1. 간단한 계산기 (산술, 복합 대입, 증감 연산자)

1. 기능
   - 사용자로부터 정수 두 개를 입력받는다(예: x, y).
   - 아래 연산들을 수행한 뒤, 결과를 각각 출력한다.
     ① x + y, x - y, x * y, x / y, x % y (산술 연산)
     ② x += y (복합 대입 후 x 출력)
     ③ x++, ++y (증감 연산 전후 값을 출력)

2. 요구사항
   - 정수 나눗셈(몫), 나머지 연산(%) 주의.
   - 복합 대입 연산(+=) 수행 전과 후를 비교해본다.
   - x++(후위), ++y(전위)를 각각 적용하여 결과를 관찰한다.

```
정수 2개를 입력하세요: 10 3

[산술 연산 결과]
10 + 3 = 13
10 - 3 = 7
10 * 3 = 30
10 / 3 = 3
10 % 3 = 1

[복합 대입]
x += y 실행 후, x = 13

[증감 연산]
x++ 실행 결과: x(출력 후) = 13, 실제 저장된 x = 14
++y 실행 결과: y = 4
```

1. 간단한 계산기 (산술, 복합 대입, 증감 연산자)

```c
/*
  파일명: op_ex1.c
  작성자: 홍길동
  설명 : 정수 2개 입력받아 산술, 복합 대입, 증감 연산자 실습
*/
#include <stdio.h>

int main(void)
{
    int x, y;
    printf("정수 2개를 입력하세요: ");
    scanf("%d %d", &x, &y);

    // 1) 산술 연산
    printf("\n[산술 연산]\n");
    printf("%d + %d = %d\n", x, y, x + y);
    printf("%d - %d = %d\n", x, y, x - y);
    printf("%d * %d = %d\n", x, y, x * y);
    printf("%d / %d = %d\n", x, y, x / y);  // 정수 나눗셈 (몫)
    printf("%d %% %d = %d\n", x, y, x % y); // 나머지 연산 시 %% 사용

    // 2) 복합 대입 (x += y)
    printf("\n[복합 대입]\n");
    x += y; // x = x + y;
    printf("x += y 실행 후, x = %d\n", x);

    // 3) 증감 연산자
    printf("\n[증감 연산]\n");
    printf("x++ 실행 전: %d, 출력 후: %d, 최종 x = %d\n", x, x++, x);
    printf("++y 실행 후: %d, y = %d\n", ++y, y);

    return 0;
}
```

2. 관계·논리 연산자 실습 (조건 판별)

1. 기능
   - 사용자로부터 점수(0 이상 100 이하)를 입력받는다.
   - 관계 연산자를 사용해 다음을 판별하고, 결과를 출력한다.
     1. 60점 이상인지? (합격/불합격)
     2. 90점 이상 100점 이하인지? (A 구간인지 여부)
     3. 0점 미만 또는 100점 초과인지? (입력 오류)
   - 논리 연산자(&&, ||, !)를 적극 활용한다.

2. 요구사항
   - 점수가 0 미만이거나 100 초과인 경우에는 "잘못된 점수"라는 경고 문구를 출력한다.
   - (선택) ?: 조건 연산자를 활용해 합격/불합격을 간단히 표현해도 좋다.

```
점수를 입력하세요: 95

입력받은 점수: 95
60점 이상인가? -> 합격
90점 이상 100점 이하인가? -> 네, A 구간입니다.
0점 미만 혹은 100점 초과인가? -> 아니오
```

2. 관계·논리 연산자 실습 (조건 판별)

```c
/*
  파일명: op_ex2.c
  작성자: 홍길동
  설명 : 점수를 입력받아 관계/논리 연산, 조건 연산자 등 사용
*/
#include <stdio.h>

int main(void)
{
    int score;
    printf("점수를 입력하세요 (0~100): ");
    scanf("%d", &score);

    // 범위 검사
    if(score < 0 || score > 100)
    {
        printf("잘못된 점수입니다!\n");
        return 0;
    }

    // 60점 이상 여부
    // 조건 연산자 ?:
    (score >= 60) ? printf("60점 이상 -> 합격\n")
                  : printf("60점 미만 -> 불합격\n");

    // 90점 이상 100점 이하 여부
    if(score >= 90 && score <= 100)
        printf("90점 이상 100점 이하 -> A 구간입니다.\n");
    else
        printf("A 구간이 아닙니다.\n");

    return 0;
}
```

3. 비트 연산자 (비트 단위 조작)

1. 기능
   - 사용자로부터 양의 정수(예: 10진수)를 입력받는다.
   - 아래와 같은 비트 연산 결과를 출력한다.
     1. ~num (비트 NOT)
     2. num << 1 (왼쪽 시프트 1)
     3. num >> 2 (오른쪽 시프트 2)
     4. (선택) num & 0xF, num | 0xF, num ^ 0xF 등 간단한 AND, OR, XOR 연산
2. 요구사항
   - 원본 수(num)와 결과 값을 10진수, 16진수 등으로 출력해보자.
   - 음수가 입력되면 어떻게 되는지도 실험해볼 수 있지만, 일단 양의 정수를 가정한다.

```
정수를 입력하세요: 10

입력값 (10진수) = 10, (16진수) = 0xa

~num (비트 NOT) = -11 (10진수)
num << 1 (왼쪽 시프트) = 20 (10진수), 0x14 (16진수)
num >> 2 (오른쪽 시프트) = 2 (10진수), 0x2 (16진수)
```

3. 비트 연산자 (비트 단위 조작)

```c
/*
  파일명: op_ex3.c
  작성자: 홍길동
  설명 : 비트 연산자 실습 (NOT, <<, >>, 기타)
*/
#include <stdio.h>

int main(void)
{
    unsigned int num; // 양의 정수 가정
    printf("양의 정수를 입력하세요: ");
    scanf("%u", &num);

    // 원본 출력
    printf("\n입력값 (10진수) = %u, (16진수) = 0x%x\n", num, num);

    // 1) 비트 NOT
    printf("~num = %d (10진수), 0x%x (16진수)\n", ~num, ~num);

    // 2) 왼쪽 시프트 1
    unsigned int leftShift = num << 1;
    printf("num << 1 = %u (10진수), 0x%x (16진수)\n", leftShift, leftShift);

    // 3) 오른쪽 시프트 2
    unsigned int rightShift = num >> 2;
    printf("num >> 2 = %u (10진수), 0x%x (16진수)\n", rightShift, rightShift);

    // (선택) AND, OR, XOR 예시
    unsigned int andVal = num & 0xF;
    unsigned int orVal  = num | 0xF;
    unsigned int xorVal = num ^ 0xF;
    printf("\nnum & 0xF  = %u (10진), 0x%x (16진)\n", andVal, andVal);
    printf("num | 0xF  = %u (10진), 0x%x (16진)\n", orVal, orVal);
    printf("num ^ 0xF  = %u (10진), 0x%x (16진)\n", xorVal, xorVal);

    return 0;
}
```