 배열은 자신의 종류를 나타내는 배열 타입이 있음
 - » 이는 집합 타입 중의 하나로 다루어짐
- 배열 타입은 요소의 개수와 요소의 타입으로 결정됨

01. 배열이란?

III. 배열의 정의

- '배열의 정의'의 개념
 - 배열을 메모리 영역에 마련하고 이름을 붙이는 것
- 배열을 정의하는 방법

요소
타입 배열명[배열의 길이];
배열의 이름

요소의 타입과 요소의 개수를 통해서
메모리 영역을 해석할 수 있습니다.

[코드 7-1] 배열의 정의

```
01 int main()  
02 {  
03     int arr[4];  
04 }
```


01. 배열이란?

IV. 인덱스

- 인덱스의 개념
 - 요소의 순번
 - 배열 요소가 N개일 때 마지막 요소의 인덱스는 N-1이 됨
 - 배열 요소는 각각 하나의 변수와 같음
 - 배열 이름, 첨자([]) 연산자, 인덱스를 결합하여 각각의 개별 요소에 접근할 수 있음

01. 배열이란?

IV. 인덱스

- 인덱스의 개념

[코드 7-2] 인덱스를 통한 요소의 접근

```
01  #include <stdio.h>
02
03  int main()
04  {
05      int arr[2];
06
07      arr[0] = 1;
08      arr[1] = 2;
09
10      printf("arr[1]: %d", arr[1]);
11  }
```

arr[1]: 2

01. 배열이란?

V. 배열의 복사

- 배열은 대입 연산자를 사용하기 어려움

[코드 7-3] 배열 사이의 복사

```
01  int main()
02  {
03      int a1 = 1;
04
05      int a2 = 2;      // 문제 없음
06
07      int a3;
08      a3 = a1;         // 문제 없음
09
10      int arr1[2];
11      arr1[0] = 1;
12      arr1[1] = 2;
13
14      int arr2[2] = arr1;  ← 오류 발생
15
16      int arr3[2];
17      arr3 = arr1;        ← 오류 발생
18  }
```

01. 배열이란?

V. 배열의 복사

- 일일이 개별 요소에 대해서 복사를 수행할 수 있음

[코드 7-4] 배열의 복사

```
01  #include <stdio.h>
02
03  int main()
04  {
05      int arr1[2];
06      arr1[0] = 1;
07      arr1[1] = 2;
08
09      int arr2[2];
10
11      for(int i = 0; i < 2; i++)
12          arr2[i] = arr1[i];
13
14      printf("arr2[0]: %d\r\n", arr2[0]);
15      printf("arr2[1]: %d", arr2[1]);
16  }
```

```
arr2[0]: 1
arr2[1]: 2
```

01. 배열이란?

확인문제1

1. int 변수가 7개 모여서 배열을 구성한다면 해당 배열이 차지하는 메모리 영역의 크기는 몇 바이트인가?

2. 다음 빈칸에 들어갈 단어를 채우시오.

배열의 타입을 결정하는 것은 요소의 와 이다. 또한 변수 하나와 같은 타입의 요소 하나인 배열은 서로 .

3. 요소의 타입이 float이고 요소의 개수가 64인 배열 arr을 정의하고 arr의 타입을 표기하시오.

(1) 정의 : _____ (2) 타입 : _____

4. 다음 코드에서 잘못된 점을 설명하시오.

```
int arr[4];  
arr[4] = 1;
```

5. 다음 코드에서 잘못된 부분은 몇 행이고 이유는 무엇인지 서술하시오.

```
01 int arr1[1];  
02 arr1[0] = 1;  
03 int arr2[1];  
04 arr2 = arr1;  
05 int a = arr2[0];
```

01. 배열이란?

LAB 7-1

알파벳 배열에 A-Z 채우기

요소의 타입이 `char`이고 요소의 개수가 26개인 배열 `alphabets`를 정의하고 각각의 요소를 'A'부터 'z'까지 채우는 코드를 작성해봅시다.

