프논 9th 수업에서 다루어지는 것들: 함수(Function): 값 반환(value returning) 타입

예1) void 함수 vs. value returning 함수

```c
#include <stdio.h>
void printSum(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;
   printSum(n1, n2); // 호출
   return 0;
}
void printSum(int a, int b) //정의
{
   int sum;
   sum = a + b;
   printf("%d", sum);
   return; //생략가능
}
```

```c
#include <stdio.h>
int calculateSum(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;
   printf("%d", calculateSum(n1, n2)); //호출
   return 0; //생략가능
}
int calculateSum(int a, int b) //정의
{
   int sum;
   sum = a + b;
   return sum; // 반드시 써야함, sum은 정수형이어야
}
// 반환값은 함수가 호출한 곳으로 작업의 결과값을 전달한다
// 반환값은 단 하나만 가능
```

예2) void 함수 vs. value returning 함수

```c
#include <stdio.h>
void printBig(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;
   printBig(n1, n2); // 호출
   return 0;
}
void printBig(int a, int b) //정의
{
   int big;
   if (a < b)
      big = b;
   else
      big = a;

   printf("큰수는 %d", big);
   return; //생략가능
}
```

```c
#include <stdio.h>
int getBig(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;

   int result;
   result = getBig(n1, n2);
   printf("큰 수는%d", result); //호출
   return 0; //생략가능
}
int

big
```

예3) 매개변수가 없는 value returning 함수의 사용예

```c
//main함수만

#include <stdio.h>
int main(void)
{
   int n1, n2;

   printf("정수를 입력하세요:");
   scanf("%d", &n1);

   printf("정수를 입력하세요:");
   scanf("%d", &n2);

   printf("입력된 수는 %d와 %d입니다\n", n1, n2);
}
```

```c
#include <stdio.h>
int getNumber(void); //원형
int main(void)
{
   int n1, n2;

   n1 = getNumber(); // 호출
   n2 = getNumber(); // 호출

   printf("입력된 수는 %d와 %d입니다\n", n1, n2);
}
int getNumber(void) //정의
{
   int num;
   printf("정수를 입력하세요:");
   scanf("%d", &num);
   return num; // 반드시 써야
}
```

예x) 코드 읽어 실행결과 예측하기

```c
#include <stdio.h>
int getBig(int a, int b); //원형
int main(void)
{
       int n1 = 10, n2 = 20;

       int result;
       result = getBig(n1, n2);
       printf("큰 수는 %d\n", result);
       return 0;
}
int getBig(int a, int b)
{
       int big;
       if (a < b)
              big = b;
       else
              big = a;

       printf("큰 수 계산됨\n");
       return big;
}
```

예y) 코드 읽어 실행결과 예측하기

```c
#include <stdio.h>
int getBig(int a, int b);
int main(void)
{
       int n1 = 10, n2 = 20;

       int result;
       getBig(n1, n2);

       return 0;
}
int getBig(int a, int b)
{
       int big;
       if (a < b)
              big = b;
       else
              big = a;

       return big;
}
```

▶함수(Function) 참고자료

LABHW 10: void 함수
- 매개변수 없는 void함수의 연습
- 매개변수를 가지는 void함수의 연습

LABHW 11: value returning 함수
- 매개변수 없는 value returning함수의 연습
- 매개변수를 가지는 value returning함수의 연습

void 함수

```
#include <stdio.h>
void f1(void); //원형
int main(void)
{
   f1(); //호출
   f1(); //호출
}
void f1(void) //정의
{
   int num = 10;
   printf("%d", num);
   return; //생략가능
}
```

```
#include <stdio.h>
void f2(int a); //원형
int main(void)
{
   int n = 10;
   f2(n);  //호출
   f2(20); //호출
}
void f2(int a) //정의
{
   printf("%d", a * a);
   return; //생략가능
}
```

```
#include <stdio.h>
void f3(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;
   f3(n1, n2); // 호출
   f3(100, 200);
}
void f3(int a, int b) //정의
{
   int sum;
   sum = a + b;
   printf("%d", sum);
   return; //생략가능
}
```

