실 습 문 제 제8장

1. **[함수의 호출 개념 이해]**

함수가 호출되면 제어의 흐름이 어떻게 변경되는지를 실습하여 본다.

```
#include <stdio.h>

void f1(void);
void f2(void);

int main(void)
{
    int i;

    f1();
    for(i = 0;i < 3; i++)
        f2();
    f1();

    return 0;
}

void f1(void)
{
    printf("/**********************/\n");
}

void f2(void)
{
    printf("/*                    */\n");
}
```

실행결과(기록하기)

(a) 이 프로그램의 실행 결과를 예측하여 보라. 실제로 실행하여 예측된 실행 결과와 비교하여 보라.

(b) 함수들의 이름을 의미있는 이름으로 변경하여 보라.

(c) 함수가 시작되면 무조건 함수의 이름을 화면에 출력하도록 프로그램을 수정하라. 어떤 출력이 생성되는가?

2. **[매개 변수가 없는 함수의 작성 및 호출]**

```c
#include <stdio.h>

void print_header(void);

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

void print_header(void)
{
    ____________________
    ____________________
}
```

(a) 다음과 같은 출력 화면을 생성하도록 print_header() 함수의 몸체를 채워서 완성하라.

출력화면

```
/*****************************/
/* 학번: 11111111            */
/* 이름: 홍길동              */
/*****************************/
```

(b) 이 프로그램에서 다음과 같은 출력 화면을 얻으려면 어떻게 수정하면 되는가? 함수를 사용하는 경우와 함수를 사용하지 않는 경우를 비교하여 보라.

출력화면

```
/*****************************/
/* 학번: 11111111            */
/* 이름: 홍길동              */
/*****************************/
/*****************************/
/* 학번: 11111111            */
/* 이름: 홍길동              */
/*****************************/
```

3. [매개 변수가 있는 함수의 작성 및 호출]

이번 문제에서는 매개 변수가 있는 간단한 함수를 작성하여 실행하여 본다. float형 변수 2개의 평균을 계산하는 함수를 작성하여 보자.

```
#include <stdio.h>

_____________________   // 함수 원형 정의

int main(void)
{
    float a, b, c;

    a = 2.0f;
    b = 3.0f;
    c =                 // a와 b를 인수로 하여 average()를 호출
    // ①
    printf("평균은 %f입니다.\n", c);
    // ③
}

float average(float x, float y)
{
    // ②
    _____________________          // 평균을 계산하여 반환한다.
}
```

실행결과(기록하기)

(a) 위의 프로그램의 빈 칸에 필요한 문장들을 채워보자. 위의 프로그램을 컴파일하고 실행하여 보라. 출력을 기록하라.

(b) scanf()를 이용하여 사용자에게서 받은 두개의 실수의 평균을 구하도록 위의 프로그램을 수정하여 보라.

4. [매개 변수가 있는 함수의 작성 및 호출]

함수를 사용하지 않고 작성된 다음 프로그램을, 함수를 사용하여 다시 작성하여 보자. 이 프로그램은 0부터 100까지의 합을 구한다.

```
#include <stdio.h>

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

    for(i = 0;i <= 100; i++)
        sum += i;

    printf("0부터 %d까지의 합은 %d입니다.\n", n, sum);
    return 0;
}
```

⬇

```
#include <stdio.h>

_____________________________          // 함수 원형 정의

int main(void)
{
    int sum = 0;

    _____________________________      // 함수 호출
    printf("0부터 100까지의 합은 %d입니다.\n", sum);
}

_____________________________          // 함수 헤더 정의
{
    int i, result = 0;
    _____________________________
    _____________________________
    _____________________________
    _____________________________
}
```

(a) 함수의 이름은 get_sum()이라고 하고 int형의 정수 n을 받아서 0에서부터 n까지의 합을 구하여 반환한다고 가정하자. 위의 프로그램의 빈칸을 채워서 완성하여 보라. 컴파일하고 실행하여 보라.

(b) 만약 0에서 100까지의 합과 0에서 200까지의 합을 차례대로 계산한다고 할 때, 함수를 사용하는 프로그램과 함수를 사용하지 않는 프로그램은 어떤 차이점이 있는가? 매개 변수를 사용하는 장점은 무엇인가?

(c) get_sum() 함수가 2개의 인수 s, e를 받아서 s에서 e까지의 정수의 합을 계산하여 반환하도록 수정하여 보라. 수정된 get_sum()을 호출하여 10에서 100까지의 합을 계산하여 보라.

5. [함수와 관련된 오류]

```
#include <stdio.h>

double fpower(double r, int n);            // ①

int main(void)
{
    double prod;

    prod = fpower(3.0, 2);                 // ②
    printf("%f\n", prod);
```

```
}

double fpower(double r, int n)
{
    int i;
    double result = 1.0;

    for(i = 0; i < n; i++)
        result *= r;

    return result;
}
```

실행결과(기록하기)

```

```

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

(b) 문장 ①의 함수 원형 선언에서 끝에 있는 세미콜론을 제거하고 컴파일 하여 보라. 어떤 일이 발생하는가?

(c) 문장 ①의 함수 원형 선언에서 반환형을 나타내는 double을 제거하고 컴파일 하여 보라. 어떤 일이 발생하는가?

