



C프로그래밍

Lecture 07. 반복 실행을 명령하는 반복문

동덕여자대학교
데이터사이언스 전공
권 범

목차

- ❖ 01. while문에 의한 문장의 반복
- ❖ 02. do~while문에 의한 문장의 반복
- ❖ 03. for문에 의한 문장의 반복
- ❖ 04. 연습 문제

01. while문에 의한 문장의 반복

02. do~while문에 의한 문장의 반복

03. for문에 의한 문장의 반복

04. 연습 문제

01. while문에 의한 문장의 반복

❖ 반복문이란? (1/2)

“Hello, World!”라는 메시지를 10번 출력하고 싶다.
어떻게 하면 좋을까요?

지금까지 우리가 공부했던 방법만을 가지고 이야기 한다면,
printf 함수 호출 문장을 10번 입력됩니다.

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World\n");
    printf("Hello, World\n");
    ...
    printf("Hello, World\n");
    printf("Hello, World\n");

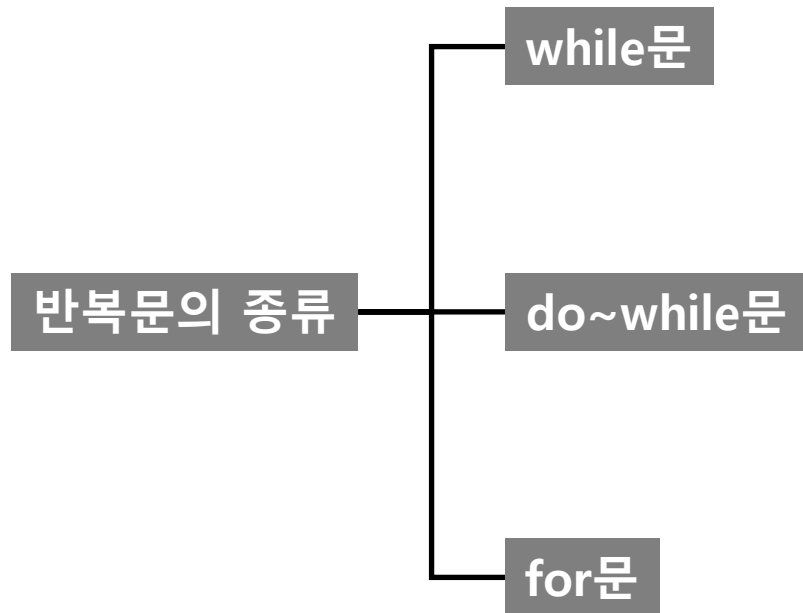
    return 0;
}
```

- ✓ 10번이 아니라 100번 출력하고 싶다고 하면 어떻게 될까요?
- ✓ 이러한 방법은 다소 비효율적임을 알 수 있습니다.

01. while문에 의한 문장의 반복

❖ 반복문이란? (2/2)

- 하나 이상의 문장을 두 번 이상 반복 실행하기 위해서 구성하는 문장



01. while문에 의한 문장의 반복

❖ while문에 의한 문장의 반복 (1/2)

```
1  /* hello_while1.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int num = 0;
7
8      while (num < 5)
9      {
10         printf("Hello, World! %d\n", num);
11         num++;
12     }
13
14     return 0;
15 }
```

while 반복문

중괄호 {} 내부 반복 영역

Hello, World 0
Hello, World 1
Hello, World 2
Hello, World 3
Hello, World 4

반복의 목적이 되는 대상
변수 num은 반복의 횟수를 조절하기 위한 것입니다.

01. while문에 의한 문장의 반복

❖ while문에 의한 문장의 반복 (2/2)

```
1  /* hello_while2.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int num = 0;
7
8      while (num < 5)
9          printf("Hello, World! %d\n", num++);
10
11     return 0;
12 }
```

반복의 대상이 한 문장이면
중괄호 { } 생략이 가능합니다.

Hello, World 0
Hello, World 1
Hello, World 2
Hello, World 3
Hello, World 4

따라서 아래와 같은 표현도 가능합니다.

```
while (num < 5)
    printf("Hello, World! %d\n", num), num++;
```

01. while문에 의한 문장의 반복

❖ 반복문 안에서도 들여쓰기를 합니다

들여쓰기를 하지 않은 것

```
#include <stdio.h>

int main(void)
{
    int num = 0;
    while (num < 5)
    {
        printf("Hello, World! %d\n", num);
        num++;
    }
    return 0;
}
```

들여쓰기를 한 것

```
#include <stdio.h>

int main(void)
{
    int num = 0;
    while (num < 5)
    {
        printf("Hello, World! %d\n", num);
        num++;
    }
    return 0;
}
```

들여쓰기를 한 것과 하지 않은 것 차이가
쉽게 눈에 들어옵니다.

01. while문에 의한 문장의 반복

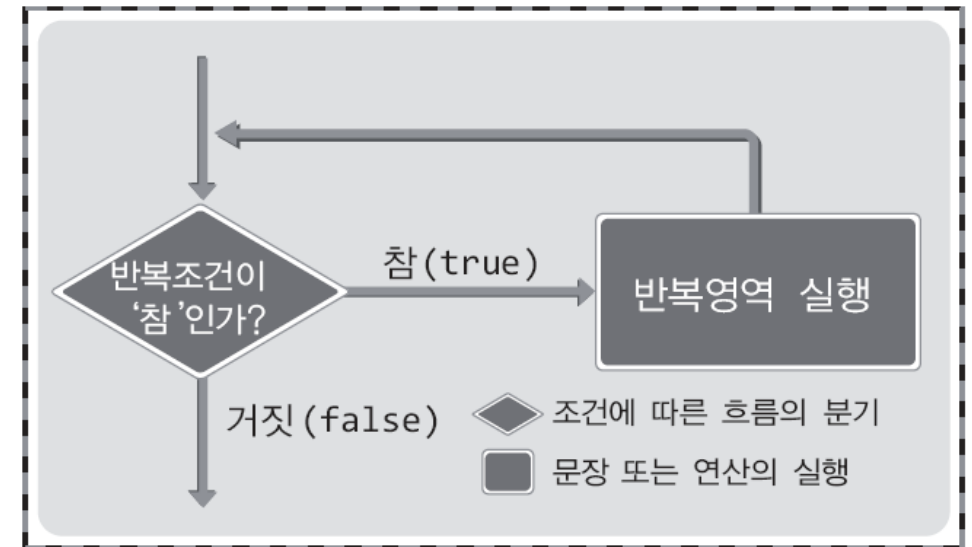
❖ while문의 구성과 실행 흐름의 세세한 관찰

```
#include <stdio.h>

int main(void)
{
    int num = 0;
    while (num < 3) // 3회 반복
    {
        printf("Hello, World! %d\n", num);
        num++;
    }
    ...
}
```

반복의 과정은 어떻게 될까요?

while문의 Flow Chart(순서도)



01. while문에 의한 문장의 반복

❖ 구구단의 출력

```
1  /* multiplication_table1.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int dan = 0, num = 1;
8      printf("몇 단? ");
9      scanf("%d", &dan);
10     while (num < 10)
11     {
12         printf("%d x %d = %d\n", dan, num, dan * num);
13         num++;
14     }
15     return 0;
16 }
```

몇 단? 4

4 x 1 = 4

4 x 2 = 8

4 x 3 = 12

4 x 4 = 16

4 x 5 = 20

4 x 6 = 24

4 x 7 = 28

4 x 8 = 32

4 x 9 = 36

구구단은 반복문을 이해하는데 사용되는 대표적인 예제입니다.
이후에 반복문의 중첩에서는 구구단 전체를 출력하는 예제를 소개합니다.

01. while문에 의한 문장의 반복

❖ 무한 루프(Infinite Loop)의 구성

