C언어

2019.06.17 월

1주차에서 배운 내용

- 프로그래밍 언어 (Low-level, Compiler,High-level)
- 형 (type)
- 함수 (function)
- 연산자
- printf()와 scanf()

2주차에서 배운 내용

- 다양한 서식문자 (%d, %f, %c, %s...)
- ASCII code
- 반복문 - for

지난 연습문제

1. 프로그램 사용자로부터 두 개의 정수를 입력받는다. 그리고 두 정수 사이의 수를 모두 뺀 결과를 출력하는 프로그램을 작성한다. 예를 들어, 3과 5가 입력되면, 3+4+5의 12가 답이다.

2. 계승(factorial)을 계산하는 프로그램을 작성해보자.

    - n! = 1 X 2 X 3 X … n
    - 예를 들어 3! = 3 X 2 X 1 = 6

3. 1번을 함수로 분리해서 구현해보자.

오늘 할 내용

- 반복문
  - while
  - do~ while
    - 반복문의 중첩
- 분기문
  - if
  - if~ else~

반복문 - While

```c
while (num < 5)
{
    printf("Hello world! %d\n", num);
    num = num + 1;
}
```

- While: ~ 동안

반복문 - While

```c
while (num < 5)
{
    printf("Hello world! %d\n", num);
    num = num + 1;
}
```

```c
while (조건)
{
    내용
}
```

- **조건**을 만족하는 동안, **내용**을 수행한다

예제

1. 9단을 while문을 써서 출력해보자. 9x1=9, 9x2=18 … 9x9=81이 출력이 되도록 한다.

2. 사용자에게 입력받은 단의 구구단을 출력하되, 역순으로 출력한다. 예를 들어 3을 입력받았다면, 3x9=27, 3x8=24 … 3x1=3이 출력이 되도록 한다.

반복문 - Do~While~

- **최소 한번은 무조건 실행을 해야 할 때 사용한다.**
- 조건을 먼저 검사하고 내용을 수행하는 while과 달리, **내용을 수행을 먼저 하고 조건을 검사**한다. 때문에 최소 한번은 무조건 실행이 된다.
-

```c
do {
    printf("Hello world!\n");
    num = num + 1;
} while(num < 3);
```

```
do {
    내용
} while(조건)
```

반복문 - Do~While~

- 최소 한번은 무조건 실행을 해야 할 때 사용한다.
- 조건을 먼저 검사하고 내용을 수행하는 while과 달리, **내용을 수행을 먼저 하고 조건을 검사**한다. 때문에 최소 한번은 무조건 실행이 된다.
-

```c
do {
    printf("Hello world!\n");
    num = num + 1;
} while(num < 3);
```

1

2

```
do {
    내용
} while(조건)
```

예제

```
while (num<10)
{
    printf("%dx%d=%d \n", dan, num, dan*num);
    num = num + 1;
}
```

```
do
{
    printf("%dx%d=%d \n", dan, num, dan*num);
    num = num + 1;
} while(num<10);
```

예제

1. 9단을 do~while문을 써서 출력해보자. 9x1=9, 9x2=18 … 9x9=81이 출력이 되도록 한다.

2. 사용자에게 입력받은 단의 구구단을 출력하되, 역순으로 출력한다. 예를 들어 3을 입력받았다면, 3x9=27, 3x8=24 … 3x1=3이 출력이 되도록 한다.

반복문의 중첩

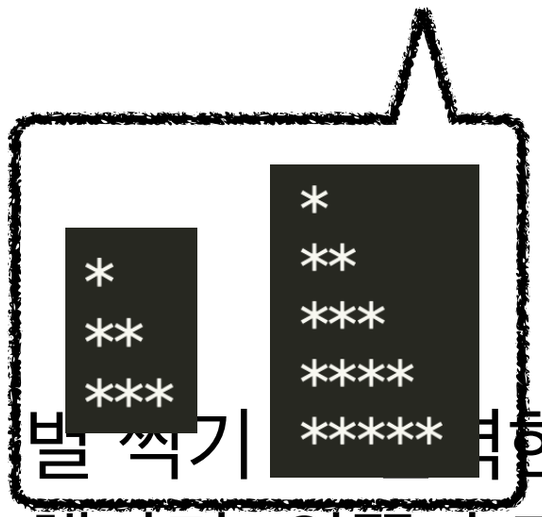
```
while (i < 10) {
    for (j = 1; j < 10; j++) {
        printf("%dx%d=%d\n", i, j, i*j);
        i = i + 1;
    }
}
```

```
for (i = 1; i < 10; i++) {
    for (j = 1; j < 10; j++) {
        printf("%dx%d=%d\n", i, j, i*j);
    }
}
```

