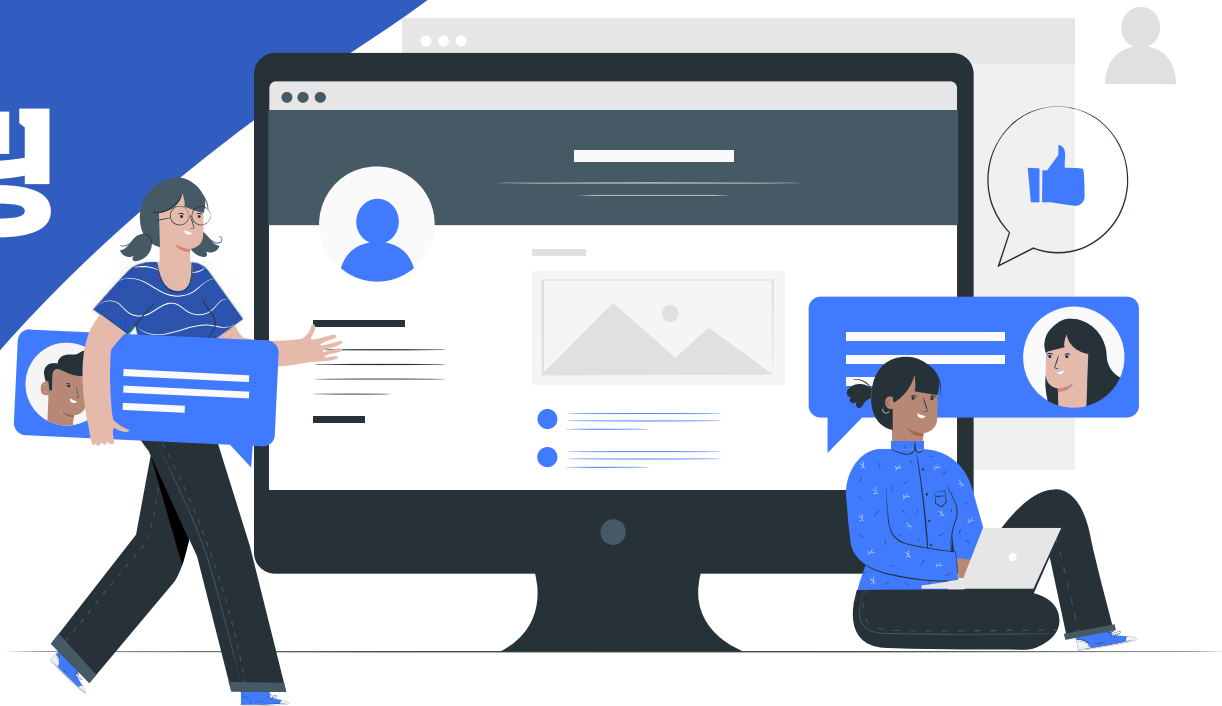


# 8주차 튜터링



**01 배열1**



# 목차

- 배열이란?
- 일반 변수와 배열의 차이
- 배열의 정의 다시 보기
- 배열의 선언 방식
  - 방법 1. 그냥 선언하기
  - 방법 2. 선언과 동시에 초기화하기
  - 방법 3. 배열을 선언할 때 크기 생략하기쓰레기값
- 배열을 초기화 하는방식
  - 방법 1. 중괄호 안에 각 배열의 요소의 초기값을 콤마로 구분하여 지정하기
  - 방법 2. 배열 요소의 첨자에 각각 값을 대입하여 초기화 하기
- 배열을 0으로 초기화 하는 방식
  - 방법 1. 반복문 사용
  - 방법 2. 쉼표 사용
    - 방법 2-1. 쉼표를 노가다로 찍기
    - 방법 2-2. {0,}
- 배열을 사용할 때의 주의해야 할 점

# 배열이란?

- 배열(array)은 연속적인 같은 타입의 변수들의 집합이다.
- 선언 방법은 다음과 같다

**타입** 배열이름**[배열크기];**

- **타입** : 배열 요소로 들어가는 변수의 타입을 명시합니다.
- **배열이름** : 말 그대로 배열의 이름입니다.
- **배열크기** : 해당 배열이 몇 개의 배열 요소를 가지게 되는지 명시합니다.