`alphabets`의 각 요소를 순회하면서 순서대로 'A'부터 'Z'까지 대입한다.

01. 배열이란?

LAB 7-1

정답

```
01  int main()
02  {
03      char alphabets[26];
04      for(int i = 0; i < 26; i++)
05      {
06          alphabets[i] = 'A' + i;
07      }
08  }
```

02

배열의 초기화

02. 배열의 초기화

I. 초기화 형식

- 배열의 초기화

- 배열을 정의할 때 각 요소들을 초기화하는 것

- 배열의 초기화 형식 1

타입 배열명[배열의 길이] = {요소0, 요소1, 요소2, ..., 요소K}; (단, $K < \text{배열의 길이}$)

요소의 개수

요소의 인덱스는 0부터 시작함

- 배열의 초기화 형식 2

타입 배열명[] = {요소0, 요소1, 요소2, ..., 요소K}; (요소 개수 = $K + 1$)

요소의 인덱스는 0부터 시작함

02. 배열의 초기화

I. 초기화 형식

[코드 7-5] 배열 초기화 형식 1

```
01  int main()
02  {
03      int arr1[3] = { 0 };
04      int arr2[3] = { 0, };
05
06      int arr3[3] = { 1 };
07      int arr4[3] = { 1, };
08
09      int arr5[3] = { 1, 2 };
10      int arr6[3] = { 1, 2, };
11
12      int arr7[3] = { 1, 2, 3 };
13      int arr8[3] = { 1, 2, 3, };
14
15      int arr9[3];
16  }
```

02. 배열의 초기화

I. 초기화 형식

[코드 7-6] 배열 초기화 형식 2

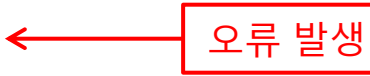
```
01  int main()
02  {
03      int arr1[] = {1};
04      int arr2[] = {1,};
05
06      int arr3[] = {1, 2};
07      int arr4[] = {1, 2,};
08
09      int arr5[];
10  }
```

02. 배열의 초기화

I. 초기화 형식

[코드 7-7] 중괄호 초기값

```
01  int main()  
02  {  
03      int arr[3];  
04      arr = { 1, 2, 3 };  
05  }
```



02. 배열의 초기화

II. 문자열 초기화

- 각각의 문자열은 문자 타입 요소로 구성된 배열로 취급함

[코드 7-8] 아스키코드 문자 배열 초기화

```
01  #include <stdio.h>
02
03  int main()
04  {
05      char s1[ ] = { 'K', 'o', 'r', 'e', 'a', '\0' };
06      char s2[ ] = { "Korea" };
07      char s3[ ] = "Korea";
08
09      printf("%s %s %s", s1, s2, s3);
10  }
```

Korea Korea Korea

02. 배열의 초기화

II. 문자열 초기화

- NULL 종료 문자열로 인해 컴퓨터가 문자열의 끝을 인지하고 출력함

[코드 7-9] NULL 종료 문자열

```
01 #include <stdio.h>
02
03 int main()
04 {
05     char s1[ ] = {'K', 'o', 'r', 'e', 'a' };
06     char s2[ ] = {'K', 'o', 'r', 'e', 'a', '\0' };
07
08     printf("%s\r\n%s", s1, s2);
09 }
```

Korea
Korea

02. 배열의 초기화

하나 더 알기

한글이나 한자 문자열을 사용하는 방법

- MBCS(MultiByteCharacter Set)은 유니코드가 나오기 전부터 한글을 비롯한 전 세계의 다양한 문자들을 표시하기 위한 방식임
- 1바이트가 아닌 여러 바이트를 이용하여 글자 하나를 나타냄
- 기존 char 타입을 그대로 이용할 수 있어 호환성이 뛰어나지만, 각 언어의 문자 간에 코드가 중복될 수도 있다는 단점이 있음

