

챕터3. 문자열

1절. 문자열

Windows 버전의 2권 3챕터 1절과 동일

2절. main 함수의 매개변수

핵심 키워드	명령 인수
여기서는 무얼 배울까	
지금까지의 모든 프로그램에서의 main 함수에는 매개변수가 없었다. main 함수의 매개변수는 존재하지 않아서 사용하지 않았던 것이 아니라 사용하지 않았기에 생략했던 것이다. main 함수로 받는 매개변수는 프로그램을 실행하는 시점에 몇 개의 문자열들을 프로그램에 넘겨줄 수 있는데, 이를 명령 인수 라고 한다. 명령 인수를 이용하면 프로그램 실행 이후에 추가적인 키보드 입력 없이도 원하는 입력에 대한 처리를 수행하게 할 수 있다. 이번 절에서는 이러한 명령 인수를 다뤄보는 시간을 갖는다.	

DELL 2023/08/22 13:33

기초 용어 정리
명령 인수
프로그램을 실행하는 시점에 넘겨줄 수 있는 문자열들

1. 명령 인수

손으로 익히는 코딩
<pre>#include <stdio.h> int main(int argc, char* argv[]) { for (int i = 0; i < argc; i++) printf("%s\n", argv[i]); }</pre>

명령 인수를 main 함수에서 사용하려면 매개변수로 문자열 배열의 크기와 문자열 배열을 추가해야 한다. 매개변수의 이름은 자유롭게 지정할 수 있지만 관습적으로 argc(argument count), argv(argument vector)를 사용한다.

DELL 2023/09/01 04:34

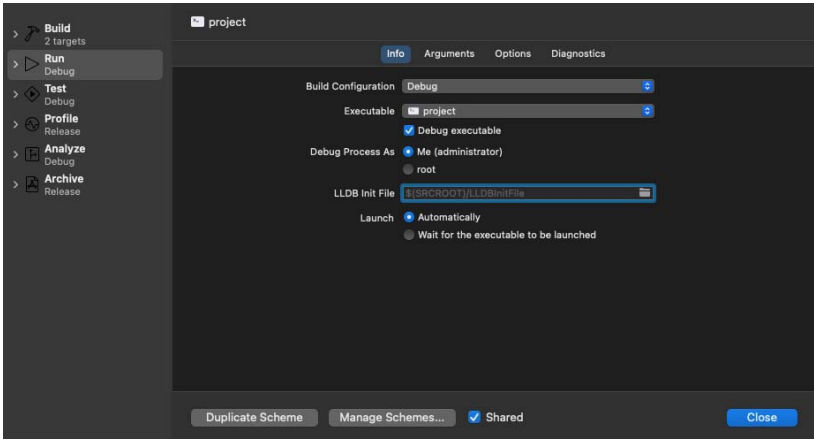
코멘트
argv의 자료형은 앞선 문자열 배열의 자료형과는 달라 보이지만 이 역시 문자열 배열을 나타냅니다. 여기서 사용하는 기호의 의미는 포인터에서 자세히 다룹니다.

```
/Users/administrator/Library/Developer/Xcode/DerivedData/project-axffnjnykipppcclaafhmptrwhl/Build/Products/Debug/project
Program ended with exit code: 0
```

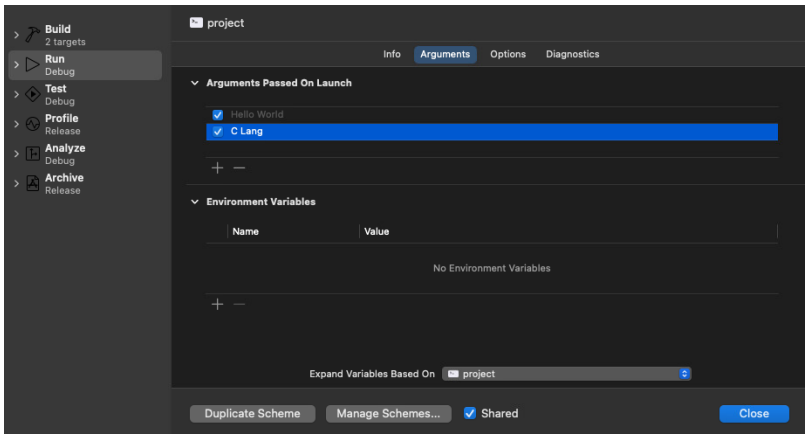
위 예제 프로그램을 작성하고 실행시키면 이 프로그램의 경로와 이름만이 출력된다. 명령 인수는 기본적으로 배열의 첫 원소에는 프로그램의 경로와 이름이 들어가고 그 뒤부터 프로그램을 실행할 때 전달한 문자열들이 저장된다. 지금은 아무런 문자열 없이 바로 실행만 하였기에 프로그램의 이름만이 나오게 된 것이다.

(1) 명령 인수 추가하기

Xcode에서 명령 인수를 추가하여 프로그램을 실행하기 위해선 프로젝트에서 명령 인수를 추가하는 작업이 필요하다.



① Xcode에서 프로그램을 실행할 때 명령 인수를 넣으려면 먼저 **Command + <**로 Edit Scheme 창으로 들어간다.



② Arguments의 Arguments Passed On Launch에서 +를 눌러 명령 인수로 전달하고자 하는 문자열을 추가한다.

```
/Users/administrator/Library/Deve
Hello
World
C
Lang
Program ended with exit code: 0
```

설정이 완료된 후 다시 프로그램을 실행하면 추가한 명령 인수가 출력된다. 위 예시에선 Hello World와 C Lang을 추가했는데, 명령 인수는 공백 문자를 기준으로 잘라 문자열들을 구성하므로 Hello, World, C, Lang 총 4개의 문자열이 추가되었다.