# 배열이란?

```
● ● ●  
  
#include <stdio.h>  
  
int main()  
{  
    int arr[5] = {1, 33, 47, 102, 155};  
  
    printf("%d\n", arr[0]);  
    printf("%d\n", arr[1]);  
    printf("%d\n", arr[2]);  
    printf("%d\n", arr[3]);  
    printf("%d\n", arr[4]);  
  
    return 0;  
}
```

- 한 번 따라 입력해 보세요~



# 일반 변수와 배열의 차이

- 일반 변수의 선언

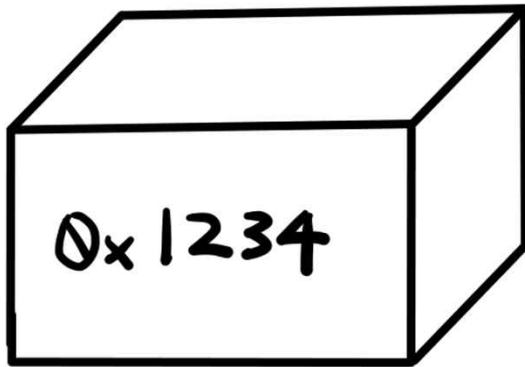
```
int a = 5;
```



# 일반 변수와 배열의 차이

- 일반 변수의 선언

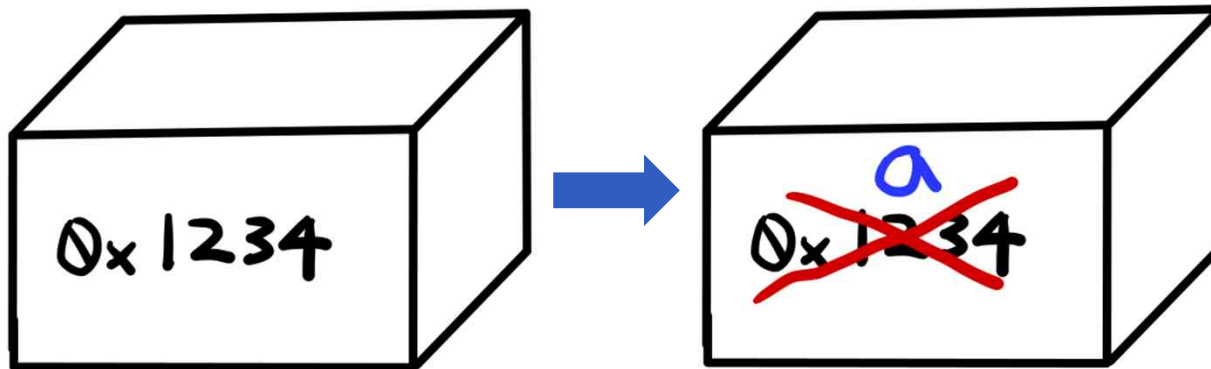
```
int a = 5;
```



