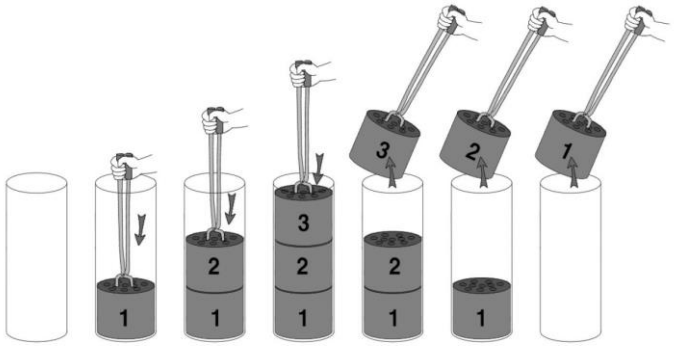
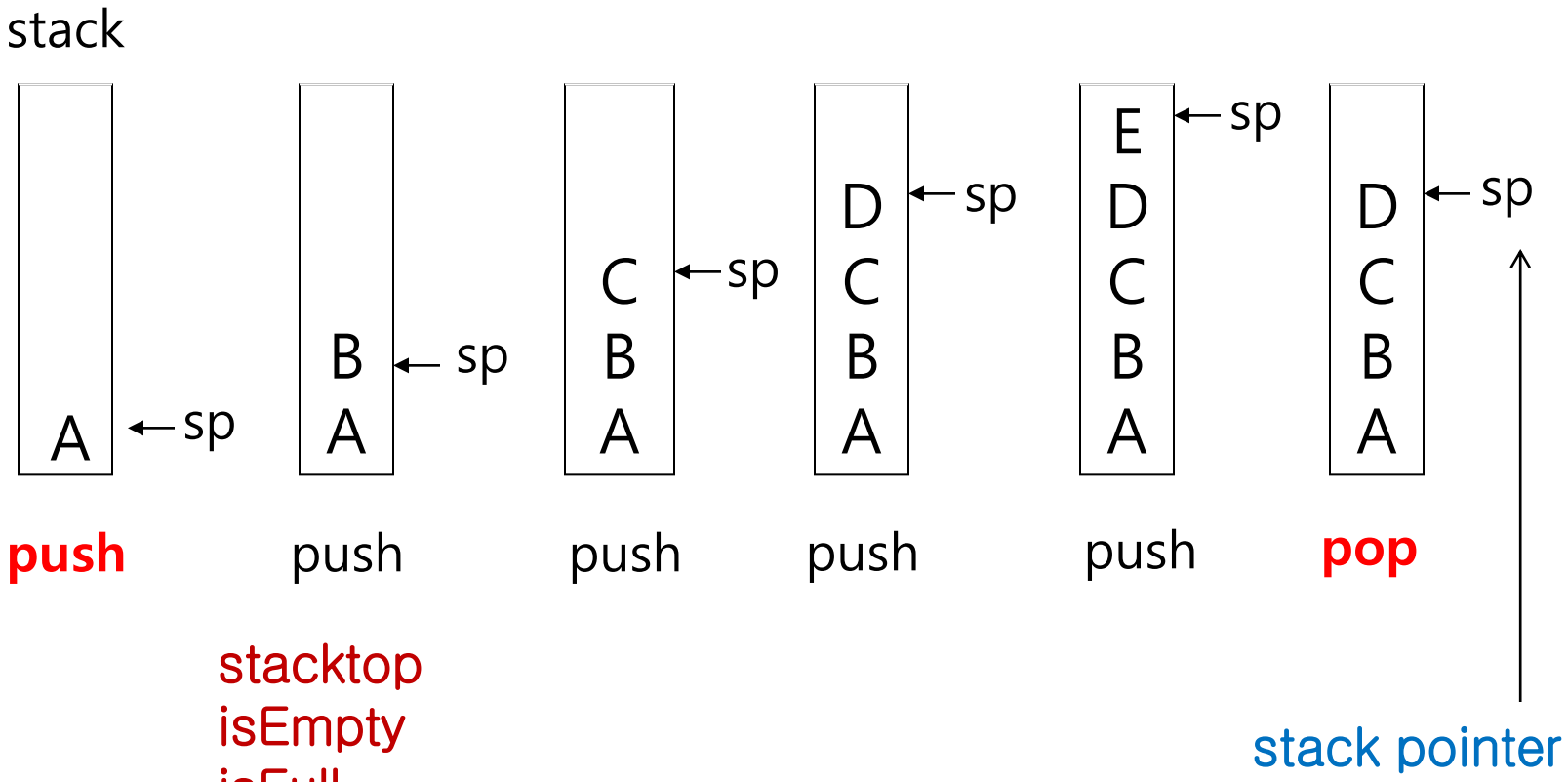
Stack Part 1

What is a Stack ?