```
1  /* multiplication_table2.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int dan = 0, num = 1;
8      printf("몇 단? ");
9      scanf("%d", &dan);
10     while (1)
11     {
12         printf("%d x %d = %d\n", dan, num, dan * num);
13         num++;
14     }
15     return 0;
16 }
```

- ✓ 숫자 1은 True(참)를 의미하므로 반복문의 조건은 계속해서 True가 됩니다.
- ✓ 이렇듯 반복문의 탈출 조건이 성립하지 않는 경우, 무한 루프를 형성한다고 합니다.



- ✓ 이러한 무한 루프는 실수로 만들어지기도 하지만, break문과 함께 유용하게 사용되기도 합니다.

01. while문에 의한 문장의 반복


❖ while문의 중첩

```
1  /* multiplication_table3.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int dan = 2, num = 0;
7
8      while (dan < 10) // 2단부터 9단까지 반복
9      {
10         num = 1; // 새로운 단의 시작을 위해 초기화
11         while (num < 10)
12         {
13             printf("%d x %d = %d\n", dan, num, dan * num);
14             num++;
15         }
16         dan++; // 다음 단으로 넘어가기 위한 증가
17     }
18     return 0;
19 }
```

바깥쪽 while문

안쪽 while문

- ✓ while문 안에 while문이 존재하고 있습니다.
- ✓ 본 예제에서는 while문을 중첩시켜 구구단 전체를 출력합니다.
- ✓ 이 예제를 통해서 중첩된 while문의 코드 흐름을 이해하도록 합니다.



```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
...
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
```


02. do~while문에 의한 문장의 반복

- 01. while문에 의한 문장의 반복
- 03. for문에 의한 문장의 반복
- 04. 연습 문제

02. do~while문에 의한 문장의 반복

❖ do~while문의 기본 구성

```
1  /* hello_do_while.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int num = 0;
7
8      do
9      {
10         printf("Hello, World!\n");
11         num++;
12     } while (num < 3);
13
14     return 0;
15 }
```



```
Hello, World!
Hello, World!
Hello, World!
```

**반복 조건을 반복문의 마지막에 진행하는 형태이기 때문에
최소한 1회는 반복 영역을 실행하게 됩니다.**

02. do~while문에 의한 문장의 반복

❖ do~while문이 자연스러운 상황 (1/2)

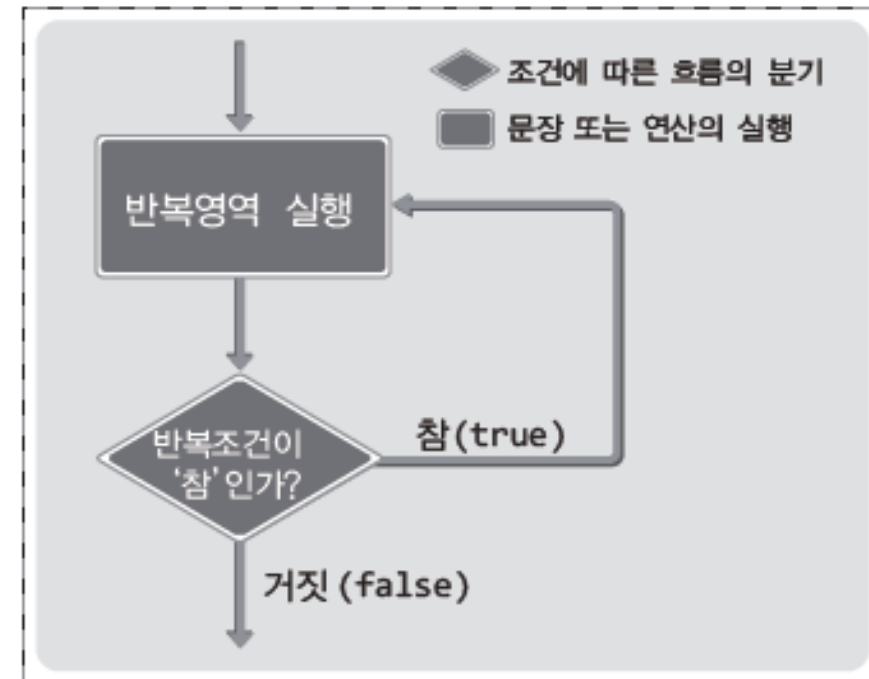
```
int num = 0;
while (num < 3)
{
    printf("Hello, World!\n");
    num++;
}
```



동일한 횟수를 반복하는 반복문들

```
int num = 0;
do
{
    printf("Hello, World!\n");
    num++;
} while (num < 3);
```

do~while문의 Flow Chart(순서도)



02. do~while문에 의한 문장의 반복

❖ do~while문이 자연스러운 상황 (2/2)

```
1  /* add_end.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int total = 0, num = 0;
8
9      do
10     {
11         printf("정수 입력 (0 to quit): ");
12         scanf("%d", &num);
13         total = total + num;
14     } while (num != 0);
15     printf("합계: %d\n", total);
16     return 0;
17 }
```

정수 입력 (0 to quit): 1
정수 입력 (0 to quit): 2
정수 입력 (0 to quit): 3
정수 입력 (0 to quit): 4
정수 입력 (0 to quit): 5
정수 입력 (0 to quit): 0
합계: 15

**사용자가 입력하는 수를 계속해서 더하는 프로그램입니다.
이러한 덧셈은 사용자가 0을 입력할 때까지 계속됩니다.**

**최소한 1회 이상 실행되어야 하는 반복문은
do~while문으로 구성하는 것이 자연스럽습니다.**

03. for문에 의한 문장의 반복

- 01. while문에 의한 문장의 반복
- 02. do~while문에 의한 문장의 반복
- 04. 연습 문제

03. for문에 의한 문장의 반복

❖ 반복문의 필수 3가지 필수요소

```
int main(void)
{
    int num = 0;    // 필수요소 1
    while (num < 3) // 필수요소 2
    {
        printf("Hi\n");
        num++;      // 필수요소 3
    }
    ...
}
```

필수요소 1 반복을 위한 변수의 선언

필수요소 2 반복의 조건 검사

필수요소 3 반복의 조건을 False(거짓)로 만들기 위한 연산

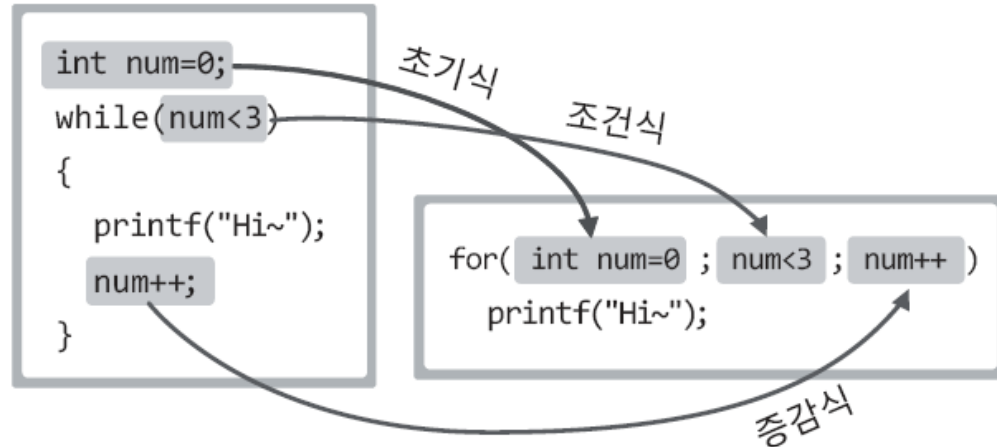
- ✓ 옆의 while문에서 보이듯이 반복문에 필요한 3가지 요소가 여러 행에 걸쳐서 분산되어 있습니다.
- ✓ 따라서 반복의 횟수가 바로 인식 불가능합니다.

- ✓ 정해진 횟수의 반복을 위해서는 하나의 변수가 필요합니다.
- ✓ 그 변수를 기반으로 하는 조건 검사가 필요합니다.
- ✓ 조건 검사가 False가 되게 하기 위한 연산이 필요합니다.

이 3가지를 한 줄에 표시하도록
돕는 것이 for문입니다.

03. for문에 의한 문장의 반복

❖ for문의 구조와 이해



