프로그래밍 기초

scanf() 활용

scanf() 함수 심화 내용

- scanf(): **공백 문자를 기준으로 입력이 분리됨**
  - 공백 문자: space, tab('\t'), carriage return('\n') 등

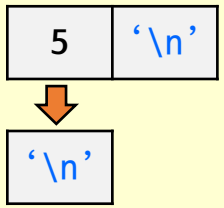
- scanf("%d", &num) **호출**
  - 입력 버퍼에 남은 공백 문자를 무시하고 숫자를 읽음
  - 입력 예 | 공백 | 10 | '\n' | -> 맨 앞의 공백 문자 무시하고 10을 읽음

- scanf("%c", &ch) **호출**
  - 앞에 new line 문자('\n')이 남아 있으면, new line 문자('\n')를 읽음
    - "%c" 옵션의 경우, 공백 문자라도 문자로 인식하기 때문
  - 입력 예 | '\n' | y | '\n' | -> scanf("%c", …)인 경우, 첫 번째 '\n'을 읽음
  - scanf("%c", &ch)를 호출하기 직전에는 입력 버퍼가 비어 있어야 됨

scanf() 호출 후 new line('\n') 제거

- **숫자와 문자를 각각 scanf() 함수로 입력 받음**
  - 첫 번째 scanf("%d", &num)는 정상 동작함
  - 두 번째 scanf("%c", &ch)는 정상 동작 하지 않음
  - 숫자 입력 후 Enter키를 누르면 개행문자('\n')가 입력됨

```c
#include <stdio.h>

int main()
{
    int num;
    char ch;

    printf("Input a number: ");
    scanf("%d", &num); // 첫 번째 scanf()
    printf("typed number: %d\n", num);

    printf("Type y or n: ");
    scanf("%c", &ch); // 두 번째 scanf() 입력 안됨
                      // 이전에 입력된 Enter키를 읽어옴
    printf("typed char: %c\n", ch);
    return 0;
}
```