- 후입선출 동작 모델 : Last In First Out (LiFO)
  - 입구와 출구가 동일 – 하나의 포인터로 액세스 관리
  - 유한 크기(finite size)
  - 액세스 메커니즘
    - push : 스택에 저장
    - pop : 스택으로부터 읽기
    - 기타 : empty check, full check
  - Example
    - 접시 쌓기, 책 쌓기 등 대부분의 쌓기는 스택 구조에서 동작




stack
A
sp
push
B
A
sp
push
C
B
A
sp
push
D
C
B
A
sp
push
E
D
C
B
A
sp
push
D
C
B
A
sp
pop
stacktop
isEmpty
isFull
stack pointer

컴퓨터에서 응용 예

- More applications related to computer science
  - **Program execution stack**
  - Evaluating expressions

main()

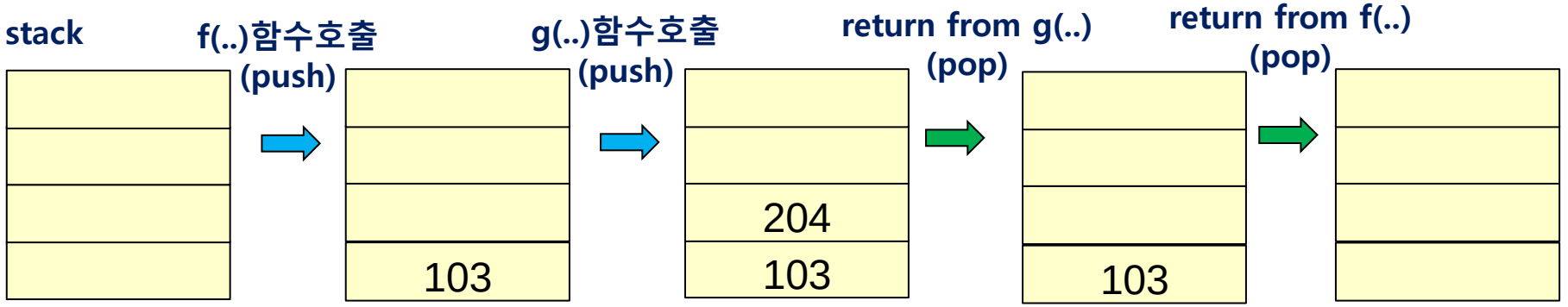
```
100    A;
101    B;
102    f(..);
103    C;
104    return;
```

f(..)

```
200    F1;
201    F2;
203    g(..);
204    F3;
205    return;
```

g(..)

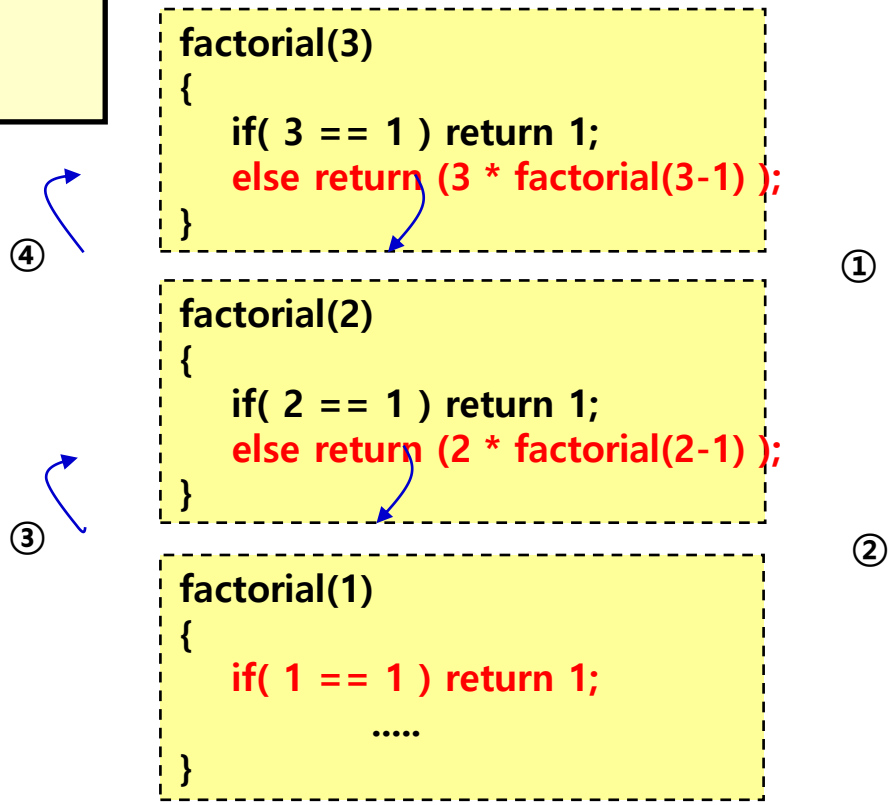
```
300    G1;
301    G2;
302    G3;
303    G4;
304    return;
```