```
for (초기식; 조건식; 증감식)
{
    // 반복의 대상이 되는 문장들
}
```

```
#include <stdio.h>

int main(void)
{
    int num;
    for (num = 0; num < 3; num++)
        printf("Hi\n");
    ...
}
```

- ✓ 일부 컴파일러는 여전히 초기식에서의 변수 선언을 허용하지 않는다.
- ✓ for문의 반복 영역도 한 줄이면 중괄호 { }의 생략이 가능합니다!

03. for문에 의한 문장의 반복

❖ for문의 흐름 이해

for문의 구성요소

- ① (초기식) 본격적으로 반복을 시작하기에 앞서 딱 한 번만 실행됩니다.
- ② (조건식) 매 반복의 시작에 앞서 실행되며, 그 결과를 기반으로 반복 유무를 결정합니다.
- ③ (증감식) 매 반복 실행 후 마지막에 연산이 이뤄집니다.

```
for (초기식; 조건식; 증감식)
{
    // 반복의 대상이 되는 문장들
}
```

➊ 첫 번째 반복의 흐름
1 → 2 → 3 → 4 [num=1]

➋ 두 번째 반복의 흐름
2 → 3 → 4 [num=2]

➌ 세 번째 반복의 흐름
2 → 3 → 4 [num=3]

➍ 네 번째 반복의 흐름
2 [num=3] 따라서 탈출!

```
1      2      4
for( int num=0 ; num<3 ; num++ )
{ 3
    printf("Hi~");
}
```


for문 흐름의 핵심

- ✓ int num = 0에 해당하는 초기화는 반복문의 시작에 앞서 딱 1회 진행!
- ✓ num < 3에 해당하는 조건의 검사는 매 반복문의 시작에 앞서 진행!
- ✓ num++에 해당하는 증감 연산은 반복 영역을 실행한 후에 진행!

03. for문에 의한 문장의 반복

❖ for문 기반의 다양한 예제 (1/2)

```
1  /* add_number.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int total = 0;
8      int j, num;
9      printf("0부터 num까지의 덧셈, num은? ");
10     scanf("%d", &num);
11
12     for (j = 0; j <= num; j++)
13         total = total + j;
14
15     printf("0부터 %d까지의 덧셈 결과: %d\n", num, total);
16     return 0;
17 }
```




0부터 num까지의 덧셈, num은? 10
0부터 10까지의 덧셈 결과: 55

03. for문에 의한 문장의 반복

❖ for문 기반의 다양한 예제 (2/2)

```
1  /* mean.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      double total = 0.0;
8      double input = 0.0;
9      int num = 0;
10
11     for (; input >= 0.0; )
12     {
13         total = total + input;
14         printf("실수 입력 (minus to quit): ");
15         scanf("%lf", &input);
16         num++;
17     }
18     printf("평균: %f\n", total/(num - 1));
19     return 0;
20 }
```




실수 입력 (minus to quit): 3.2323
실수 입력 (minus to quit): 5.1891
실수 입력 (minus to quit): 2.9297
실수 입력 (minus to quit): -1.0
평균: 3.783700

03. for문에 의한 문장의 반복

❖ for문의 중첩

```
1  /* for_overlap.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int dan, num;
7
8      for (dan = 2; dan < 10; dan++)
9      {
10         for (num = 1; num < 10; num++)
11         {
12             printf("%d x %d = %d\n", dan, num, dan * num);
13         }
14         printf("\n");
15     }
16     return 0;
17 }
```



```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
...
9 x 6 = 54
9 x 7 = 63
9 x 8 = 72
9 x 9 = 81
```

- ✓ for문의 중첩은 while, do~while문의 중첩과 다르지 않습니다.
- ✓ 구구단 전체를 출력하는 왼편의 예제를 통해서 for문의 중첩을 이해하도록 합니다.

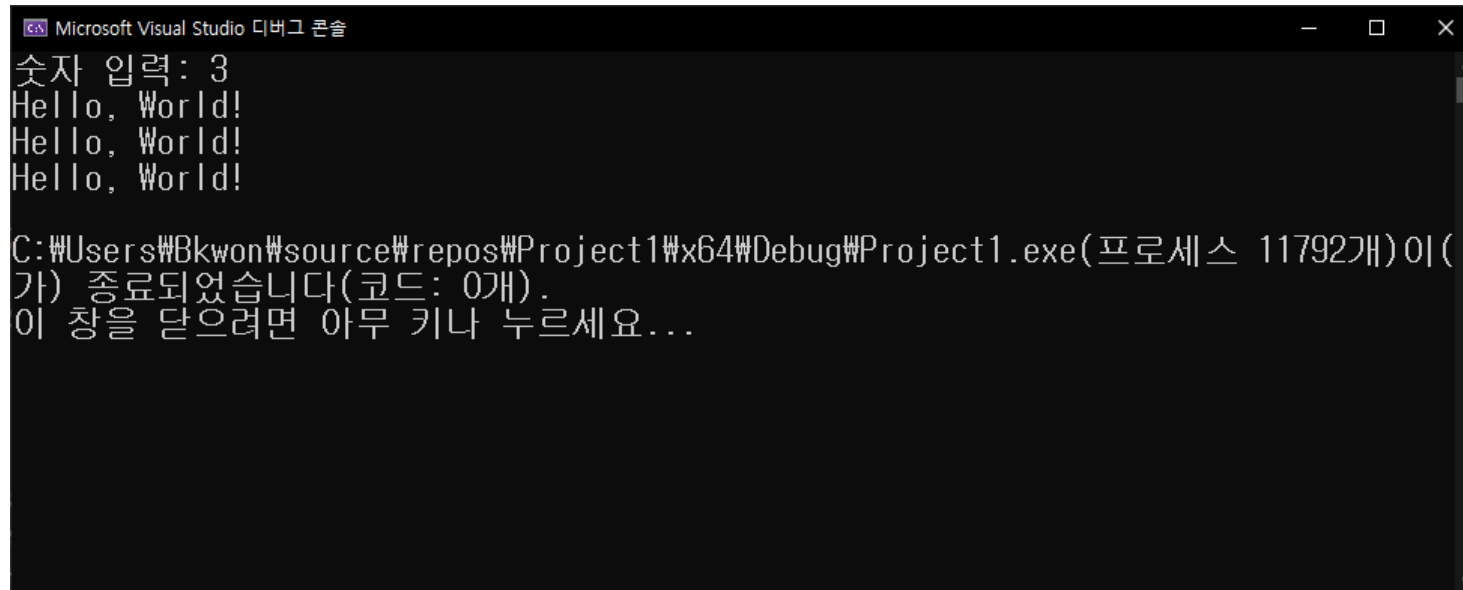
04. 연습 문제

- 01. while문에 의한 문장의 반복
- 02. do~while문에 의한 문장의 반복
- 03. for문에 의한 문장의 반복

04. 연습 문제

❖ (while문) 연습 문제 1.

- 사용자로부터 숫자를 하나 입력받아서, 그 수만큼 "Hello, World!"를 출력하는 프로그램을 작성해 보세요.



```
Microsoft Visual Studio 디버그 콘솔
숫자 입력: 3
Hello, World!
Hello, World!
Hello, World!