value returning 함수 - 호출 함수가 값을 가진다!

```
#include <stdio.h>
int g1(void); //원형
int main(void)
{
   printf("%d", g1()); //호출
   return 0; //생략가능
}
int g1(void) //정의
{
   int num = 10;
   return num; //정수형
}
```

```
#include <stdio.h>
int g2(int a); //원형
int main(void)
{
   int n = 10, square;
   square = g2(n); //호출
   printf("%d", square);
   return 0; //생략가능
}

int g2(int a) //정의
{
   return a * a; //정수형
}
```

```
#include <stdio.h>
int g3(int a, int b); //원형
int main(void)
{
   int n1 = 10, n2 = 20;
   int total;

   total = g3(n1, n2); //호출
   printf("%d", total);
   return 0; //생략가능
}
int g3(int a, int b) //정의
{
   int sum;
   sum = a + b;
   return sum; //정수형
}
```

- 원형에서는 변수이름 생략가능 예: void g3(int, int); //원형
- 호출하려는 함수를 먼저 정의하는 경우 원형 생략이 가능하다
- main에서의 return 0은 성공적으로 프로그램이 끝나는 경우를 의미하며 생략가능

LAB 11

■ **LAB11_0 제곱 출력하기**

□ **LAB11_0_1(매개변수가 없는 value returning 함수의 연습)**
아래의 오른쪽의 프로그램을 완성하고 왼쪽의 void함수와 비교해보라.

실행예(입력없음)

```
제곱은 25
```

```
#include <stdio.h>
void printSquare1(void);
int main(void)
{
  printSquare1();
}

void printSquare1(void)
{
  int x = 5;
  printf("제곱은 %d\n", x * x);
}
```

```
#include <stdio.h>
int square1(void);
int main(void)
{
  int result;
  result = square1();
  printf("제곱은 %d\n", result);
}

int square1(void)
{
   int x = 5;
   ...
}
```

□ **LAB11_0_2(매개변수를 가지는 value returning 함수의 연습)**
아래의 오른쪽 프로그램을 완성하고 왼쪽의 void함수와 비교해보라.

실행예(입력없음)

```
제곱은 25
```

```
#include <stdio.h>
void printSquare2(int);
int main(void)
{
   printSquare2(5);
}

void printSquare2(int x)
{
  printf("제곱은 %d\n", x * x);
}
```

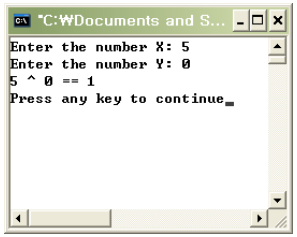
```
#include <stdio.h>
//원형 채우기
int main(void)
{
  int result;
  result = square2(5);
  printf("제곱은 %d\n", result);
}

int square2(          )
{
  ...
}
```

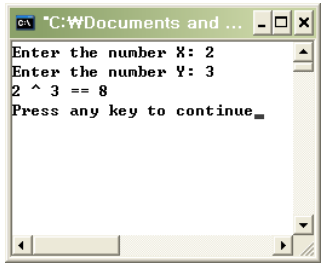
■ **LAB11_1 두 수사이의 합 구하기(매개변수를 가지는 value returning 함수의 연습)**
아래의 실행결과를 갖는 프로그램을 sum1ToN 함수를 사용하여 써라.

```
1 부터 5 까지의 합은 15
1 부터 7 까지의 합은 28
1 부터 10 까지의 합은 55
```

```
#include <stdio.h>
int sum1ToN(int n); // 함수의 원형
int main(void)
{
   printf("1 부터 %d 까지의 합은 %d\n", 5, sum1ToN(5));
   printf(                                          );
   printf(                                          );
}

int sum1ToN(int n) // 함수의 정의
{
   // 1 부터 n 까지의 합을 계산해서  그 값을 return 한다.

}
```

■ **LAB11_2 거듭제곱을 구하는 함수**
두 수(x와 y)를 입력받아서 x의 y승을 구하는 함수를 정의하세요.






```
#include <stdio.h>
int pow(int a, int b);
int main(void)
{
       int x, y;

       printf("Enter the number X: ");
       scanf("%d", &x);
       printf("Enter the number Y: ");
       scanf("%d", &y);

       printf(...); /* 이부분을 함수 호출로 채우세요 */
}

/* 여기에 함수 원형과 일치하는 함수를 정의하세요*/
```

■ **LAB11_3 성적을 입력받아서 학점을 결정해서 출력하는 프로그램을 작성하라.**
(80점 이상이면 A, 50점 이상이면 B, 그 외에는 F).
main함수는 그대로 사용한다.

