Array Stack

Insertion and deletion takes place at one end (TOP).

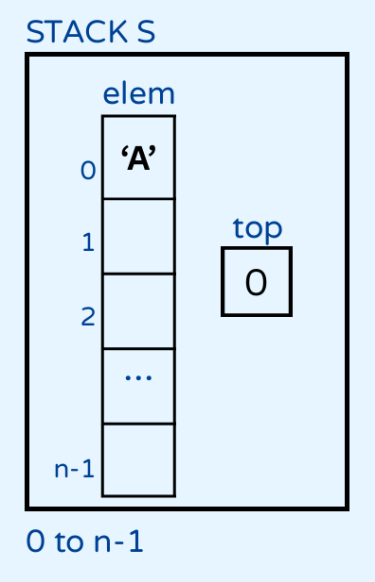
- FIRST IN LAST OUT
- LAST IN FIRST OUT

Important:

You are not allowed to directly access the content of the Stack from outside the operations. You must use the operations for all actions such as traversal and accessing.

Variation 1

Stack is a static array. TOP starts from **first position** and moves to the right. It is initialized to **-1** indicating empty.




```
#define MAX 10
typedef struct {
   int items[MAX];
   int top;
} Stack;
Stack S;
```

Operations	Checklist	Example

`Stack* initialize();`	☐ Allocate memory for a Stack structure ☐ Initialize the top of the stack to -1 to indicate it's empty ☐ Return the pointer to the stack	`Stack* S = initialize();`
`bool isFull(Stack* s);`	☐ top == MAX - 1	
`bool isEmpty(Stack* s);`	☐ top == -1	

`void push(Stack* s, int value);`	☐ Check if the stack is full
	☐ Increment the top
	☐ Place the new value at the current top position
		`Before:`
		`  items: [ 1, 3, 2, 5, ... ]`
		`  top: 3`
		
		`push(S, 4);`
		
		`After:`
		`  items: [ 1, 3, 2, 5, 4, ... ]`
		`  top: 4`

`int pop(Stack* s);`	☐ Check if the stack is empty
	☐ Get the value at the current top of the stack
	☐ Decrement the top
	☐ Return the retrieved value
		`Before:`
		`  items: [ 1, 3, 2, 5, ... ]`
		`  top: 3`
		
		`int value = pop(S);`
		
		`After:`
		`  items: [ 1, 3, 2, 5, ... ]`
		`  top: 2`

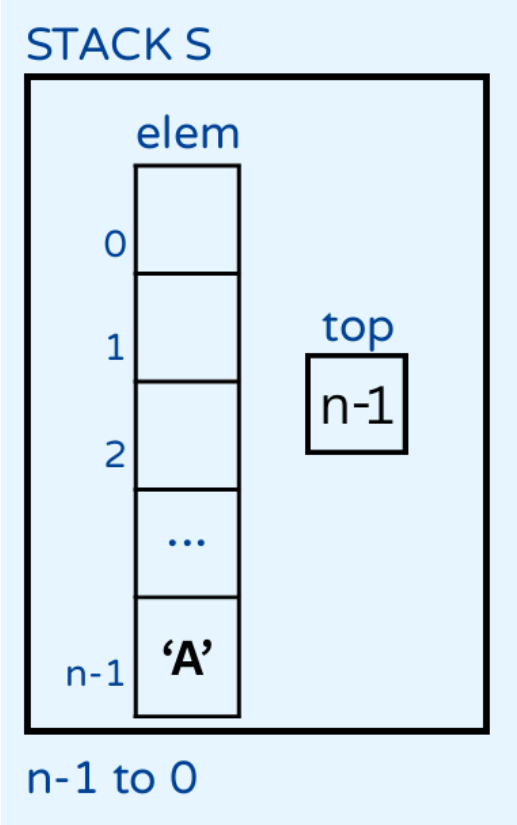
`int peek(Stack* s);`	☐ Check if the stack is empty (top == -1) ☐ Return the value at the top of the stack	
`int top(Stack *s);`	☐ Return the value of top	
`void display(Stack* s);`	☐ Check if the stack is empty (top == -1) ☐ Iterate from the top of the stack down to the bottom (index 0) ☐ Print each element	

Note:

For most operations, it is also common to return a **boolean value** representing whether the operation is successful or not.

Variation 2

Stack is a static array. TOP starts from **last position** and moves to the left. It is initialized as **MAX** indicating empty.




```
#define MAX 10
typedef struct {
    int items[MAX];
    int top;
} Stack;
Stack S;
```