1. push

SP의 값을 1씩 증가시키면서 call_stack에 데이터 값을 저장하고,

sprintf_s 함수를 사용해 stack_info[SP] 버퍼에 데이터에 대한 설명을 출력한다.

```
// func2의 스택 프레임 형성 (함수 프롤로그 + push)
// arg1, arg2를 스택에 push
call_stack[++SP] = arg2;
stack_info[SP][0] = '\0';
sprintf_s(stack_info[SP], sizeof(stack_info[SP]), "%s", "arg2");
call_stack[++SP] = arg1;
stack_info[SP][0] = '\0';
sprintf_s(stack_info[SP], sizeof(stack_info[SP]), "%s", "arg1");
// Return Address를 스택에 push
call_stack[++SP] = -1;
stack_info[SP][0] = '\0';
sprintf_s(stack_info[SP], sizeof(stack_info[SP]), "%s", "Return Address");
// SFP를 스택에 push
call_stack[++SP] = FP;
stack_info[SP][0] = '\0';
sprintf_s(stack_info[SP], sizeof(stack_info[SP]), "%s", "func2 SFP");
// 현재 스택 프레임을 FP에 저장
FP = SP;
// var_2를 스택에 push
call_stack[++SP] = var_2;
stack_info[SP][0] = '\0';
sprintf_s(stack_info[SP], sizeof(stack_info[SP]), "%s", "var_2");
```

결과:

```
====== Current Call Stack ======
5 : var_1 = 100    <=== [esp]
4 : func1 SFP    <=== [ebp]
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================

====== Current Call Stack ======
10 : var_2 = 200    <=== [esp]
9 : func2 SFP = 4    <=== [ebp]
8 : Return Address
7 : arg1 = 11
6 : arg2 = 13
5 : var_1 = 100
4 : func1 SFP
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================

====== Current Call Stack ======
15 : var_4 = 400    <=== [esp]
14 : var_3 = 300
13 : func3 SFP = 9    <=== [ebp]
12 : Return Address
11 : arg1 = 77
10 : var_2 = 200
9 : func2 SFP = 4
8 : Return Address
7 : arg1 = 11
6 : arg2 = 13
5 : var_1 = 100
4 : func1 SFP
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================
```

2. pop

pop_stack 함수를 작성, SP를 1씩 감소시켜 스택에서 데이터를 가장 최근에 저장된 것부터 제거한다.

```
void pop_stack() {
    if (SP == -1) {
        printf("Stack underflow!\n");
        return;
    }
    SP--;
}
```

```
// func2의 스택 프레임 제거 (함수 에필로그 + pop)
pop_stack(); // var_2
FP = call_stack[FP];
pop_stack(); // func2 SFP
pop_stack(); // Return Address
pop_stack(); // arg1
pop_stack(); // arg2

print_stack();
```

결과:

```
====== Current Call Stack ======
15 : var_4 = 400    <=== [esp]
14 : var_3 = 300
13 : func3 SFP = 9    <=== [ebp]
12 : Return Address
11 : arg1 = 77
10 : var_2 = 200
9 : func2 SFP = 4
8 : Return Address
7 : arg1 = 11
6 : arg2 = 13
5 : var_1 = 100
4 : func1 SFP
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================

====== Current Call Stack ======
10 : var_2 = 200    <=== [esp]
9 : func2 SFP = 4    <=== [ebp]
8 : Return Address
7 : arg1 = 11
6 : arg2 = 13
5 : var_1 = 100
4 : func1 SFP
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================

====== Current Call Stack ======
5 : var_1 = 100    <=== [esp]
4 : func1 SFP    <=== [ebp]
3 : Return Address
2 : arg1 = 1
1 : arg2 = 2
0 : arg3 = 3
================================

Stack is empty.
```