C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 11792개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

04. 연습 문제

❖ (while문) 연습 문제 1. 정답 및 해설

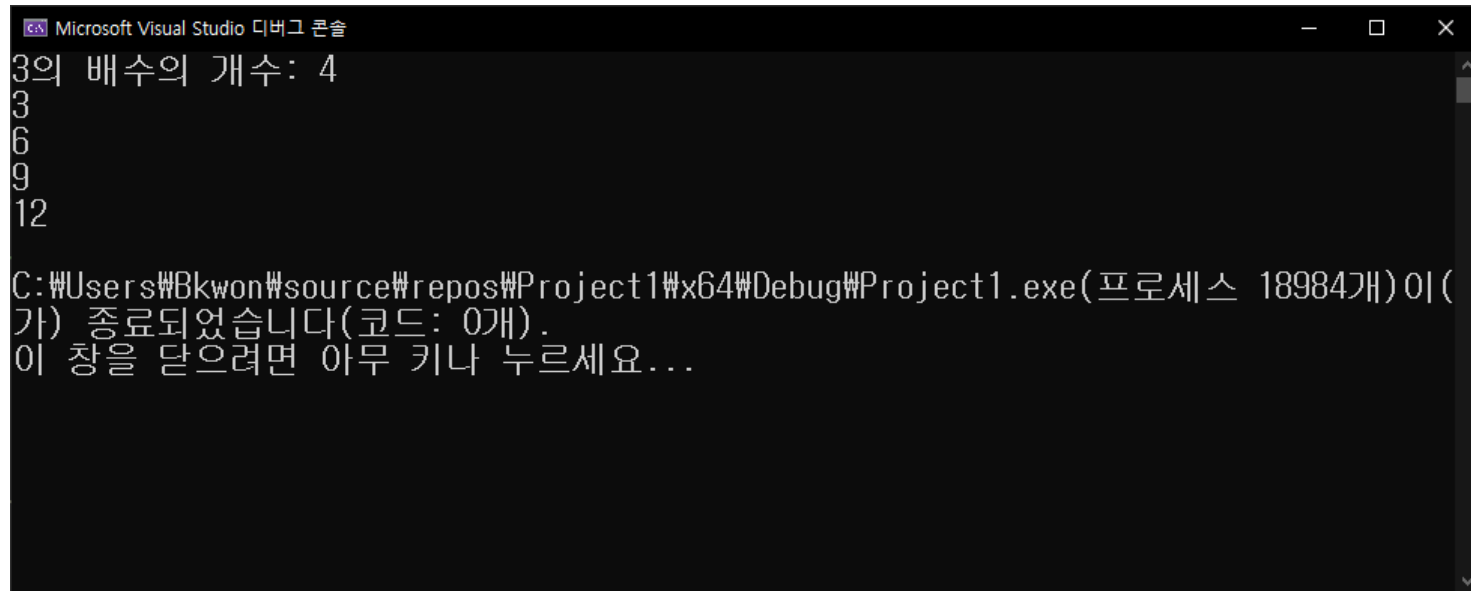
```
1  /* example1.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int num;
8      int j = 0;
9
10     printf("숫자 입력: ");
11     scanf("%d", &num);
12
13     while (j < num)
14     {
15         printf("Hello, World!\n");
16         j = j + 1;
17     }
18     return 0;
19 }
```

숫자 입력: 3
Hello, World!
Hello, World!
Hello, World!

04. 연습 문제

❖ (while문) 연습 문제 2.

- 사용자로부터 하나의 숫자를 입력받은 다음, 그 수만큼 3의 배수를 출력하는 프로그램을 작성하세요.
만약에 사용자로부터 5를 입력받았다면, 3 6 9 12 15를 출력해야 합니다.



```
Microsoft Visual Studio 디버그 콘솔
3의 배수의 개수: 4
3
6
9
12

C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 18984개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

04. 연습 문제

❖ (while문) 연습 문제 2. 정답 및 해설

```
1  /* example2.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int num;
8      int j = 1;
9
10     printf("3의 배수의 개수: ");
11     scanf("%d", &num);
12
13     while (j < num + 1)
14     {
15         printf("%d\n", j * 3);
16         j = j + 1;
17     }
18     return 0;
19 }
```

3의 배수의 개수: 4

3

6

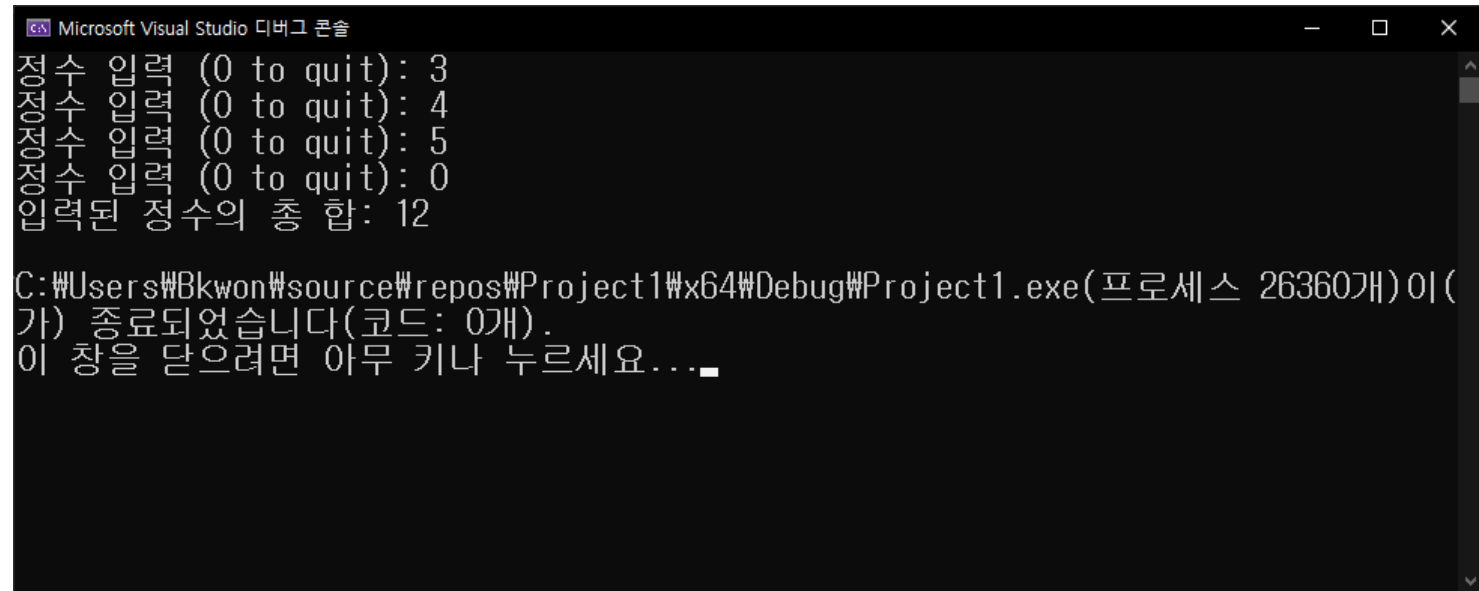
9

12

04. 연습 문제

❖ (while문) 연습 문제 3.

- 사용자가 입력하는 정수를 계속해서 더해 나가는 프로그램을 작성해 보세요.
만약에 0이 입력되면 지금까지 입력된 정수의 덧셈 결과를 출력하고 프로그램을 종료시킵니다.



