**DSA Assignment 1**

[**https://github.com/PrathamAsrani/DSA\_C/blob/master/assignment\_stack.c**](#_top)

#include <stdio.h>

#include <stdlib.h>

struct Stack

{

    int \*arr;

    int top, size;

};

/\*Stack Implimentation Function\*/

void create(struct Stack \*s);

int isFull(struct Stack \*s);

int isEmpty(struct Stack s);

int push(struct Stack \*s);

int pop(struct Stack \*s);

int peek(struct Stack \*s);

/\*Stack Implimentation Function\*/

int main()

{

    struct Stack stack;

    create(&stack);

    push(&stack);

    int top\_val = peek(&stack);

    printf("Peek : %d\n", top\_val);

    push(&stack);

    int top\_value = peek(&stack);

    printf("Peek : %d\n", top\_value);

    return 0;

}

void create(struct Stack \*s)

{

    s->top = -1;

    printf("Enter the size of 'Stack' : ");

    scanf("%d", &(s->size));

    s->arr = (int\*)malloc(s->size \* sizeof(int));

}

int isFull(struct Stack \*s)

{

    if (s->top == s->size - 1)

    {

        int increase;

        printf("Press '1' for increase memory allocation, else press any other number\n");

        scanf("%d", &increase);

        int newSize = s->size+(s->size/2);

        printf("newSize : %d\n", (newSize));

        s->size = newSize;

        if(increase == 1){

            s->arr = realloc(s->arr, newSize\*sizeof(int));

            return 0;

        }

        else{

            return 1;

        }

    }

    else

    {

        return 0;

    }

}

int isEmpty(struct Stack s)

{

    if (s.top == -1)

    {

        return 1;

    }

    else

    {

        return 0;

    }

}

int push(struct Stack \*s)

{

    if (isFull(s))

    {

        printf("Stack is full\n");

    }

    else

    {

        int data;

        printf("Enter element : \n");

        for(int i = 0; i < s->size; i++){

            scanf("%d", &data);

            s->top++;

            \*(s->arr+s->top) = data;

        }

    }

}

int pop(struct Stack \*s)

{

    if (isEmpty(\*s))

    {

        printf("Stack is empty\n");

    }

    else

    {

        int top = s->arr[top];

        s->top--;

        return top;

    }

}

int peek(struct Stack \*s)

{

    if (isEmpty(\*s))

    {

        printf("Stack is empty\n");

    }

    else

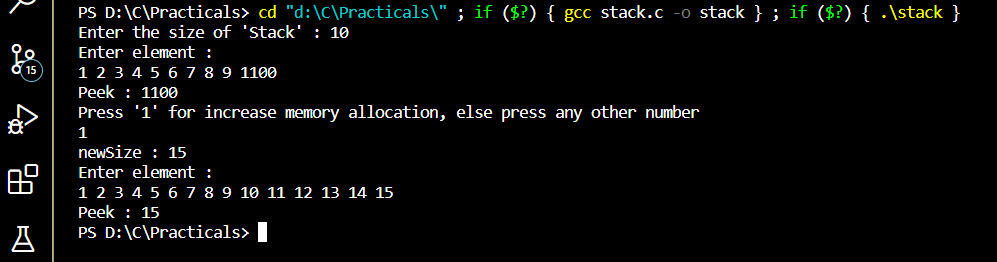
    {

        return \*(s->arr+s->top);

    }

}

**Output:**

****

**Result : Hence we successfully Modify the isfull function such that I'd the stack becomes full resize it by increasing the capacity 1.5 times the current capacity.**