02. 배열의 초기화

하나 더 알기

한글이나 한자 문자열을 사용하는 방법

[코드 7-10] char 배열로 한글 문자열 초기화

```
01  #include <stdio.h>
02
03  int main()
04  {
05      char s[ ] = "난생처음 C 프로그래밍";
06
07      printf("%s", s);
08  }
```

난생처음 C 프로그래밍

02. 배열의 초기화

확인문제2

1. 다음과 같이 배열이 정의되어 있을 때 `arr[6]`의 값은 무엇인가?

```
int arr[8] = {0, 1, 2, 3, 4, 5, };
```

2. 다음과 같이 배열이 정의되어 있을 때 `str`의 요소의 개수는 몇 개인가?

```
char str[] = "0123456789";
```

02. 배열의 초기화

LAB 7-2

소수 여부 확인하기

20 이하의 정수를 입력받아서 소수인지 여부를 확인하는 프로그램을 작성해봅시다. 단, 20 이하의 소수로 초기화된 배열을 이용하여 소수 판단의 근거로 활용합니다.

20 이하 정수를 입력하세요.

18

소수가 아닙니다.

- 1 배열 초기화를 통해서 20 이하 소수 배열 `prime`을 정의한다.
- 2 `do~while`문을 이용하여 20 이하 정수를 입력받는다.
- 3 반복문을 통해서 입력받은 수 `n`과 배열의 요소들을 각각 비교하여 같은 경우 소수로 판단하여 출력하고 `return 0;`을 통해서 `main`을 반환하면서 프로그램을 종료한다.
- 4 반복문에서 빠져나온 경우 소수가 아니다.
- 5 대문자 코드에 32를 더하면 소문자 코드이다.

02. 배열의 초기화

LAB 7-2

정답

```
01  #define _CRT_SECURE_NO_WARNINGS
02  #include <studio.h>
03
04  int main()
05  {
06      int prime[ ] = { 2, 3, 5, 7, 11, 13, 17, 19 };
07      int n;
08
09      do
10      {
11          printf("20 이하 정수를 입력하세요.\r\n");
12          scanf("%d", &n);
13      }
14      while(n > 20);
15
```

02. 배열의 초기화

LAB 7-2

정답

```
16     for(int i = 0; i < 8; i++)
17     {
18         if(n == prime[i])
19         {
20             printf("소수입니다.");
21             return 0;
22         }
23     }
24
25     printf("소수가 아닙니다.");
26 }
```

03

다차원 배열

03. 다차원 배열

I. 2차원 배열

- 2차원 배열의 개념
 - 배열이 요소인 배열
- 2차원 배열의 형식

타입 배열명[배열 요소의 개수][배열 요소 속 요소의 개수];

03. 다차원 배열

I. 2차원 배열

- 2차원 배열의 예시

```
Ramen Box[8][5];
```

- Ramen : 배열(박스)의 요소(멀티팩) 속 요소(라면)의 타입
- Box : 배열의 이름
- 8 : 요소(멀티팩)의 개수
- 5 : 요소(멀티팩) 속 요소(라면)의 개수



그림 7-2 라면 박스와 멀티팩의 2차원 배열

03. 다차원 배열

I. 2차원 배열

[코드 7-8] 아스키코드 문자 배열 초기화

```
01  #include <studio.h>
02
03  int main()
04  {
05      int arr[2][3] = { { 11, 12, 13 }, { 21, 22, 23 } }
06
07      for(int i = 0; i < 2; i++)
08      {
09          for(int j = 0; j < 3; j++)
10              printf("%d ", arr[i][j]);
11          printf("\r\n");
12      }
13  }
```

11 12 13

21 22 23

03. 다차원 배열

II. 2차원 배열의 구조

- 2차원 배열은 행렬과 비슷하긴 하나, 실제 메모리 구조가 격자 모양인 것은 아님

| | | |
|-----------|-----------|-----------|
| arr[0][0] | arr[0][1] | arr[0][2] |
| arr[1][0] | arr[1][1] | arr[1][2] |