```
Microsoft Visual Studio 디버그 콘솔
정수 입력 (0 to quit): 3
정수 입력 (0 to quit): 4
정수 입력 (0 to quit): 5
정수 입력 (0 to quit): 0
입력된 정수의 총 합: 12

C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 26360개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...■
```

04. 연습 문제

❖ (while문) 연습 문제 3. 정답 및 해설

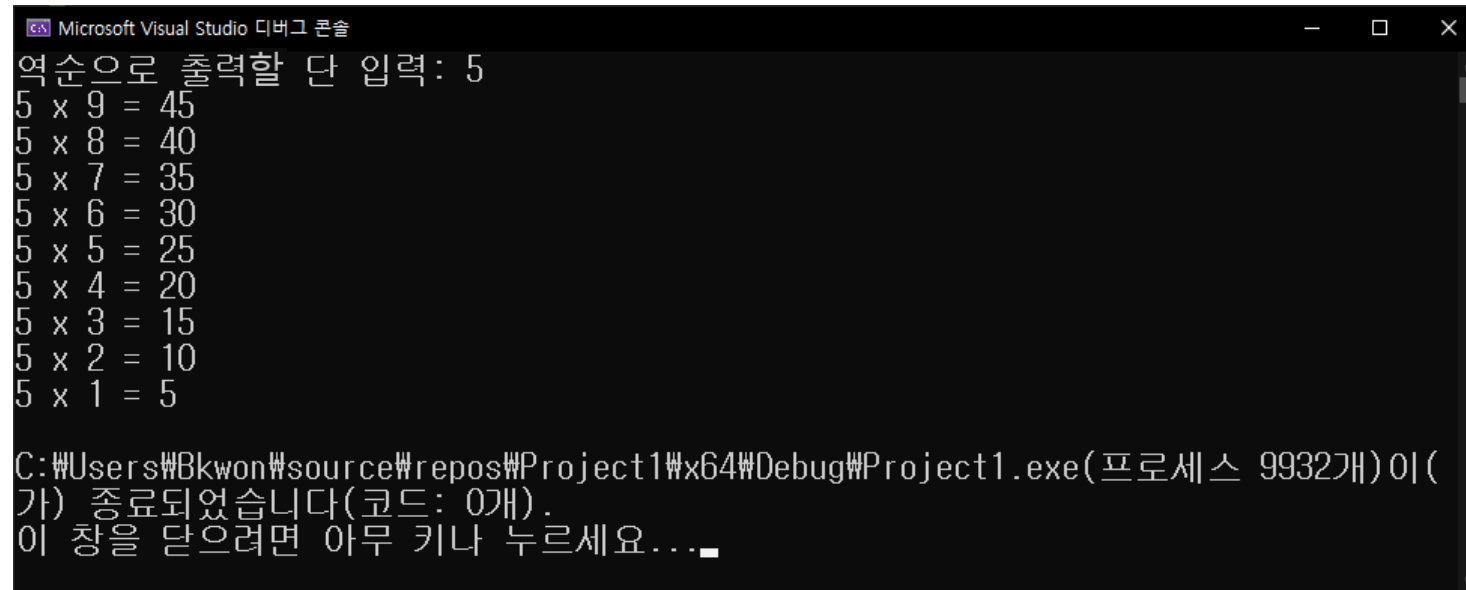
```
1  /* example3.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int num = 1;
8      int result = 0;
9
10     while (num != 0)
11     {
12         printf("정수 입력 (0 to quit): ");
13         scanf("%d", &num);
14         result = result + num;
15     }
16     printf("입력된 정수의 총 합: %d\n", result);
17     return 0;
18 }
```

정수 입력 (0 to quit): 3
정수 입력 (0 to quit): 4
정수 입력 (0 to quit): 5
정수 입력 (0 to quit): 0
입력된 정수의 총 합: 12

04. 연습 문제

❖ (while문) 연습 문제 4.

- 사용자로부터 입력받은 숫자에 해당하는 구구단을 출력하되, 역순으로 출력하는 프로그램을 작성해 보세요.



```
Microsoft Visual Studio 디버그 콘솔
역순으로 출력할 단 입력: 5
5 x 9 = 45
5 x 8 = 40
5 x 7 = 35
5 x 6 = 30
5 x 5 = 25
5 x 4 = 20
5 x 3 = 15
5 x 2 = 10
5 x 1 = 5

C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 9932개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...■
```

04. 연습 문제

❖ (while문) 연습 문제 4. 정답 및 해설

```
1  /* example4.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int dan;
8      int j = 9;
9
10     printf("역순으로 출력할 단 입력: ");
11     scanf("%d", &dan);
12
13     while (0 < j)
14     {
15         printf("%d x %d = %d\n", dan, j, dan * j);
16         j = j - 1;
17     }
18     return 0;
19 }
```

역순으로 출력할 단 입력: 3

3 x 9 = 27

3 x 8 = 24

3 x 7 = 21

3 x 6 = 18

3 x 5 = 15

3 x 4 = 12

3 x 3 = 9

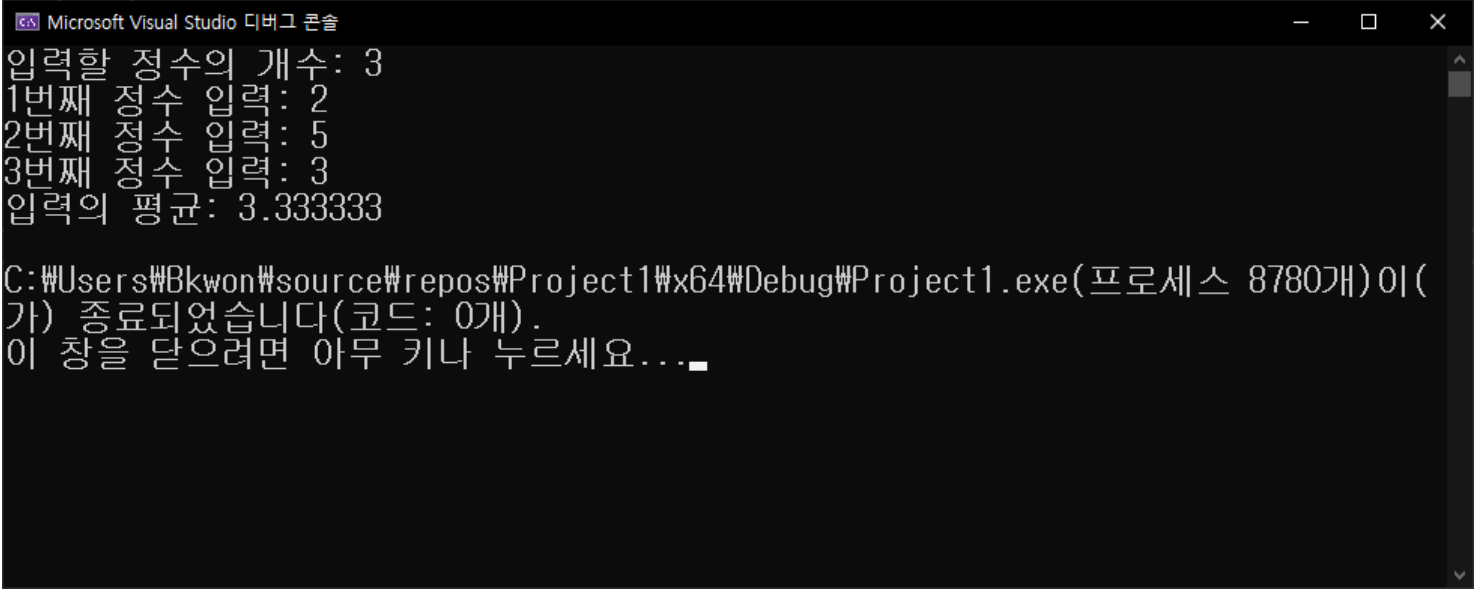
3 x 2 = 6

3 x 1 = 3

04. 연습 문제

❖ (while문) 연습 문제 5.

- 입력된 정수의 평균을 구하는 프로그램을 작성해 보세요. 제일 먼저, 입력할 정수의 개수를 사용자로부터 입력받습니다. 그리고 그 수만큼 정수를 입력받아서 평균값을 출력해 줍니다. 입력받은 값은 정수이지만, 평균값은 실수가 될 것입니다.