실행예:
Enter a score: 85
Grade is A!

```
#include <stdio.h>
int scoreReading(void);
char gradeDecision(int s1);
int main(void)
{
       int score;
       score = scoreReading();
       printf("Grade is %c!\n", gradeDecision(score));
}

int scoreReading(void)
{
       int s;
       printf("Enter a score: ");
       ...
}

char gradeDecision(int s1)
{
       //별도의 지역변수를 선언하지 말라
}
```

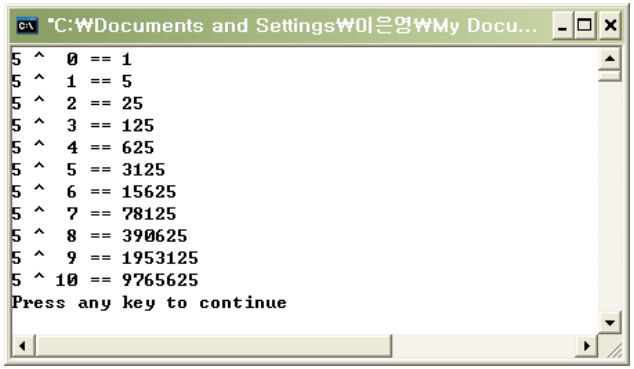
HW 11

■ **HW11_1(매개변수를 가지는 value returning 함수의 연습)**
아래의 실행결과를 갖는 프로그램을 sumMToN 함수를 사용하여 써라.

```
2부터 5까지의 합은 14
3부터 7까지의 합은 25
2부터 10까지의 합은 54
```

```
#include <stdio.h>
int sumMToN(int m, int n); // 함수의 원형

int main(void)
{
   printf("%d 부터 %d 까지의 합은 %d\n", 2, 5, ___   _); //함수의 호출부분 추가
   printf(                                          );
   printf(                                          );
}

int sumMToN(int m, int n) // 함수의 정의
{

   // 이곳에 m부터 n까지의 합을 계산하여
   // 그 합을 return하는 함수 정의부분을 코딩한다




}
```

■ **HW11_2 (함수의 호출)**
LAB11_2에서 정의한 pow(int a,int b) function을 이용해서 5의 0승에서 10승까지 화면에 출력하는 프로그램을 작성하세요.
주의사항 : LAB11_2에서 정의한 pow함수를 그대로 사용하라.



■ **HW11_3 두개의 성적을 입력받아서 평균을 계산해서 학점을 결정해서 출력하는 프로그램을 작성하라. 80점 이상이면 A, 50점 이상이면 B, 그 외에는 F**

실행예:
```
Enter a score: 50
Enter a score: 60
Grade is B!
```

```
#include <stdio.h>
int scoreReading(void);
char gradeDecision(int s1, int s2);
int main(void)
{
	int score1, score2;

	//scoreReading()을 이용해서 score1과 score2를 입력받는다.
	...


	printf("Grade is %c!", gradeDecision(..............));
}

int scoreReading(void)
{
	int s;
	printf("Enter a score: ");
	...


}

char gradeDecision(int s1, int s2)
{
	int average;
	char grade;// 반드시 사용









}
```

■ **HW11_4**

□ **HW11_4_0(소수 판별)**

입력받은 정수(2이상)가 소수(prime number)인지를 검사하여 "소수이다" 혹은 "소수가 아니다"를 출력하도록 프로그램을 작성하라. 이때는 main 함수만 사용한다. 실행예는 아래와 같다.