# 일반 변수와 배열의 차이

- 일반 변수의 선언

```
int a = 5;
```

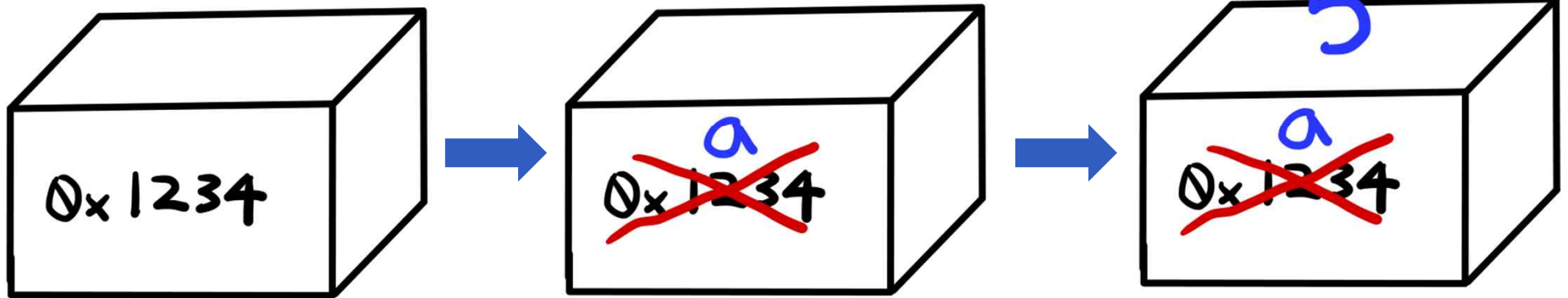




# 일반 변수와 배열의 차이

- 일반 변수의 선언

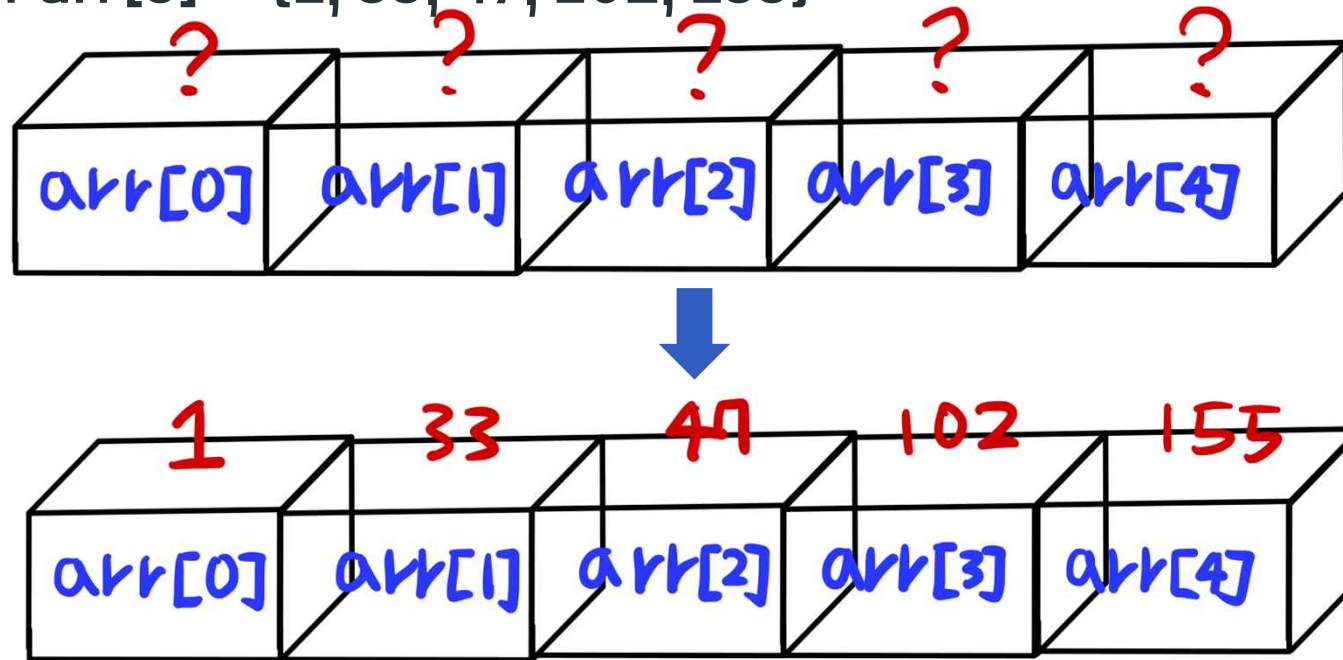
```
int a = 5;
```



# 일반 변수와 배열의 차이

- 배열의 선언

int arr[5] = {1, 33, 47, 102, 155}



# 일반 변수와 배열의 차이

데이터 타입    크기



배열의 요소





```
int arr[5] = {1, 33, 47, 102, 155};
```



배열의 이름



# 배열의 선언 방식

- 배열의 선언 방식에는
  - 크게 세 가지가 있다.
  - **방법 1. 그냥 선언하기**
  - **방법 2. 선언과 동시에 초기화하기**
  - **방법 3. 배열을 선언할 때 크기 생략하기**
- 
- 