```
Microsoft Visual Studio 디버그 콘솔
입력할 정수의 개수: 3
1번째 정수 입력: 2
2번째 정수 입력: 5
3번째 정수 입력: 3
입력의 평균: 3.333333
C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 8780개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...■
```

04. 연습 문제

❖ (while문) 연습 문제 5. 정답 및 해설

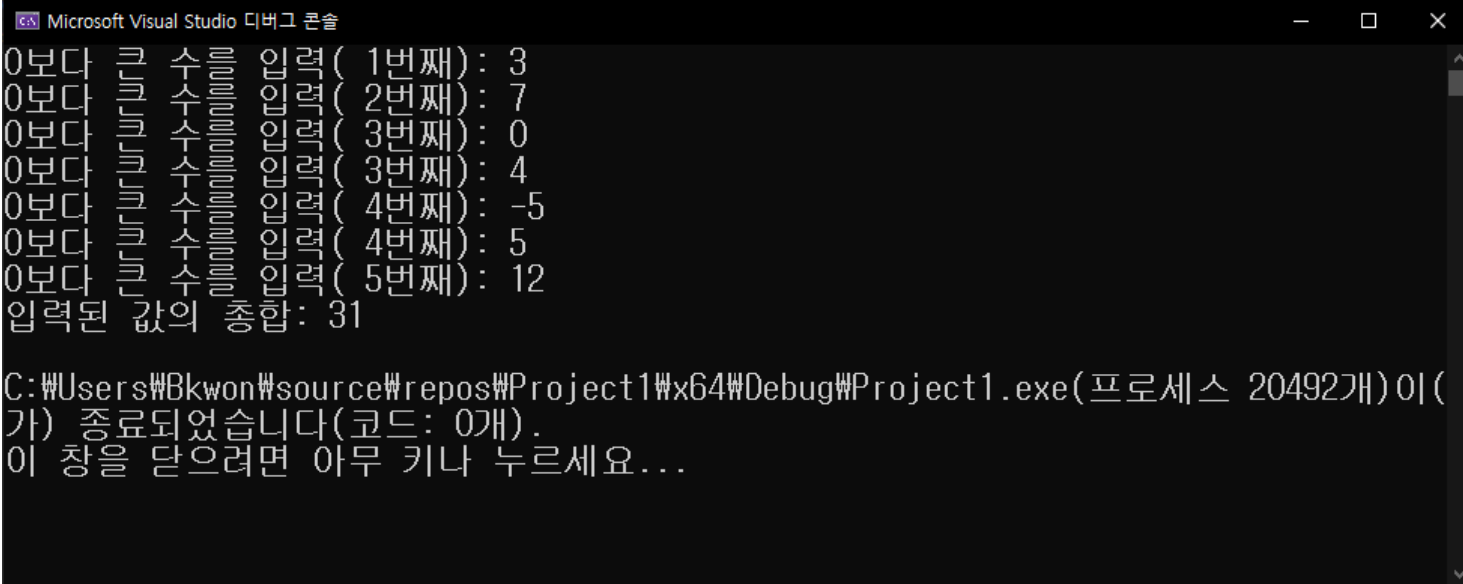
```
1  /* example5.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4  int main(void)
5  {
6      int total = 0, j = 0;
7      int num, input;
8
9      printf("입력할 정수의 개수: ");
10     scanf("%d", &num);
11     while (j < num)
12     {
13         printf("%d번째 정수 입력: ", j + 1);
14         scanf("%d", &input);
15         total = total + input;
16         j = j + 1;
17     }
18     printf("입력의 평균: %f\n", (double)total/num);
19     return 0;
20 }
```

입력할 정수의 개수: 3
1번째 정수 입력: 2
2번째 정수 입력: 5
3번째 정수 입력: 3
입력의 평균: 3.333333

04. 연습 문제

❖ (while문의 중첩) 연습 문제 6.

- 사용자로부터 총 5개의 정수를 입력받아서 그 수의 합을 출력해 주는 프로그램을 구현해 보세요.
단, 한 가지 조건이 있습니다. 정수는 반드시 0보다 큰 수(0 초과)이어야 합니다.
만약에 0이하의 수를 입력받을 경우에는 입력으로 인정하지 않고 다시 입력받도록 구현해야 합니다.



```
Microsoft Visual Studio 디버그 콘솔
0보다 큰 수를 입력( 1번째): 3
0보다 큰 수를 입력( 2번째): 7
0보다 큰 수를 입력( 3번째): 0
0보다 큰 수를 입력( 3번째): 4
0보다 큰 수를 입력( 4번째): -5
0보다 큰 수를 입력( 4번째): 5
0보다 큰 수를 입력( 5번째): 12
입력된 값의 총합: 31

C:\Users\Bkwon\source\repos\Project1\Project1.exe(프로세스 20492개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...
```

04. 연습 문제

❖ (while문의 중첩) 연습 문제 6. 정답 및 해설

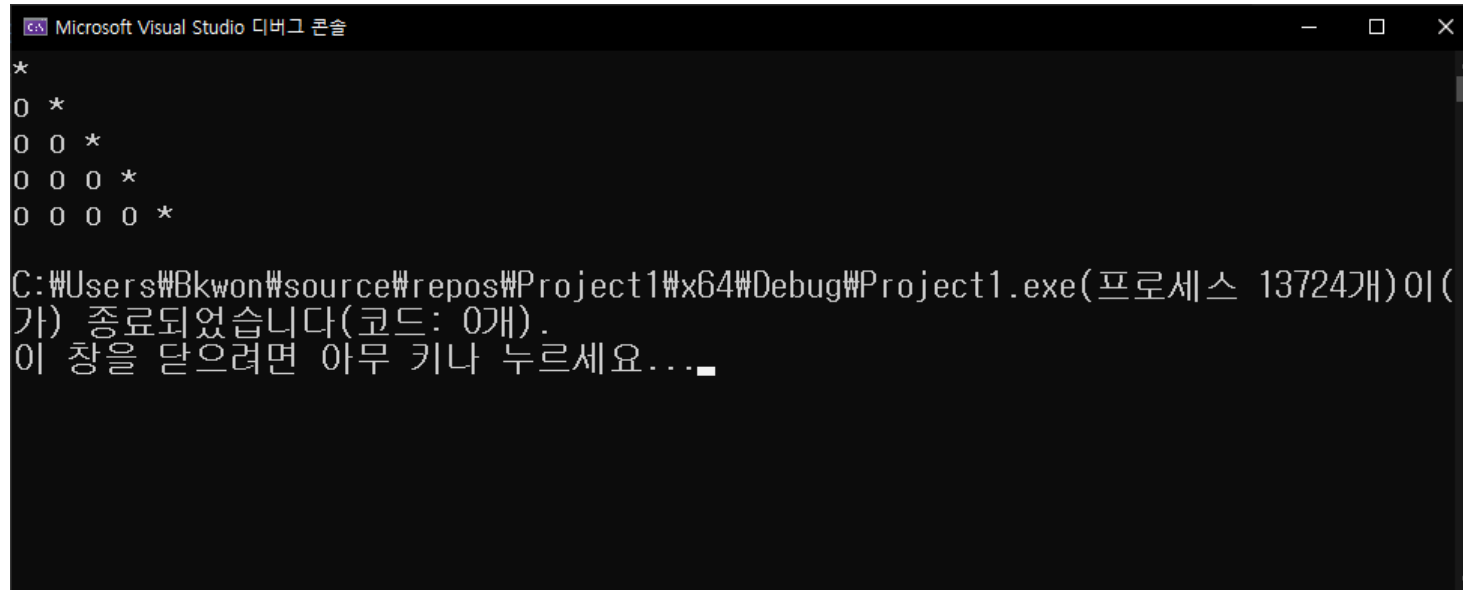
```
1  /* example6.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4  int main(void)
5  {
6      int total = 0, j = 1, input = 0;
7      while (j <= 5)
8      {
9          while (input <= 0)
10         {
11             printf("0보다 큰 수를 입력(%d번째): ", j);
12             scanf("%d", &input);
13         }
14         total = total + input;
15         j = j + 1;
16         input = 0;
17     }
18     printf("입력된 값의 총합: %d\n", total);
19     return 0;
20 }
```

0보다 큰 수를 입력(1번째): 3
0보다 큰 수를 입력(2번째): 7
0보다 큰 수를 입력(3번째): 0
0보다 큰 수를 입력(3번째): 4
0보다 큰 수를 입력(4번째): -5
0보다 큰 수를 입력(4번째): 5
0보다 큰 수를 입력(5번째): 12
입력된 값의 총합: 31

04. 연습 문제

❖ (while문의 중첩) 연습 문제 7.

- 다음과 같은 출력을 보일 수 있도록 프로그램을 작성하되 반드시 while문을 중첩시켜서 구현해 보세요.