- 재귀함수(recursive function) 예 : factorial 구하기에서 stack 사용

```
int factorial(int n)
{
    if( n == 1 ) return(1);
    else return (n * factorial(n-1) );
}
```

factorial(3) = 3 * factorial(2)
= 3 * 2 * factorial(1)
= 3 * 2 * 1
= 3 * 2
= 6



Stack ADT(Abstract Data Type)

data structure : a **finite ordered list** with zero or more elements.
methods:

for all stack ∈ Stack, item ∈ element, **max_stack_size** ∈ positive integer
Stack createS(max_stack_size) ::=
create an empty stack whose maximum size is
max_stack_size
Boolean isFull(stack, max_stack_size) ::=
if (number of elements in stack == max_stack_size)
return TRUE
else return FALSE
Stack push(stack, item) ::=
if (IsFull(stack)) stack_full
else insert item into top of stack and **return**
Boolean isEmpty(stack) ::=
if(stack == CreateS(max_stack_size))
return TRUE
else return FALSE
Element pop(stack) ::=
if(IsEmpty(stack)) **return**
else remove and return the item on the top of the stack.

Stack ADT의 구현

- **createS(MAX_STACK_SIZE)**
  - Stack의 생성 : (기본 변수 혹은 구조체의) 1차원 배열, 혹은 Linked list

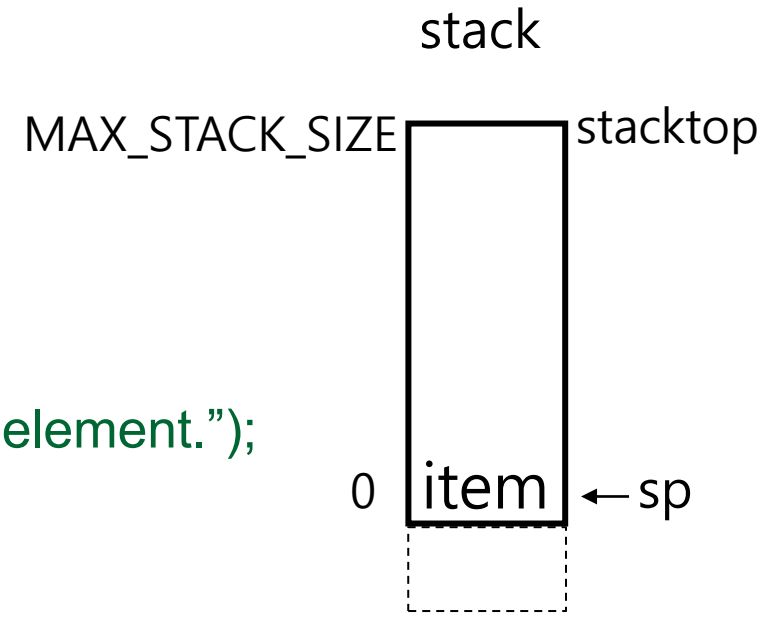
```
#define MAX_STACK_SIZE        100       /* maximum stack size */
  int stack[MAX_STACK_SIZE] ;             /* integer stack */

  struct stack {                          /* 구조체 배열로 생성한 스택 */
            int key;
            /* other fields */
  };
  typedef structure stack STACK;
  STACK stack[MAX_STACK_SIZE];

  int sp = -1;
```

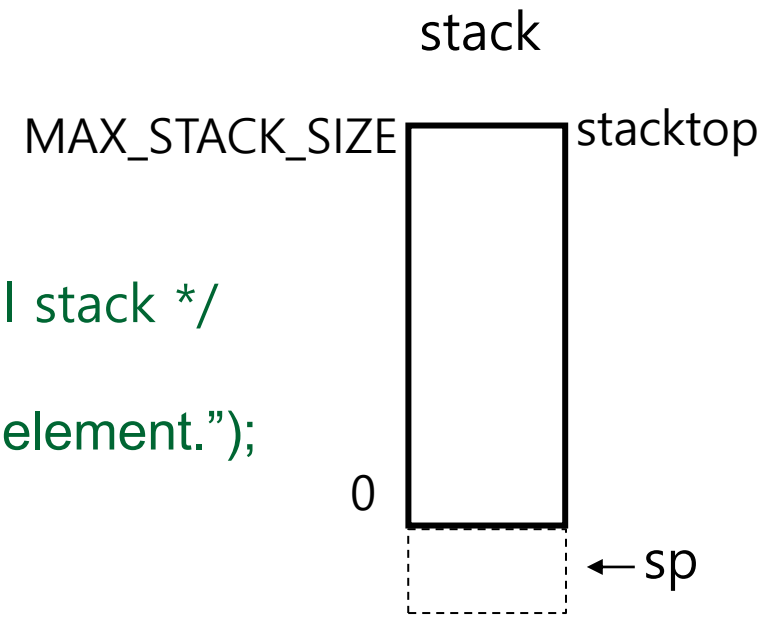
- **push(item)**

```
void push(int item)
{
 /* add an item to the global stack */
     if (isFull())  {
          printf("Stack is full, cannot add element.");
          exit(STACK_FULL);
     }
     stack[++sp] = item;
}
```



