실 습 문 제 제7장

1. **[반복문의 개념]**

```
#include <stdio.h>

int main(void)
{
    int i = 0; // ①

    while(i < 5)
    {
        printf("Hello World!\n");
        i++;       // ②
    }
}
```

실행결과(기록하기)

(a) 위의 프로그램을 컴파일하고 실행하여 보라. 어떤 출력이 생성되는가?

(b) ②번 문장 다음에 변수 i의 값을 출력하는 문장을 추가하라. i의 값은 어떻게 변화되는가?

	반복 #1	반복 #2	반복 #3	반복 #4	반복 #5
변수 i의 값					

(c) ①번 문장에서 만약 변수 i를 0으로 초기화하지 않으면 어떻게 되는지 살펴보라.

(d) ②번 문장이 생략되면 실행 결과가 어떻게 되는가? 이러한 상황을 무엇이라고 부르는가? Ctrl+C 키를 눌러서 프로그램을 종료하라.

(e) 사용자로부터 정수를 입력받아서 그 수만큼 "Hello World!"를 출력하도록 위의 프로그램을 수정하여 보라.

2. **[반복문의 종류]**

반복에는 2가지의 종류가 있다. 첫 번째는 미리 반복하는 횟수를 정해놓고 반복을 실행하는 것이고 두 번째는 사용자가 특정한 값을 입력할 때까지 반복을 계속하는 방법이다. 이번 문제에서는 사용자에게서 성적을 입력받아서 성적의 평균을 구해보자.

```c
#include <stdio.h>

int main(void)
{
    int i = 0, sum = 0;
    int score;

    while(i < 3)
    {
        scanf("%d", &score);
        sum += score;
        i++;
    }
    printf("평균= %d\n", sum / i);
}
```

실행결과(기록하기)

(a) 위의 프로그램을 컴파일하고 실행하여 보라. 어떤 출력이 생성되는가?

(b) 위의 프로그램을 for 문을 이용하여 다시 작성하여 보라.

(c) 위의 프로그램은 반복 횟수가 미리 결정되어 있는 경우에 사용할 수 있다. 이것을 사용자가 특별한 값을 입력하면 반복을 중단하도록 수정하여 보라. 여기서는 사용자가 음수의 값을 입력하면 반복을 중단하도록 하여 보라. break 문을 사용하라.

(d) 성적이 100점을 넘을 수는 없다고 가정하자. 만약 사용자가 100을 넘는 성적을 입력하면 입력으로 인정하지 않고 다시 입력하도록 하여 보라. 아래는 실행 예이다. 중첩 반복문을 사용하여 구현하여 보라.

```
101
성적은 100을 넘을 수 없습니다. 다시 입력하시오.
102
성적은 100을 넘을 수 없습니다. 다시 입력하시오.
100
90
80
-1
평균 = 90
```

(e) 평균이 실수값으로 계산되도록 프로그램을 수정하라.

3. [do...while 문]

do...while 문은 대개 조건에 관계없이 적어도 한번은 실행되어야 하는 상황에 주로 사용된다. 특히, 사용자에게 적어도 한번은 입력을 받아야 하는 경우에 사용된다.

이런 사례로 사용자로부터 입력되는 비밀번호를 검사하는 프로그램을 작성하여 보자. 사용자에게서 비밀번호를 받아서 검사하여서 일치하면 "HELLO"라는 메시지를 출력한다. 만약 일치하지 않았으면 되풀이하여 비밀번호를 입력하도록 한다. 이 문제에서는 일단은 사용자로부터 하나의 정수는 받아야 검사를 할 수 있다. 따라서 이런 경우에는 while 문보다는 do..while 문이 더 적합하다고 할 수 있다.

```
#include <stdio.h>

int main(void)
{
    const int password = 1234;
    int code;

    do
    {
        printf("비밀번호를 입력하시오:");
        scanf("%d", &code);
    } while(code != password);

    printf("성공적으로 로그인하였습니다.\n");
    return 0;
}
```

실행결과(기록하기)

(a) 위의 프로그램을 컴파일하고 실행하여 보라. 어떤 출력이 생성되는가?

(b) 위의 프로그램을 while 문을 이용하여 다시 작성하여 보라. 원래의 프로그램과 비교하여 보라.

(c) 사용자는 3번의 시도 안에 일치하는 비밀번호를 입력하도록 수정하여 보라. 만약 사용자가 3번 시도에도 불구하고 비밀번호를 정확히 입력하지 못했다면 프로그램을 종료하도록 수정하라.

4. [for 문]

실수의 거듭 제곱 값을 계산하는 프로그램을 작성하여 보자. 사용자로부터 하나의 실수 r과 거듭 제곱 횟수를 나타내는 정수 n을 입력받아서 r^n을 구하여 화면에 출력한다.

```
#include <stdio.h>

int main(void)
{
    float prod = 1.0, r;   // ①
```

```
    int i, n;

    printf("실수와 정수를 입력하시오:");
    scanf("%f %d", &r, &n);

    for(i = 0; i < n; i++)
    {
        prod *= r;          // ②
    }
    printf("%f\n", prod);
}
```

실행결과(기록하기)

```

```

(a) 위의 프로그램을 컴파일하고 실행하여 보라. 어떤 출력이 생성되는가?

(b) ①번 문장에서 만약 prod를 0.0으로 초기화하면 어떻게 되는가? 1.0으로 초기화하는 이유는 무엇인가?

(c) 한 번의 거듭제곱의 계산이 끝나게 되면 사용자에게 계속할 것인지를 물어보고 만약 'Y'가 입력되면 다시 새로운 실수와 정수를 입력받아서 거듭제곱 계산을 반복하도록 프로그램을 수정하라. 사용자가 'N'이라고 입력하면 프로그램을 종료한다. 사용자가 'Y'나 'N'를 제외한 다른 문자를 입력하면 입력을 무시하고 다시 입력하도록 한다. continue와 break, goto 등을 사용해보자.

(d) 1.0부터 10.0까지 0.5간격으로 세제곱표를 출력하도록 위의 프로그램을 수정하라.

5. **[응용]**

반복문을 이용하여 피보나치 수열을 구하는 프로그램을 작성하여 보자. 피보나치 수열은 0, 1, 1, 2, 3, 5, 8, 13, …과 같이 앞의 두 항을 더한 값이 다음 항이 되는 수열이다.

```
#include <stdio.h>

int main(void)
{
    long a = 0, b = 1, c;
    int i;

    for(i = 2; i <= 10; i++)
    {
        _________ ;    // 변수 a와 b를 더하여 변수 c에 대입한다.
        _________ ;    // 변수 b의 값을 변수 a로 옮긴다.
        _________ ;    // 변수 c의 값을 변수 b로 옮긴다.
```

```
    }
    return 0;
}
```

실행결과(기록하기)

(a) 빈 칸을 채우고 컴파일하고 실행하여 보라.

(b) 위의 프로그램에서 피보나치 수열의 각 항을 출력하도록 출력문을 추가하라.

(c) 사용자에게 몇 번째 항까지 출력할 것인지를 물어보고 그 항까지만 출력하도록 위의 프로그램을 변경하라.