그림 7-3 행렬 모양으로 설명한 2차원 배열

03. 다차원 배열

II. 2차원 배열의 구조

- 메모리는 논리적인 선형 구조를 지님
- 모든 바이트마다 주소가 부여되어 있음

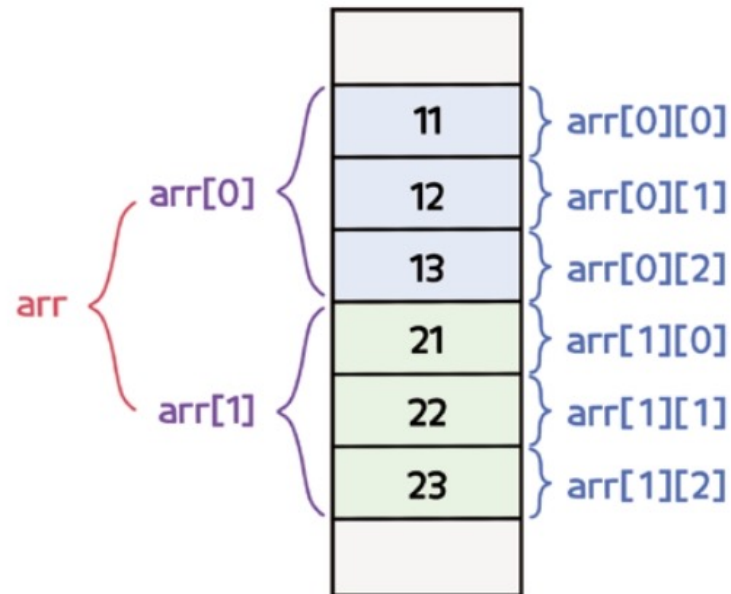


그림 7-4 실제 2차원 배열 메모리 구조

03. 다차원 배열

확인문제3

1. 스팸은 3개 소포장으로 4개씩 한 세트(set)로 팔기도 한다. 스팸의 타입이 SPAM일 때 스팸 한 세트(set)를 2차원 배열로 정의하시오.
2. 스팸 선물세트의 스팸은 가로 3개, 세로 4개로 배치되어 있다. 스팸 선물세트를 1차원 스팸 배열로 표시하고, 가장 왼쪽 위쪽의 스팸과, 가장 오른쪽 아래쪽의 스팸을 인덱스를 통해서 나타내시오.

03. 다차원 배열

LAB 7-3

구구단 게임

구구단 게임 프로그램을 작성해봅시다. 구구단 게임은 2차원 배열 table에 미리 구구단의 결과를 저장하고 입력된 단과 항을 table에서 조회하여 출력하는 방식을 사용하여 입력받은 구구단의 결과를 출력합니다.

구구단의 단과 항을 공백으로 구분하여 입력하세요.

3 4

3 * 4 = 12

- 1 구구단을 2차원 배열로 정의할 경우 배열의 타입은 `int[8][9]`이다.
- 2 이중 반복문을 사용하여 단은 2부터 9까지, 항은 1부터 9까지 반복하면서 배열의 요소에 구구단 계산 값을 저장하고, 출력할 때는 인덱스를 이용하여 배열 요소의 값을 가져온다.

03. 다차원 배열

LAB 7-3

정답

```
01  #define _CRT_SECURE_NO_WARNINGS
02  #include <studio.h>
03
04  int main()
05  {
06      int r, c;
07      printf("구구단의 단과 항을 공백으로 구분하여 입력하세요.\r\n");
08      scanf("%d%d", &r, &c);
09
10      int table[8][9] = { { 0 } };
11      for(int i = 2; i < 10; i++)
12          for(int j = 1; j < 10; j++)
13              table[i-2][j-1] = i * j;
14
15      printf("%d * %d = %d", r, c, table[r-2][c-1]);
16  }
```

04

배열의 크기와 길이

04. 배열의 크기와 길이

I. 배열의 크기와 길이의 관계

- 배열의 크기

- 배열이 차지하는 메모리 영역의 크기(바이트 단위)