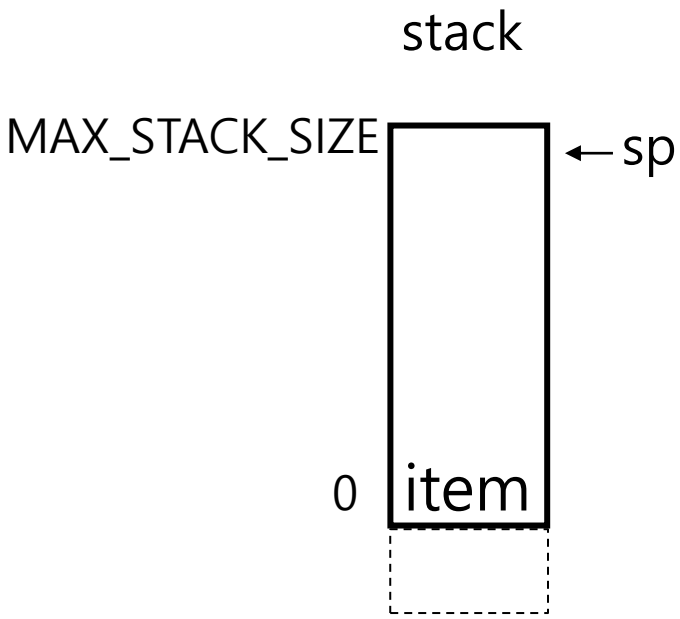
- **pop()**

```
int pop()
{
 /* return the top item from the global stack */
     if (isEmpty())  {
          printf("Stack is full, cannot add element.");
          exit(STACK_EMPTY);
     }
     return stack[sp--];
}
```



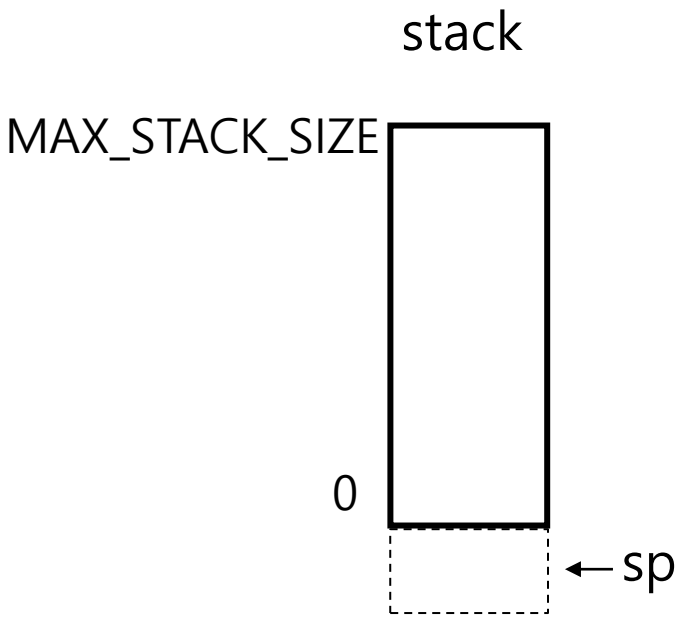
- **isFull()**

```
int isFull()
{
 /* check sp reaches stacktop */
      if (sp >= MAX_STACK_SIZE-1)  {
            return TRUE;
      }
      else return FALSE ;
}
```



- **isEmpty()**

```
int isEmpty()
{
 /* check sp reaches stack bottom */
      if (sp == -1)  {
            return TRUE;
      }
      else return FALSE ;
}
```



```
int main()
{
        int e;
        push(20);
        push(40);
        push(60);
        printf(" Begin Stack Test ...₩n");
        while(!isempty()) {
                e = pop();
                printf("value = %d₩n", e);
        }
}
```