```
Microsoft Visual Studio 디버그 콘솔


*
0 *
0 0 *
0 0 0 *
0 0 0 0 *

C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 13724개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...■
```

04. 연습 문제

❖ (while문의 중첩) 연습 문제 7. 정답 및 해설

```
1  /* example7.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int j = 0, k = 0;
7
8      while (j < 5)
9      {
10         while (k < j)
11         {
12             printf("o ");
13             k = k + 1;
14         }
15         k = 0;
16         j = j + 1;
17         printf("*\n");
18     }
19     return 0;
20 }
```

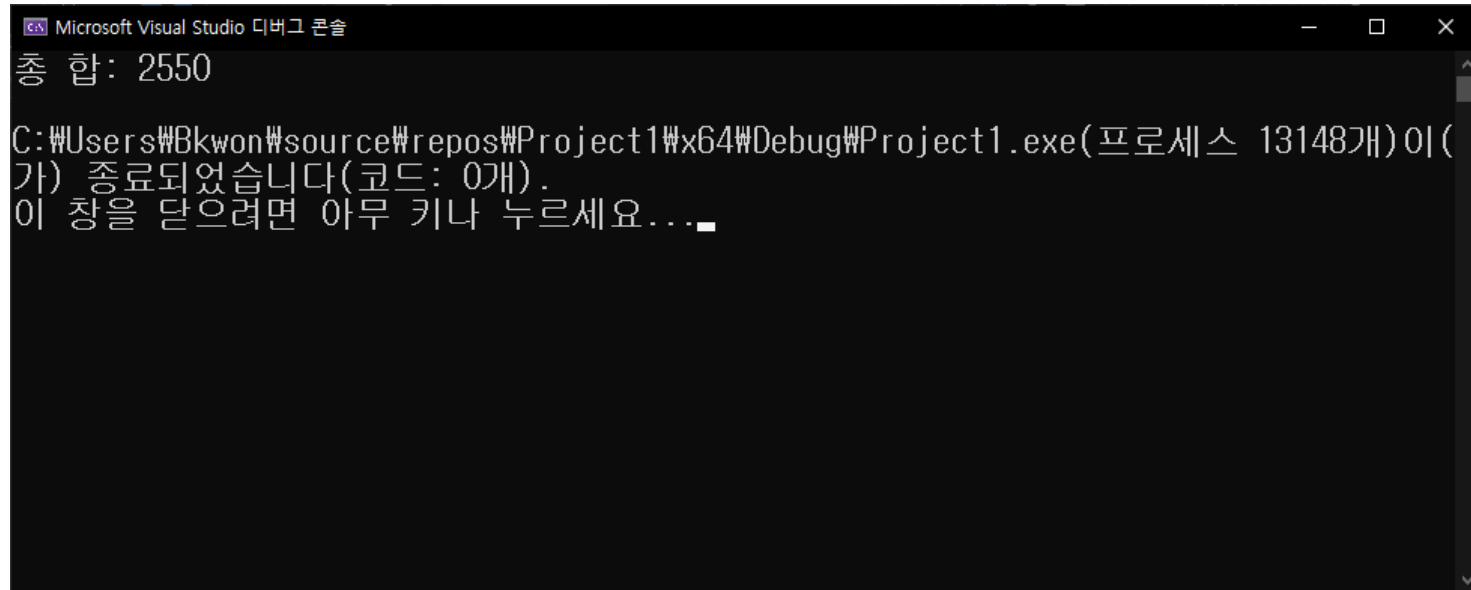


```
*
o *
o o *
o o o *
o o o o *
```

04. 연습 문제

❖ (do~while문) 연습 문제 8.

- 0과 100사이에 존재하는 짝수의 합을 계산해서 출력하는 프로그램을 do~while문을 이용해서 작성해 보세요. 참고로 출력 결과는 2550이 되어야 합니다.



```
Microsoft Visual Studio 디버그 콘솔
총 합: 2550
C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 13148개)이(가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요...■
```

04. 연습 문제

❖ (do~while문) 연습 문제 8. 정답 및 해설

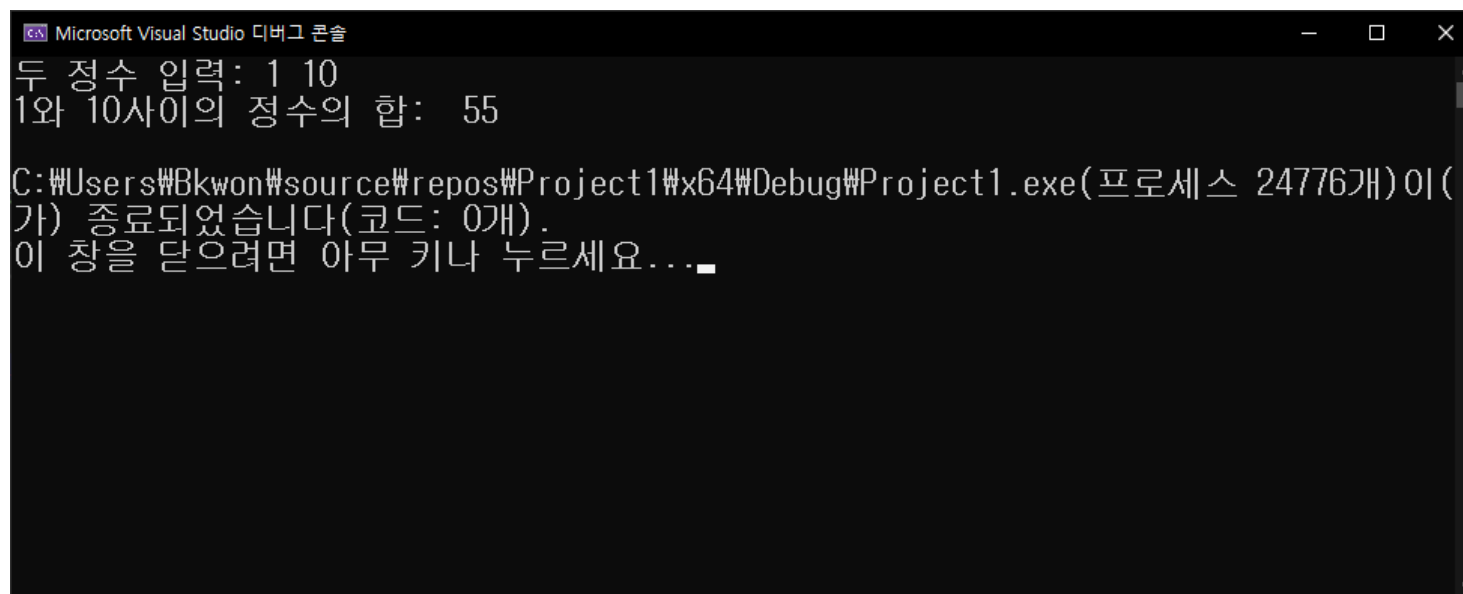
```
1  /* example8.c */
2  #include <stdio.h>
3
4  int main(void)
5  {
6      int num = 0, sum = 0;
7
8      do
9      {
10         sum = sum + num;
11         num = num + 2;
12     } while (num <= 100);
13
14     printf("총 합: %d\n", sum);
15     return 0;
16 }
```

총 합: 2550

04. 연습 문제

❖ (for문) 연습 문제 9.

- 두 개의 정수를 입력받아서 그 사이에 존재하는 정수들의 합을 구하는 프로그램을 작성해 보세요.
예를 들어, 3과 5를 입력받는다면 $3+4+5$ 가 답이 됩니다.
문제를 조금 쉽게 하기 위해서, 첫 번째로 입력받은 숫자보다
두 번째로 입력받은 숫자가 더 크다는 조건을 걸도록 하겠습니다.