실행예 1
```
Enter a number: 8
소수가 아닙니다.
```

실행예 2
```
Enter a number: 13
소수입니다.
```

□ **HW11_4_1(논리 8의 함수화)**

단계1 : 함수 작성 & 테스트
num이 소수이면 1을 아니면 0을 반환하는
함수 isPrime()을 작성하라. 원형은 다음과 같다.
```
int isPrime(int num) ;
```

함수를 작성한후 main함수를 작성하여 테스트해보라

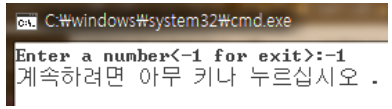
실행예는 다음과 같다.

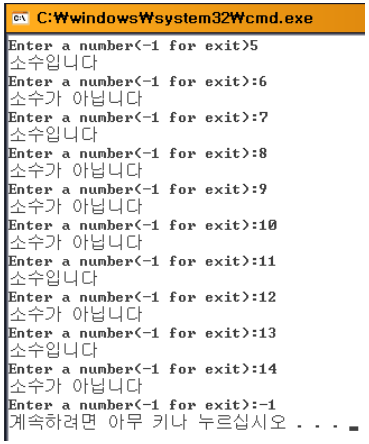
실행예 :
```
Enter a number : 5
소수입니다.
```

실행예 :
```
Enter a number : 8
소수가 아닙니다.
```

단계2 : 테스트 드라이버 main함수 작성 :
이를 테스트하기 위한 main함수를 작성하여
아래처럼 실행되게 하라

힌트 : 감시값 제어 반복문 사용,
do while과 while중 어느 것을 사용해야하나 ?





프로젝트(메르센 소수)

■ **HW(메르센 소수)**

메르센 소수를 출력하는 프로그램을 작성하라. 메르센 소수란 무엇인가?

메르센 소수란 무엇인가?
http://program.tving.com/tvn/hotbrain/1/Vod/View/180085 (4분 52초)

- 가능한 많은 메르센 소수를 출력하라.
- 몇 개까지 가능한가?
  일단 n = 2..30으로 프로그램을 돌려본 후
  2..35까지 시도해보자.
- 실행결과 A) n이 32되는 시점에서 에러가 난다. 에러가 나는 이유는?
- 실행결과 B)처럼 에러가 나지 않게 하려면 어떻게 해야하나?

실행결과 A)

```
C:₩windows₩system32₩cmd.exe
Enter a number:35
(2^2 - 1) = 3은 메르센 소수이다
(2^3 - 1) = 7은 메르센 소수이다
(2^5 - 1) = 31은 메르센 소수이다
(2^7 - 1) = 127은 메르센 소수이다
(2^13 - 1) = 8191은 메르센 소수이다
(2^17 - 1) = 131071은 메르센 소수이다
(2^19 - 1) = 524287은 메르센 소수이다
(2^31 - 1) = 2147483647은 메르센 소수이다
(2^32 - 1) = -1은 메르센 소수이다
(2^33 - 1) = -1은 메르센 소수이다
(2^34 - 1) = -1은 메르센 소수이다
(2^35 - 1) = -1은 메르센 소수이다
계속하려면 아무 키나 누르십시오 . . .
```

실행결과 B)

```
C:₩windows₩system32₩cmd.exe
Enter a number:35
(2^2 - 1) = 3은 메르센 소수이다
(2^3 - 1) = 7은 메르센 소수이다
(2^5 - 1) = 31은 메르센 소수이다
(2^7 - 1) = 127은 메르센 소수이다
(2^13 - 1) = 8191은 메르센 소수이다
(2^17 - 1) = 131071은 메르센 소수이다
(2^19 - 1) = 524287은 메르센 소수이다
(2^31 - 1) = 2147483647은 메르센 소수이다
계속하려면 아무 키나 누르십시오 . . .
```

```
#include <stdio.h>
//2의 n승을 반환하는 함수
int twoToThePower(int n)
{




}

// x가 소수이면 1을 아니면 0을 반환하는 함수
int isPrime(int x)
{




}

int main(void)
{




}
```