# 방법 1. 그냥 선언하기

```
#include<stdio.h>

int main()
{
    int arr[10];
    for (int i = 0; i < 10; i++)
    {
        arr[i] = i;
        printf("%d ", arr[i]);
    }

    return 0;
}

// 결과
// 0 1 2 3 4 5 6 7 8 9
```

## 방법 2. 선언과 동시에 초기화하기

```
#include <stdio.h>

int main()
{
    int arr[5] = {1, 33, 47, 102, 155};

    return 0;
}
```

## 방법 3. 배열을 선언할 때 크기

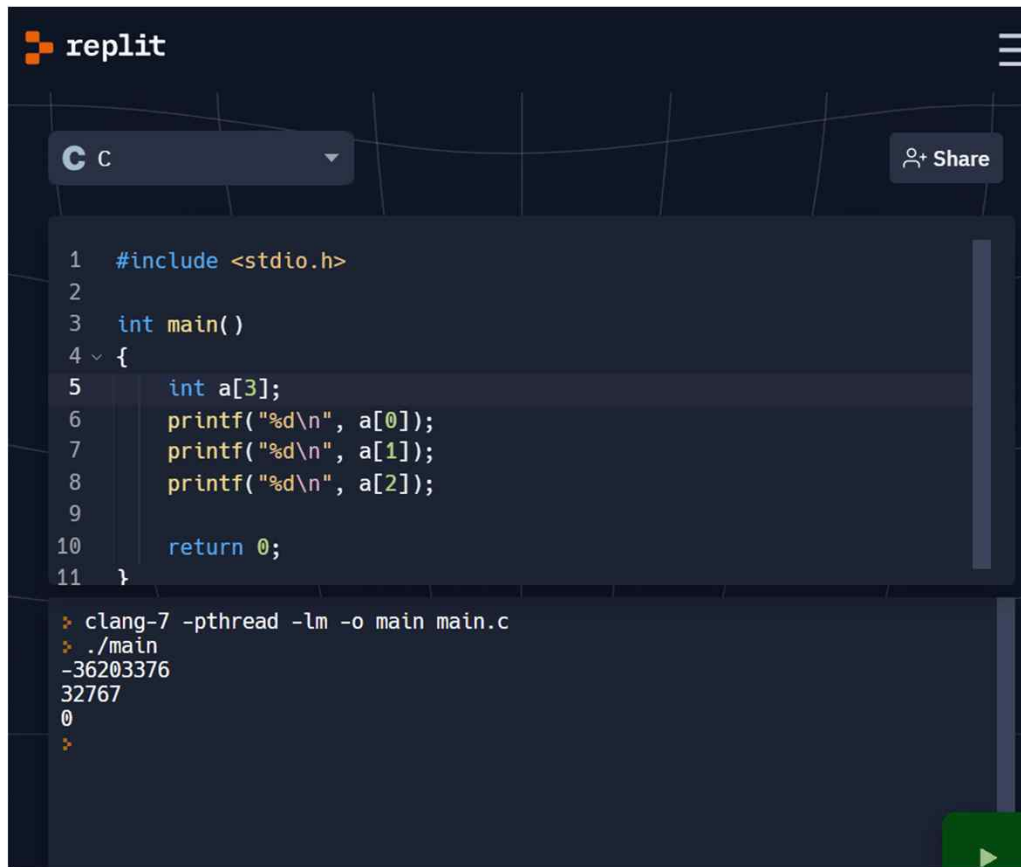
```
#include <stdio.h>

int main()
{
    int arr[] = {1, 33, 47, 102, 155}

    return 0;
}
```

- 배열을 선언할 때 값을 초기화한다면 배열의 크기 생략이 가능하다.
- 초기화를 하지 않을 때는 생략할 수 없다.