2. 명령 인수의 활용

명령 인수를 이용하여 프로그램을 만들었다면 프로그램 실행 이후에 scanf()로 입력받지 않아도 내가 원하는 입력값에 대해 처리하도록 만들어줄 수 있다.

(1) 문자열 비교하기

두 문자열이 같은지 비교하는 방법은 간단하다. 첫 번째 조건은 두 문자열의 길이가 같아야 한다는 것이고, 두 번째 조건은 두 문자열을 구성하는 각각의 글자가 모두 일치해야 한다는 것이다.

손으로 익히는 코딩

```
#include <stdio.h>

int main(int argc, char* argv[])
{
    int isSame = 0;
    for (int i = 0;; i++)
    {
        if (argv[1][i] == '\0' && argv[2][i] == '\0')
        {
            isSame = 1;
            break;
        }
        else if (argv[1][i] == '\0' || argv[2][i] == '\0' || argv[1][i]
!= argv[2][i])
        {
            isSame = 0;
            break;
        }
    }

    printf("%d", isSame);
}
```

위의 예제는 명령 인수로 두 문자열을 받고, 두 문자열이 같다면 isSame이 1, 다르다면 0

이 나오도록 연산을 수행한다.

```
for (int i = 0;; i++)
```

문자열의 길이는 곧바로 알 수 없으므로 조건을 배워둔 채로 for문을 시작한다. i는 현재 비교하고 있는 글자의 위치이다.

```
if (argv[1][i] == '\0' && argv[2][i] == '\0')
{
    isSame = 1;
    break;
}
```

현재 보고 있는 글자가 두 문자열 모두 널 문자라면 이 둘은 같은 문자열이다. 중간에 다른 문자가 나오는 즉시 반복을 멈출 것이기에 널 문자까지 도달했다는 의미는 그 앞에 있는 모든 글자가 동일하다는 의미이고, 또 글자 수까지 똑같다는 뜻이기 때문이다.

```
int isSame = 0;
else if (argv[1][i] == '\0' || argv[2][i] == '\0' || argv[1][i]
!= argv[2][i])
{
    isSame = 0;
    break;
}
```

1번 문자열이 먼저 끝나거나, 2번 문자열이 먼저 끝나거나, 1번 문자열과 2번 문자열의 글자가 서로 다르다면 즉시 반복을 종료한다. 이 세 경우에 해당하면 두 문자열은 서로 다르기에 isSame을 0으로 설정하고, 그렇지 않다면 지금까지 확인한 모든 글자는 동일하므로 그 뒤를 더 반복하여 검사를 진행한다.

(2) 알파벳 빈도수 세기

정수의 빈도수를 배열에 저장해서 사용하듯 알파벳의 빈도수 또한 배열에 저장할 수 있다. 알파벳 a, b, c가 순서대로 0, 1, 2의 인덱스로 나타내는 것이 효과적이므로 문자에 알파벳 a를 빼서 배열의 인덱스를 구할 수 있게 한다.

손으로 익히는 코딩

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

```
int main(int argc, char* argv[])
```

```

{
    int cnt[26] = { 0, };

    for (int i = 1; i < argc; i++)
    {
        for (int j = 0; argv[i][j] != '\0'; j++)
        {
            if ('a' <= argv[i][j] && argv[i][j] <= 'z')
                cnt[argv[i][j] - 'a']++;
        }
    }

    for (int i = 0; i < 26; i++)
        printf("%c : %d\n", 'a' + i, cnt[i]);
}

```

위의 예제에서는 알파벳 소문자의 빈도수만을 셸지만 몇 개의 조건문과 수식을 이용해 다른 문자에 대한 빈도수도 셀 수 있다.

```
int cnt[26] = { 0, };
```

cnt 배열은 알파벳 소문자 26자의 빈도수를 저장하기 위한 배열이다. 0번 인덱스부터 순서대로 a, b, c, ... 의 빈도수가 저장된다.

```

for (int i = 1; i < argc; i++)
{
    for (int j = 0; argv[i][j] != '\0'; j++)

```

명령 인수로 들어올 모든 문자열에 대해서 각각의 글자를 반복하는 부분이다. argv[i]는 명령 인수 중 하나의 문자열을 의미하므로 argv[i][j]는 그 문자열을 이루는 한 문자이다.

```

    if ('a' <= argv[i][j] && argv[i][j] <= 'z')
        cnt[argv[i][j] - 'a']++;

```

argv[i][j]가 알파벳 소문자라고 한다면 여기에 문자 a를 빼서 a로부터 얼마나 떨어진 글자인지를 구할 수 있다. 이 값이 cnt 배열의 인덱스로 사용된다.

```
for (int i = 0; i < 26; i++)  
    printf("%c : %d\n", 'a' + i, cnt[i]);
```

글자를 모두 셴 이후 이를 출력하기 위한 반복문이다. 알파벳 26자를 모두 출력하는데, 0번 인덱스를 다시 알파벳 a로 바꿔주기 위해 문자 a에 인덱스를 더하여 그 문자를 구한다.

3절. 표준 문자열 함수

Windows 버전의 2권 3챕터 3절과 동일

연습문제 : Windows 버전과 동일

한 장으로 요약하기 : Windows 버전과 동일

[1쪽] [메모:1] DELL

2023/08/22 13:33

기초 용어 정리

명령 인수

프로그램을 실행하는 시점에 넘겨줄 수 있는 문자열들

[1쪽] [메모:2] DELL

2023/09/01 04:34

코멘트

argv의 자료형은 앞선 문자열 배열의 자료형과는 달라 보이지만 이 역시 문자열 배열을 나타냅니다.

여기서 사용하는 기호의 의미는 포인터에서 자세히 다룹니다.