```
Microsoft Visual Studio 디버그 콘솔
두 정수 입력: 1 10
1와 10사이의 정수의 합: 55
C:\Users\Bkwon\source\repos\Project1\x64\Debug\Project1.exe(프로세스 24776개)이(
가) 종료되었습니다(코드: 0개).
이 창을 닫으려면 아무 키나 누르세요..._
```

04. 연습 문제

❖ (for문) 연습 문제 9. 정답 및 해설

```
1  /* example9.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int num1, num2;
8      int sum = 0, j;
9
10     printf("두 정수 입력: ");
11     scanf("%d %d", &num1, &num2);
12     for (j = num1; j <= num2; j++)
13     {
14         sum = sum + j;
15     }
16     printf("%d와 %d사이의 정수의 합: %d\n", num1, num2, sum);
17     return 0;
18 }
```

두 정수 입력: 1 10
1과 10사이의 정수의 합: 55

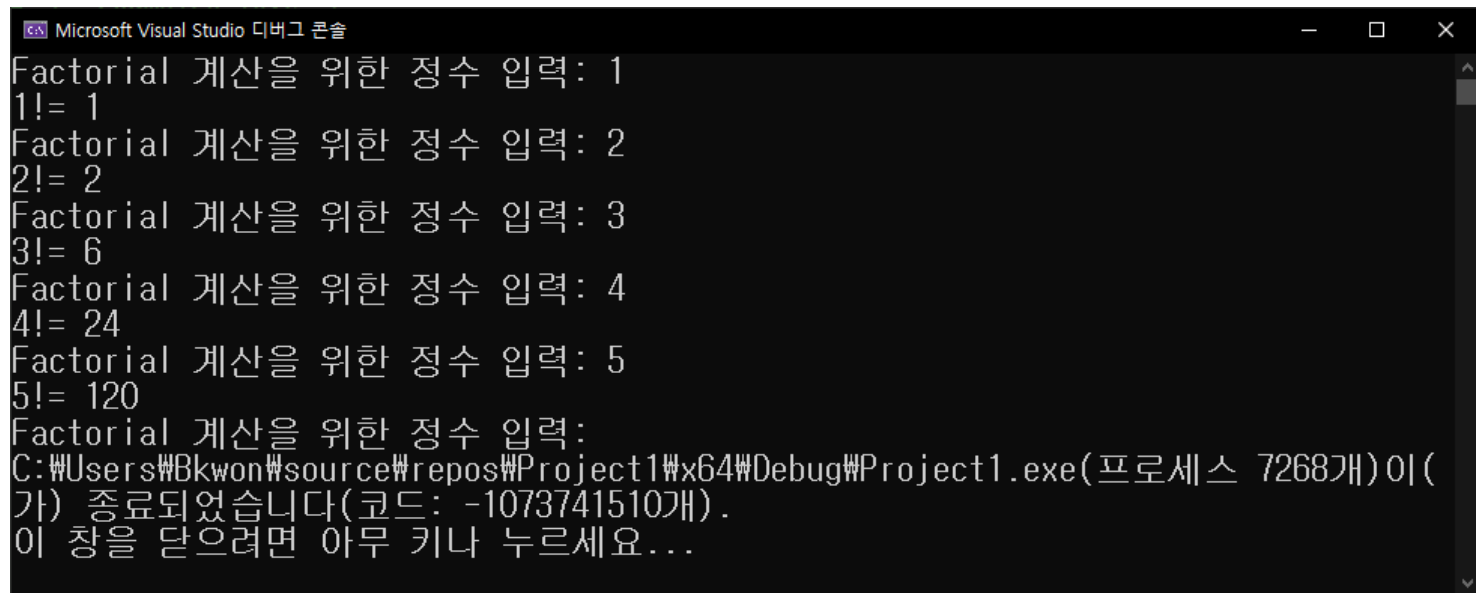
04. 연습 문제

❖ (for문) 연습 문제 10.

- 계승(Factorial)을 계산하는 프로그램을 작성해 보세요. 사용자로부터 n 이라는 수를 입력받습니다. 그러면 n 의 계승 $n!$ 을 계산해서 출력해 줍니다. 공식은 다음과 같습니다.

$$n! = 1 \times 2 \times 3 \times \cdots n$$

단, 이번 문제는 무한 루프로 구성합니다. 즉 사용자에게는 서비스가 계속될 것입니다.
언제까지? Ctrl + C 키를 누를 때 까지



```
Microsoft Visual Studio 디버그 콘솔
Factorial 계산을 위한 정수 입력: 1
1!= 1
Factorial 계산을 위한 정수 입력: 2
2!= 2
Factorial 계산을 위한 정수 입력: 3
3!= 6
Factorial 계산을 위한 정수 입력: 4
4!= 24
Factorial 계산을 위한 정수 입력: 5
5!= 120
Factorial 계산을 위한 정수 입력:
C:\Users\Bkwon\source\repos\Project1\Debug\Project1.exe(프로세스 7268개)이(
가) 종료되었습니다(코드: -1073741510개).
이 창을 닫으려면 아무 키나 누르세요...
```

04. 연습 문제

❖ (for문) 연습 문제 10. 정답 및 해설

```
1  /* example10.c */
2  #define _CRT_SECURE_NO_WARNINGS
3  #include <stdio.h>
4
5  int main(void)
6  {
7      int n, j, result;
8      for (; ;)
9      {
10         result = 1;
11         printf("Factorial 계산을 위한 정수 입력: ");
12         scanf("%d", &n);
13         for (j = 1; j <= n; j++)
14         {
15             result = result * j;
16         }
17         printf("%d! = %d\n", n, result);
18     }
19     return 0;
20 }
```

Factorial 계산을 위한 정수 입력: 1
1! = 1
Factorial 계산을 위한 정수 입력: 2
2! = 2
Factorial 계산을 위한 정수 입력: 3
3! = 6
Factorial 계산을 위한 정수 입력: 4
4! = 24
...

- ❖ 01. while문에 의한 문장의 반복
- ❖ 02. do~while문에 의한 문장의 반복
- ❖ 03. for문에 의한 문장의 반복
- ❖ 04. 연습 문제

THANK YOU!

Q & A

- Name: 권범
- Office: 동덕여자대학교 인문관 B821호
- Phone: 02-940-4752
- E-mail: bkwon@dongduk.ac.kr