실 습 문 제 제9장

1. **[함수의 범위] 전역 변수와 지역 변수에 대하여 살펴보자.**

```
#include <stdio.h>
int a;      // 전역 변수

int main(void)
{
    int b;
            // ①
    {
        int c;
            // ②
    }
    {
        int d;
            // ③
    }
    return 0;
}

void f(void)
{
    int e;
            // ④
}
```

(a) 각각의 변수가 전역 변수인지 지역 변수인지를 말하라.

(b) 각각의 변수의 범위를 표시하라.

(c) ①, ②, ③, ④ 위치에서 사용이 가능한 변수들을 조사하여 보자. 구체적으로 ①, ②, ③, ④ 위치에서 변수들의 값을 출력하는 문장을 삽입하여 어떤 변수들이 접근이 가능한지를 확인하라.

(d) 전역 변수와 지역 변수의 초기값은 무엇인지 살펴보라.

(e) 변수 a, b, c, d, e를 각각 1, 2, 3, 4, 5로 초기화한 후에 모든 변수들의 이름을 x로 통일한다. ①, ②, ③, ④ 위치에서 x의 값을 출력하여 보라. 전역 변수와 지역 변수가 이름이 같은 경우, 어떤 변수가 사용되는가?

2. **[전역 변수의 사용] 간단한 연산의 결과를 전역 변수를 이용하여 전달하여 보자.**

```
#include <stdio.h>

int add(int x, int y);     // 함수 원형 정의
```

```c
int main(void)
{
    int result;

    result = add(10, 20);
    printf("%d\n", result);

    return 0;
}

int add(int x, int y)
{
    // ①
    return x + y;
}
```

(a) 각각의 변수가 전역 변수인지 지역 변수인지를 말하라.

(b) 함수의 매개 변수에 다른 값을 대입할 수 있는가? ①번 위치에서 매개 변수 x와 y에 각각 30과 40을 대입하여 보라. 매개 변수는 지역 변수인가?

(c) 전역 변수 sum을 정의하고 이 전역 변수 sum을 이용하여 덧셈의 결과를 add()에서 main()으로 전달하여 보라. 전역 변수를 사용하는 방법의 장점과 단점은 무엇일까?

3. [static 변수의 사용] 정적 지역 변수는 함수가 종료되어도 값을 유지한다.

```c
#include <stdio.h>
void f(void);      // 함수 원형 정의

int main(void)
{
    f();
    f();
    f();
    return 0;
}

void f(void)
{
    static int x = 0;
    int y = 0;

    x++;
    y++;
    printf("x = %d, y = %d\n", x, y);
}
```

실행결과(기록하기)

(a) 위의 프로그램을 실행하고 그 결과를 기록하라. 결과를 설명하여 보라. 지역 변수 앞에 static을 붙이면 어떻게 되는가?

(b) main()에서 for 반복 구조를 이용하여 f()를 20번 호출하도록 코드를 수정하라.

(c) static 지역 변수를 이용하여 어떤 함수가 몇 번이나 호출되었는지 쉽게 알 수 있다. f()가 10번 이상 호출되면 x와 y의 값을 출력하는 printf() 문이 더 이상 수행되지 않도록 위의 코드를 변경하라.

4. **[extern 키워드의 사용] extern은 다른 코드 파일에서 정의한 변수를 참조할 때에 사용된다.**

```
// s1.c 파일
#include <stdio.h>
extern int x;      // ①

int main(void)
{
    extern int y; // ②
    x = 10;
    y = 20;

    return 0;
}
int y;
```

(a) 위의 프로그램에서는 extern 키워드가 2곳에서 사용되었다. 어떤 목적으로 사용되었는지를 설명하라.

(b) 위의 프로그램을 컴파일하여 보라. 어떤 오류가 발생하는가? 오류의 원인은 무엇인가?

(c) 또 하나의 소스 파일인 s2.c를 만들고 그 곳에 변수 x를 전역 변수로 선언한 후에 s1.c와 같이 컴파일, 링크하여 실행 파일을 생성하여 보라. 오류가 없어지는가? 주의할 점은 같은 프로젝트 안에 소스 파일 s2.c를 생성하여야 한다.

(d) s2.c 파일에서 변수 x를 정의할 때에 앞에 static을 붙이면 어떻게 되는가? s1.c의 extern int x와 연결되는가? 전역 변수 앞에 static을 붙이면 어떤 의미가 되는가?

5. **[재귀 호출] 다음은 1부터 n까지의 합을 재귀적으로 구하는 소스이다.**

```
#include <stdio.h>
int recursive(int n);

int main(void)
```

```
{
    int n = 0, sum;

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

    sum = recursive(n);
    printf("%d \n", sum);

    return 0;
}
int recursive(int n)
{
    // ①
    if(n <= 1) return 1;  // ②
    else return n + recursive(n-1); // ③
}
```

(a) recursive() 함수가 호출되면, 'recursive' 라는 함수 이름과 매개 변수 n의 값을 출력하는 문장을 ①의 위치에 삽입하라. 그 결과를 설명하라.

(b) ② 문장을 삭제하고 실행하여 보라. 어떤 결과가 얻어지는가?

(c) ② 문장에서 recursive(n-1)을 recursive(n)으로 변경하여 실행하여 보라. 어떤 결과가 얻어지는가?

(d) 재귀 호출을 사용하지 않고 반복 구조를 사용하여 recursive()를 다시 작성하여 보라.

(e) $1^2 + 2^2 + \cdots + n^2$을 재귀적으로 계산하도록 recursive()를 변경하여 보라.