4.6 커맨드 라인

개요

프로그램을 **명령줄(Command-line)**에서 실행시킬 때, 보통 **./프로그램_이름** 같은 명령어로 실행했을 것입니다. C에서는 프로그램의 **명령행 인자(command-line arguments)**들을 명시할 수 있고, **명령줄에 인자들을 명시하여 사용자가 프로그램의 main 함수에 인자들을 전해줄 수 있게 합니다.** 이 기능은 GetString처럼 프로그램이 실행 중일 때 입력값을 전달받는 방법 대신 사용할 수 있습니다.

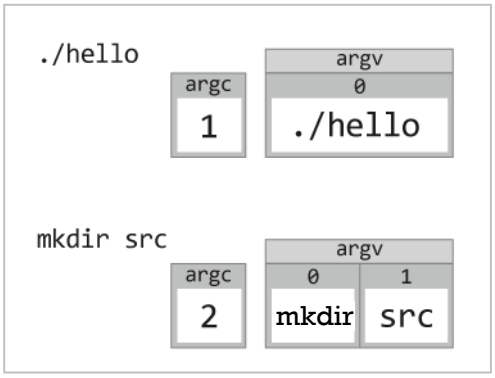
핵심개념

* 명령줄
* 명령행 인자
* argc
* argv

argc, argv

여러분이 써보았을 법한 make, cd, clang, mkdir와 같은 명령어 프로그램들은 모두 명령행 인자들을 받습니다. C에서 **명령행 인자**들은 **main 함수에 입력값으로 전달**됩니다. 하지만 우리가 이전에 작성했던 main 함수에서는 void를 써주었기 때문에 아무런 인자도 받지 않았습니다.

명령행 인자를 받기 위해서는, main 함수가 **정수형**인 **argc**와 **문자열 배열**인 **argv**라는 두 인자를 받아야 합니다. **argc**는 argument count의 약자로, **명령줄에 전달되는 인자의 수**를 나타냅니다. 공백으로 구분되는 각 단어는 하나의 인자로 인식되고, ./hello와 같이 프로그램을 호출하는 것도 하나의 인자로 인식됩니다. **argv**는 argument vector의 약자로, **인자들 그 자체를 나타내는 배열**입니다. 배열의 각 값은 문자열입니다.



▲ 〈그림 1〉

〈그림 1〉을 통하여 argc와 argv의 의미에 대해서 살펴봅시다. ./hello와 같이 프로그램을 인자 없이 실행시킬 때 argc와 argv를 살펴보면, 프로그램을 호출하는 것만이 유일한 인자이기 때문에 argc는 1이 됩니다. 반면, argv는 인덱스 0에 문자열 ./hello를 원소 하나만 갖고 있는 배열이 될 것입니다.

mkdir src와 같이 인자가 있는 프로그램을 실행시킬 때 argc와 argv를 보면, 명령줄에 두 개의 인자가 전달되어 argc는 2가 될 것이고, argv는 인덱스 0에는 문자열 mkdir, 인덱스1에는 문자열 src로, 두 개의 원소를 가지고 있는 배열이 될 것입니다.

명령행 인자 사용하기

```
1  #include <stdio.h>
2
3  int main(int argc, char* argv[] )
4  {
5      for (int i = 0; i < argc; i++)
6      {
7          printf("%s\n", argv[i]);
8      }
9  }
```

▲ 〈코드 1〉

〈코드 1〉은 명령행 인자를 받는 프로그램입니다. 3행에서 main 함수가 argc와 argv 인자를 포함하는 것으로 바뀌었습니다. argv의 크기는 3행에서 명시하지 않았으므로 크기와 상관없이 그 어떤 배열이나 main 함수로 전달될 수 있습니다.

main 함수 안에서, 프로그램은 배열을 인덱스 0부터 시작하여 하나씩 올라가며 i가 argc보다 작을 때까지 순환합니다. 그리고 **접근 가능한 가장 큰 argv의 인덱스는 argc-1이기 때문에** 그 곳에서 멈추는 것이 중요합니다. 배열이 인덱스 0부터 시작하기 때문이죠. 매번 순환할 때마다 프로그램은 argv[i]를 출력합니다. 이 프로그램의 실행 결과를 보면, 각 명령행 인자가 한 줄씩 출력됩니다.