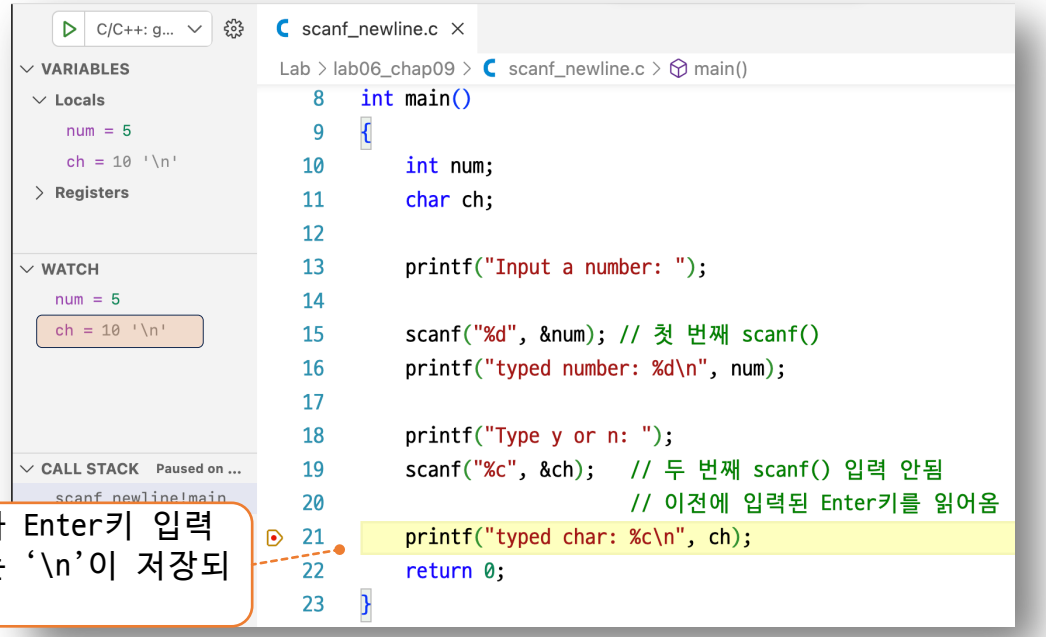


- 첫 번째 scanf()에서 숫자와 Enter키 입력
- 두 번째 scanf()에서 ch에는 '\n'이 저장되어 있음

실행 결과

```
Input a number: 5 Enter
typed number: 5
Type y or n: typed char:
```

'y'나 'n' 입력 안됨



scanf() 호출 이후 개행 문자('\n') 제거 방법

- **1) scanf("%d", &n); 호출 후 char ch = getchar(); 호출**
  - getchar() 함수가 입력 버퍼에 남아 있는 '\n'을 읽고 입력 버퍼에서 제거
  - 권장 방법

- **2) scanf("%d%c", &n, &ch); 호출**
  - ch에 '\n' 문자 저장되며 ch는 사용하지 않음

- **3) scanf(" %c", &ch);**
  - " %c" 앞에 공백 추가
  - Enter('\n') 키를 공백으로 인식해서 '\n'을 무시하고 다음 문자를 읽음

scanf() 호출 후 new line('\n') 제거

방법 #1

```c
#include <stdio.h>

int main()
{
    int num;
    char ch;

    printf("Input a number: ");
    scanf("%d", &num);
    printf("typed number: %d\n", num);

    // 방법 1
    ch = getchar();

    printf("Type y or n: ");
    scanf("%c", &ch);
    printf("typed char: %c\n", ch);
    return 0;
}
```

- getchar() 호출로 버퍼에 남아 있는 Enter키를 읽음

방법 #2

```c
#include <stdio.h>

int main()
{
    int num;
    char ch;

    printf("Input a number: ");
    scanf("%d%c", &num, &ch); // 방법 2
    printf("typed number: %d\n", n);

    printf("Type y or n: ");
    scanf("%c", &ch);
    printf("typed char: %c\n", ch);
    return 0;
}
```

- 정수 및 문자(Enter키)를 입력 받고 처리하지 않음

```
Input a number: 5
number: 5
Type y or n:  y
typed char: y
```

scanf() 호출 후 new line('\n') 제거

방법 #3

```c
#include <stdio.h>

int main()
{
    int num;
    char ch;

    printf("Input a number: ");
    scanf("%d", &num);
    printf("typed number: %d\n", num);

    printf("Type y or n: ");
    scanf(" %c", &ch);
    printf("typed char: %c\n", ch);
    return 0;
}
```

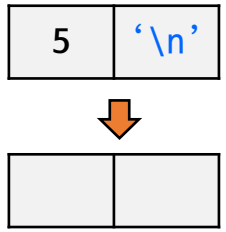
입력 버퍼에 남아 있는 모든 공백 문자(tab, space, Enter등) 무시

```
Input a number: 5
number: 5
Type y or n:  y
typed char: y
```

scanf() 호출 후 new line('\n') 처리 과정 예제

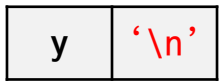
```c
do
{
    printf("Input an integer number: ");
①   scanf("%d", &num);
    ch = getchar();

    printf("%d: ", num);
    if (evenodd(num) == 0)
        printf("Even number \n");
    else
        printf("Odd number \n");

    printf("absolute(%d): %d\n", num, absolute(num));
    printf("-------------------------\n");
    printf("Again? (y/n): ");

②   scanf("%c", &ch);
    //getchar();        // 없어도 문제 없음

} while (ch == 'y' || ch == 'Y');
```

Input an integer number: 5 Enter↵

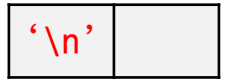


① scanf("%d", &num); getchar(); 이후

Again? (y/n): y Enter↵

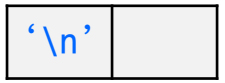


② scanf("%c", &ch); 이후



Input an integer number: 10 Enter↵

① scanf("%d", &num);



scanf("%d")로 숫자를 입력 받으면
-첫 번째 공백 문자('\n')를 무시하고 정수값만 읽음