- 배열의 길이

- 요소의 개수

- 배열의 크기와 길이 구하기

① 배열의 크기(바이트 단위) = `sizeof(배열)`

② 배열의 크기(바이트 단위) = `sizeof(요소의 타입) * 요소의 개수`



배열의 요소 개수 = `sizeof(배열) / sizeof(요소의 타입)`

04. 배열의 크기와 길이

I. 배열의 크기와 길이의 관계

[코드 7-12] 배열의 길이 구하기

```
01  #include <stdio.h>
02
03  int main()
04  {
05      int arr[8]
06      int size = sizeof(arr);
07      int length = size / sizeof(int);
08
09      printf("배열의 크기:%d\r\n배열의 길이:%d", size, length);
10  }
```

배열의 크기:32

배열의 길이:8

04. 배열의 크기와 길이

I. 배열의 크기와 길이의 관계

[코드 7-13] 배열의 길이, 문자열의 길이

```
01  #include <stdio.h>
02
03  int main()
04  {
05      int arr[] = { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 };
06      int alen = sizeof(arr) / sizeof(arr[0]);
07
08      char str[ ] = "C Programming for the fisrt time";
09      int slen = sizeof(str) / sizeof(str[0]) - 1;
10
11      printf("alen: %d\r\nslen: %d", alen, slen);
12  }
```

alen: 10

slen: 32

04. 배열의 크기와 길이

I. 배열의 크기와 길이의 관계

- 문자열의 길이 구하기

[코드 7-14] strlen 함수로 문자열의 길이 구하기

```
01  #include <stdio.h>
02  #include <string.h>
03
04  int main()
05  {
06      char str[] = "0123456789";    // 10자
07      int slen = strlen(str);
08
09      printf("slen: %d", slen);
10  }
```

slen: 10

04. 배열의 크기와 길이

하나 더 알기

MBCS에는 적용할 수 없는 strlen 함수

[코드 7-15] MBCS 문자열의 길이

```
01 #include <studio.h>
02 #include <string.h>
03
04 int main()
05 {
06     char str[ ] = "난생처음 c 프로그래밍";
07     int slen1 = sizeof(str) / sizeof(str[0]) - 1;
08     int slen2 = strlen(str);
09
10     printf("slen1: %d\r\nstrlen: %d", slen1, slen2);
11 }
```

```
slen1: 21
strlen: 21
```

04. 배열의 크기와 길이

하나 더 알기

MBCS에는 적용할 수 없는 strlen 함수

- MBCS 문자열의 길이를 구하는 방법
 - (1) 문자 배열을 순회하면서 MBCS 문자임을 확인하면서 일일이 셈
 - (2) 유니코드로 변환하여 계산

04. 배열의 크기와 길이

II. 가변 길이 배열

- 가변 길이 배열의 개념

- 배열의 길이로 변수를 사용할 수 있는 것
- C99 이전에는 컴파일 타임에 배열의 길이가 결정되어야 해서 숫자 리터럴이 사용되었음

[코드 7-12] 배열의 길이 구하기

```
01  int main()  
02  {  
03      int arr1[3];    // 문제없음  
04  
05      int N = 3;  
06      int arr2[N]; ← 오류 발생  
07  }
```

04. 배열의 크기와 길이

하나 더 알기

비주얼 스튜디오에서는 지원하지 않는 가변 길이 배열

- 가변 길이 배열은 프로그램 실행 중 오류를 일으킬 수 있어 비주얼 스튜디오에서 의도적으로 지원하지 않음
- [코드 7-16]은 다른 온라인 컴파일러(Online Compiler)를 사용하여 컴파일되는 것을 확인할 수 있음

04. 배열의 크기와 길이

II. 가변 길이 배열

- 가변 길이 배열의 개념

[코드 7-17] 배열 크기의 한계

```
01  #include <stdio.h>
02
03  int main()
04  {
05      int arr[1024 * 1024 * 4] = { 0 };
06      printf("%d", arr[0]);
07  }
```