# 쓰레기 값 (garbage value)



The screenshot shows a Replit environment with a C program. The code defines an array `a` of size 3 and prints its elements. The output shows memory addresses for `a[0]` and `a[1]`, and `0` for `a[2]`.

```
replit
C
1 #include <stdio.h>
2
3 int main()
4 {
5     int a[3];
6     printf("%d\n", a[0]);
7     printf("%d\n", a[1]);
8     printf("%d\n", a[2]);
9
10    return 0;
11 }
```

```
clang-7 -pthread -lm -o main main.c
./main
-36203376
32767
0
```

- 메모리에 변수가 할당되기 전에는 우리가 모르는 이상한 값이 들어가 있다.
- 이런 값을 쓰레기 값 이라고 한다.



## 쓰레기 값 (garbage value)

- 앞의 슬라이드에서 보았다시피 배열을 선언한 뒤 초기화를 안 하면 쓰레기 값이 들어간다.
- 쓰레기 값이 들어있으면 프로그래머가 값을 넣는 과정에서 실수가 일어날 수 있다.
- 실수를 안 하기 위해서 애초에 선언과 동시에 0으로 초기화하는 습관을 들이면 좋다



# 배열을 초기화 하는 방식

- 배열을 초기화 하는 방식에는
- **방법 1. 중괄호 안에 각 배열의 요소의 초기값을 콤마로 구분하여 지정하기**
- **방법 2. 배열 요소의 첨자에 각각 값을 대입하여 초기화 하기**

... 있다.



# 방법 1. 중괄호 안에 (이하생략) 지정하기

```
#include <stdio.h>

int main()
{
    int arr[5] = {1, 33 , 47, 102, 155};

    return 0;
}
```

## 방법 2. (이하생략) 각각 값을 대입하여 초기화

```
#include <stdio.h>

int main()
{
    int arr[5];

    arr[0] = 35;
    arr[1] = 33;
    arr[2] = 47;
    arr[3] = 102;
    arr[4] = 155;

    return 0;
}
```

# 배열을 0으로 초기화 하는 방식

- 배열을 0으로 초기화 하는 방식에는
- 방법 1. 반복문 사용
- 방법 2. 쉼표 사용
  - 방법 2-1. 쉼표를 노가다로 찍기
  - 방법 2-2. {0,}

... 있다.

# 방법 1. 반복문 사용

```
● ● ●  
  
#include <stdio.h>  
  
int main()  
{  
    int arr[20];  
  
    for(int i = 0; i < 20; i++)  
    {  
        arr[i] = 0;  
    }  
  
    return 0;  
}
```

## 방법 2-1. 십표를 노가다로 찍기

```

#include <stdio.h>

int main()
{
    int arr[20] = {0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0,
                  0, 0, 0, 0, 0};

    return 0;
}
```

## 방법 2-2. {0,}



```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int arr[20] = {0, }
```

```
    return 0;
```

```
}
```



## 방법 2-2. {0,}

```

#include <stdio.h>

int main()
{
    int arr[5] = {3, };
    /*
    만약 {3, 0}으로 찍을 경우에는
    int arr[5] = {3, 0, 0, 0, 0}; 과 같다.
    */

    return 0;
}
```

# 배열을 사용할 때의 주의해야 할 점

- 배열의 첨자는 '1'이 아닌 '0'부터 시작한다.
- 배열명은 그 자체가 배열의 시작 주소를 가리킨다.
- 배열의 선언과 함께 초기화할 경우에는 배열의 크기를 생략한다.
- 문자열을 표한 할 경우
  - NULL문자('\0')를 포함해서 배열의 크기를 선언
  - scanf()함수에서 앰퍼센트(&)를 선언하지 않음
- 배열의 크기와 초기값의 관계
  - "초기값 > 배열의 크기"일 경우에는 에러 발생
  - "초기값 < 배열의 크기"일 경우에는 나머지는 0으로 초기화



END