```c
#include<stdio.h>
#include<stdlib.h>

struct stack{
    int size ;
    int top;
    int * arr;
};

int isEmpty(struct stack* ptr){
    if(ptr->top == -1){
            return 1;
        }
        else{
            return 0;
        }
}

int isFull(struct stack* ptr){
    if(ptr->top == ptr->size - 1){
        return 1;
    }
    else{
        return 0;
    }
}

void push(struct stack* ptr, int val){
    if(isFull(ptr)){
        printf("Stack Overflow! Cannot push %d to the stack\n", va
    }
    else{
        ptr->top++;
        ptr->arr[ptr->top] = val;
    }
}

int pop(struct stack* ptr){
    if(isEmpty(ptr)){
        printf("Stack Underflow! Cannot pop from the stack\n");
        return -1;
    }
    else{
        int val = ptr->arr[ptr->top];
        ptr->top--;
        return val;
    }
}

int peek(struct stack* sp, int i){
    int arrayInd = sp->top -i + 1;
    if(arrayInd < 0){
        printf("Not a valid position for the stack\n");
        return -1;
    }
    else{
        return sp->arr[arrayInd];
    }
}

int stackTop(struct stack* sp){
    return sp->arr[sp->top];
}

int stackBottom(struct stack* sp){
    return sp->arr[0];
}

int main(){
    // struct stack s;
    // s.size = 80;
    // s.top = -1;
    // s.arr = (int *) malloc(s.size * sizeof(int));

    struct stack *sp = (struct stack *) malloc(sizeof(struct stac
    sp->size = 10;
    sp->top = -1;
    sp->arr = (int *) malloc(sp->size * sizeof(int));
    printf("Stack has been created successfully\n");

    push(sp, 1);
    push(sp, 23);
    push(sp, 99);
    push(sp, 75);
    push(sp, 3);
    push(sp, 64);
    push(sp, 57);
    push(sp, 46);
    push(sp, 89);
    push(sp, 6); // ---> Pushed 10 values
    // push(sp, 46); // Stack Overflow since the size of the st
    printf("After pushing, Full: %d\n", isFull(sp));
    printf("After pushing, Empty: %d\n", isEmpty(sp));

    printf("Popped %d from the stack\n", pop(sp)); // --> Last in
    printf("Popped %d from the stack\n", pop(sp)); // --> Last in
    printf("Popped %d from the stack\n", pop(sp)); // --> Last in

    // Printing values from the stack
    for (int j = 1; j <= sp->top + 1; j++)
    {
        printf("The value at position %d is %d\n", j, peek(sp, j)
    }

    printf("The top most value of this stack is %d\n", stackTop(sp
    printf("The bottom most value of this stack is %d\n", stackBot

    return 0;
}

```