04. 배열의 크기와 길이

확인문제4

1. 다음 코드의 실행 결과는 무엇이며, 문자열 "0123456789"의 길이와 어떤 차이가 있는지 서술하시오.

```
01  #include <stdio.h>
02
03  int main()
04  {
05      char str[16] = "0123456789";
06      int len = sizeof(str) / sizeof(str[0]) - 1;
07      printf("len: %d", len);
08  }
```

2. 가변 길이 배열이 위험할 수 있을 때는 언제인지 구체적으로 설명하시오.

04. 배열의 크기와 길이

LAB 7-4

문자열의 길이 구하기

다음과 같이 str 배열에 들어간 문자열의 길이를 구하는 코드를 작성해봅시다. 단, strlen을 사용하지 말고 strlen 구현 방식을 이용해야 합니다.

```
char str[128] = "abcdefghijklmnopqrstuvwxyz0123456789";
```

문자열의 길이: 36

반복문을 통해서 문자 배열의 요소를 하나씩 순회하면서 요소의 값이 NULL임을 확인한다. 요소의 값이 NULL(0 또는 '\0')인 경우 문자열이 끝난 위치이므로 그 때의 인덱스가 길이이다.

04. 배열의 크기와 길이

LAB 7-4

정답

```
01  #include <stdio.h>
02
03  int main()
04  {
05      char str[] = "abcdefghijklmnopqrstuvwxyz0123456789";
06
07      for(int i = 0; i < 128; i++)
08      {
09          if(str[i] == '\0')
10          {
11              printf("문자열의 길이: %d", i);
12              break;
13          }
14      }
15  }
```

[실전예제]

문자 배열 합치기

[실전예제] 문자 배열 합치기

[문제]

다음의 문자 배열 2개를 합쳐서 "난생처음 C 프로그래밍"을 저장하는 배열 str을 만들고 출력해봅시다.

```
char str[64];  
char str1[] = "난생처음";  
char str2[] = "C 프로그래밍";
```

실행 결과

난생처음 C 프로그래밍

[실전예제] 문자 배열 합치기

[해결]

1. 목표 문자 배열 `str`의 인덱스로 `si`를 이용하여,
복사할 문자 배열 `str1`, `str2`의 인덱스로 `i`를 사용한다.
2. 반복문을 통해서 `str1`의 요소를 순회하면서 `NULL`이 아닌 경우
`str` 요소에 복사하고, `NULL`을 만날 때는 반복문을 탈출한다.
그 이후 띄어쓰기를 위하여 `str` 요소에 공백(' ')을 넣고 인덱스 `si`는
1 증가시킨 후 다시 반복문을 통해서 `str2`를 `str`에 복사한다.
3. `str`이 이미 종괄호 초기값으로 모든 요소가 `0`으로 초기화
되었으므로 `NULL(0)`으로 차 있는 상태이다. 따라서 기본적으로
`NULL` 종료 문자열이 된다.

[실전예제] 100 이하의 소수 출력하기

[해결]

```
01  #include <stdio.h>
02
03  int main()
04  {
05      char str[64] = {0};
06
07      char str1[ ] = "난생처음";
08      char str2[ ] = "C 프로그래밍";
09
10      int si = 0, i = 0;
11
12      for(i = 0; i < sizeof(str1); i++)
13      {
14          if(str1[i] != '\0')
15          {
16              str[si] = str1[i];
17              si++;
18          }
19          else
```

[실전예제] 100 이하의 소수 출력하기

[해결]

```
20
21         break;
22     }
23     str[si] = ' ';
24     si++;
25
26     for(i = 0; i < sizeof(str2); i++)
27     {
28         if(str2[i] != '\0')
29         {
30             str[si] = str2[i];
31             si++;
32         }
33         else
34             break;
35     }
36
37     printf("%s", str);
38 }
```

Thank you!