(d) 문장 ②에서 함수 호출을 할 때 fpower(3.0); 와 같이 하나의 인수만을 가지고 호출하여 보라. 어떤 일이 발생하는가?

(e) 함수 원형을 완전히 제거한 후에 컴파일하여 실행하여 보라. 어떤 일이 발생하는가?

(f) return 문장을 다음과 같이 변경하면 어떻게 되는가?

```
return (result);
```

6. **[함수 원형]**

다음은 2개의 실수 나눗셈을 수행하는 함수를 사용한 프로그램이다. 나눗셈의 결과는 double형으로 반환된다.

```
#include <stdio.h>

double divide(double n, double d);  // ①

int main(void)
{
    double result;

    result = divide(3.0, 10.0);            // ②
```

```
    printf("나눗셈 결과는 %f입니다.\n", result);
}

double divide(double n, double d)
{
    return n / d;
}
```

실행결과(기록하기)

(a) 위의 프로그램을 컴파일하고 실행하여 보라. 출력을 기록하라.

(b) 위의 함수 원형이 정의된 문장 ①을 주석 처리한 후에 컴파일하여 보라. 어떤 오류가 발생하는가? 오류 메시지를 해석하고 설명하여 보라. 실습 후 원상태로 복구하라.

(c) 함수 원형 정의를 다음과 같이 변경하고 실행하여 보라. 어떤 차이점이 있는가?

```
double divide(double, double);      // ①
```

(d) 문장 ②를 `result = divide(3, 10);`으로 변경하여 실행하여 보라. 함수의 인수로 실수 대신 정수를 주었는데도 같은 결과가 생성되는 이유는 무엇인가?

(e) 문장 ②를 `result = divide(3, 10);`으로 변경한 상태에서 함수 원형 정의를 다음과 같이 변경하여 보라. 즉 이번에는 함수의 인수에 대해서는 정의하지 않는다. 어떤 결과가 생성되는가? 실습 후 원상태로 복구하라.

```
double divide();  // ①
```

(f) 위의 함수 원형이 정의된 문장 ①을 `main()` 함수 안으로 이동하여서 컴파일, 실행하여 보라. 차이점은 무엇인가? 실습 후 원상태로 복구하라.

(g) `divide()` 함수 안에서 매개 변수 `d`가 0이 아닌 경우에만 나눗셈이 되도록 코드를 수정하여 보라.

(h) 함수 원형 정의를 제거하고 `divide()` 함수의 위치와 `main()` 함수의 위치를 서로 바꾼 후에 프로그램을 컴파일하고 실행하여 보라. 어떤 차이점이 있는가?

(i) 같은 함수 원형을 여러 번 나열해도 문제가 없는지를 확인하기 위하여 문자 ①을 반복하여 보라. 어떤 결과가 나타나는가? 그리고 그 이유는 무엇인가?

```
double divide(double n, double d);  // ①
double divide(double n, double d);  // ①
```

7. [함수의 반환값]

함수는 단 하나의 값만을 반환할 수 있다. 값을 반환하는 문장은 `return` 키워드를 사용한다. 3개의 정수 중에서 가장 큰 수를 결정하여 반환하는 함수인 `get_max3()`을

작성하여 보자.

```
#include <stdio.h>

int get_max2(int x, int y);
int get_max3(int x, int y, int z);

int main(void)
{
    int max;

    max = get_max3( 1, 2, 3 ); // 정수 1, 2, 3 중에서 가장 큰 수를 반환한다.
    printf("가장 큰 수는 %d입니다.\n", max);

    return 0;
}

int get_max2(int x, int y)
{
    if( x > y )
    _____________________________
    else
    _____________________________
}

int get_max3(int x, int y, int z)
{
    _____________________________
    _____________________________
    _____________________________
}
```

(a) get_max2() 함수의 반환값을 이용하지 않고도, 세 수 중에 가장 큰 값을 찾아낼 수 있도록 get_max3() 함수를 작성하여 보라.

(b) if-else 제어 구조를 이용하고, 몸체 안에서 get_max2() 함수를 사용하는 꼴로 get_max3() 함수를 작성하여 보라.

(c) get_max2() 함수의 반환값을 이용하여서 get_max3() 함수를 작성하는 경우, 한 줄로도 작성이 가능한가?

8. [인수와 매개 변수]

함수 호출시에 함수에 주어지는 값들을 인수(argument)라고 하고 함수의 정의에서 인수를 받기 위한 변수들을 매개 변수(parameter)라고 한다. 함수가 호출되면 인수의 값이 매개 변수로 복사되어서 전달된다.

```c
#include <stdio.h>

void f(int x, int y);

int main(void)
{
    int a, b;

    a = 10;
    b = 20;

    printf("f() 호출전a = %d, b = %d\n", a, b);
    f(a, b);
    printf("f() 호출후a = %d, b = %d\n", a, b);

    return 0;
}

void f(int x, int y)
{
    x = 30;
    y = 40;
    printf("f() x = %d, y = %d\n", x, y);
}
```

실행결과(기록하기)

(a) 위의 프로그램을 컴파일하여 실행하고 결과를 기록하라.

(b) 위의 프로그램에서 인수와 매개 변수를 지적하라. 인수와 매개 변수의 관계를 설명하라.

(c) main()의 변수 a와 b를 x와 y로 변경한 후에 컴파일하여 실행하여 보라. 결과가 달라지는가?