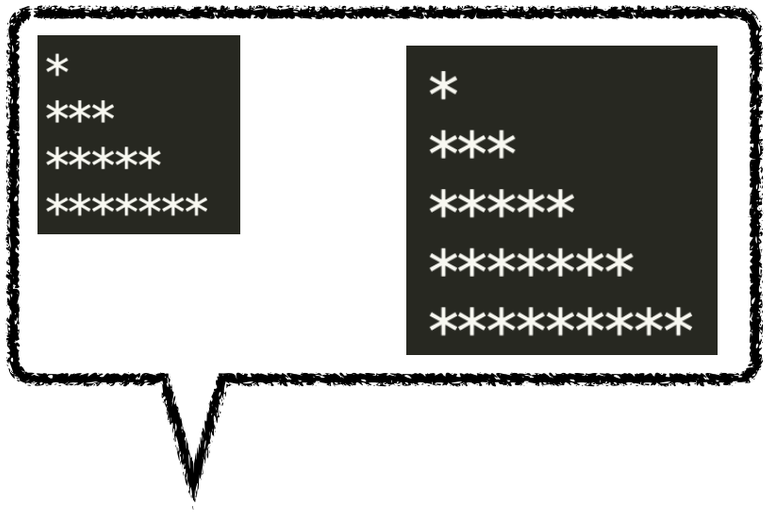
2*1 = 2	3*1 = 3	…	9*1 = 4
2*2 = 4	3*2 = 6	…	4*2 = 8
2*3 = 6	3*3 = 9	…	4*3 = 12
…	…	…	…
2*9 = 9	3*9 = 27	…	9*9 = 36

I * j = i*j
i= 2~9
j= 1~9

예제



- 별 찍기 [illegible]력한 숫자 만큼 별을 찍어주려고 한다. 만약 3을 입력 했다면, 왼쪽과 같은 결과가 나온다. 만약 5를 입력 했다면 오른쪽과 같은 결과가 나온다.



- 별 찍기 2. 입력한 숫자 만큼, 단 2개씩의 간격을 두어 별을 찍어주려고 한다. 만약 7을 입력 했다면, 왼쪽과 같은 결과가 나온다. 만약 9를 입력 했다면 오른쪽과 같은 결과가 나온다.

분기문 - if

```c
if (num%2 == 0) {
    printf("%d", num);
}
```

```c
if (조건) {
    내용
}
```

- 조건이 **참**이면, **내용**을 수행한다

분기문 - if else

```c
if (num%2 == 0) {
    printf("2의 배수 입니다");
} else if (num%3 == 0) {
    printf("3의 배수 입니다");
} else if (num%5 == 0) {
    printf("5의 배수 입니다")
}
```

```c
if (조건 1) {
    내용 1
} else if (조건 2) {
    내용 2
} else if (조건 3) {
    내용 3
}
```

- 조건 1이 **참**이면 **내용 1**을 수행한다
- 조건 2가 **참**이면 **내용 2**를 수행한다
- 조건 3이 **참**이면 **내용 3**을 수행한다

예제

1. 입력받은 숫자까지 홀수의 합을 더한다. 예를 들어, 6을 입력 받으면 1+3+5인 9를 출력한다

2. 입력받은 숫자까지 3의 배수의 합을 더한다. 예를 들어, 10을 입력 받으면 3+6+9인 18을 출력한다

과제

Due date: 06/21 금

- **For**

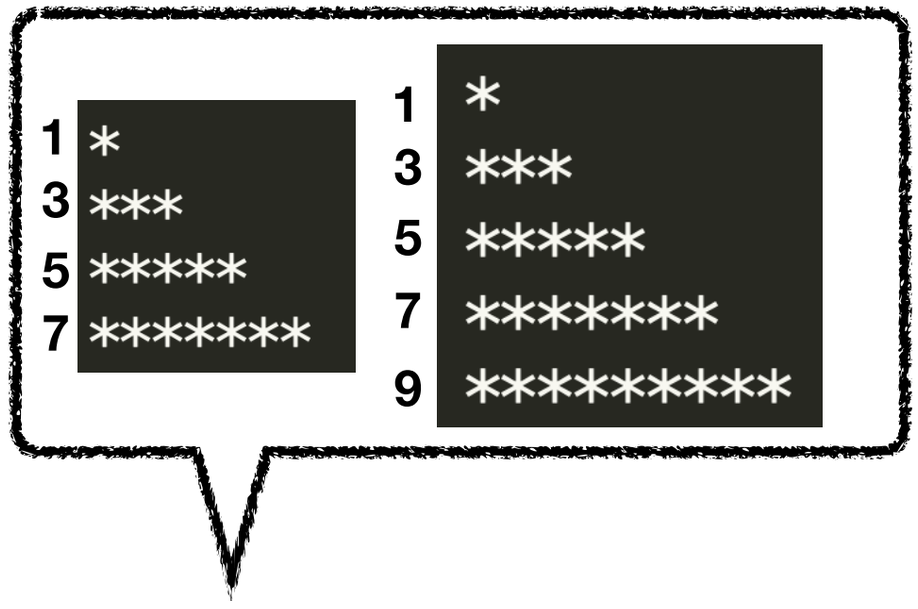
2. 계승(factorial)을 계산하는 프로그램을 작성해보자.
   - n! = 1 X 2 X 3 X … n
   - 예를 들어 3! = 3 X 2 X 1 = 6

- **While**

2. 사용자에게 입력받은 단의 구구단을 출력하되, 역순으로 출력한다. 예를 들어 3을 입력받았다면, 3x9=27, 3x8=24 … 3x1=3이 출력이 되도록 한다.

과제

Due date: 06/21 금



- **반복문의 중첩**

  별 찍기 2. 입력한 숫자 만큼, 단 2개씩의 간격을 두어 별을 찍어주려고 한다. 만약 7을 입력 했다면, 왼쪽과 같은 결과가 나온다. 만약 9를 입력 했다면 오른쪽과 같은 결과가 나온다.

- **If**

  2. 입력받은 숫자까지 3의 배수의 합을 더한다. 예를 들어, 10을 입력 받으면 3+6+9인 18을 출력한다