push함수 : stack_info배열에 넣을 변수 설명, call_stack에 넣을 값을 매개변수 받음,
스택포인터+1 후 매개변수로 받은 변수 설명과 값을 각각 배열의 스택포인터 번째 번지에 넣음,
strcpy 함수가 while문 보다 간결하고 직관적이라 생각하여 사용

```
void push(char* s, int a) {
    SP++;
    call_stack[SP] = a;
    strcpy(stack_info[SP], s);
}
```

pop함수 : 매개변수로 아무것도 받지 않고 단순히 스택포인터-1을 실행함,
조사해본 결과 실제 메모리 에서도 값을 초기화 시키지 않고 pop함

```
void pop() {
    SP--;
}
```

local_var_push함수 : 지역변수 갯수, 지역변수 배열, 지역변수 설명 배열(2차원)을 매개변수로 받음,
지역변수 갯수 만큼 미리 스택포인터 증가후 for문을 돌리며 변수 설명과 값을 배열에 넣음

```
void local_var_push(int count, int arr[], char str[][10]) {
    SP += count;
    for (int i = 0; i < count; i++) {
        call_stack[SP - count +i +1] = arr[i];
        strcpy(stack_info[SP - count + i + 1], str[i]);
    }
}
```

local_var_pop함수 : 지역변수 갯수를 매개변수로 받아서 그 갯수만큼 스택포인터 감소시킴,
pop함수와 거의 유사

```
void local_var_pop(int count) {
    SP -= count;
}
```

스택프레임 형성 : 매개변수, 리턴주소, SFP(초기값:-1)를 push하고 FP에 SP값을 넣어, 출력시 ebp가 SFP를
가르키게함, 지역변수는 local_var_push를 사용

스택프레임 제거 : 형성의 역순으로 진행, local_var_pop을 사용하여 지역변수를 pop 하고 SFP가 가르키는 주소를
FP에 대입하고 그뒤에 모두 pop함