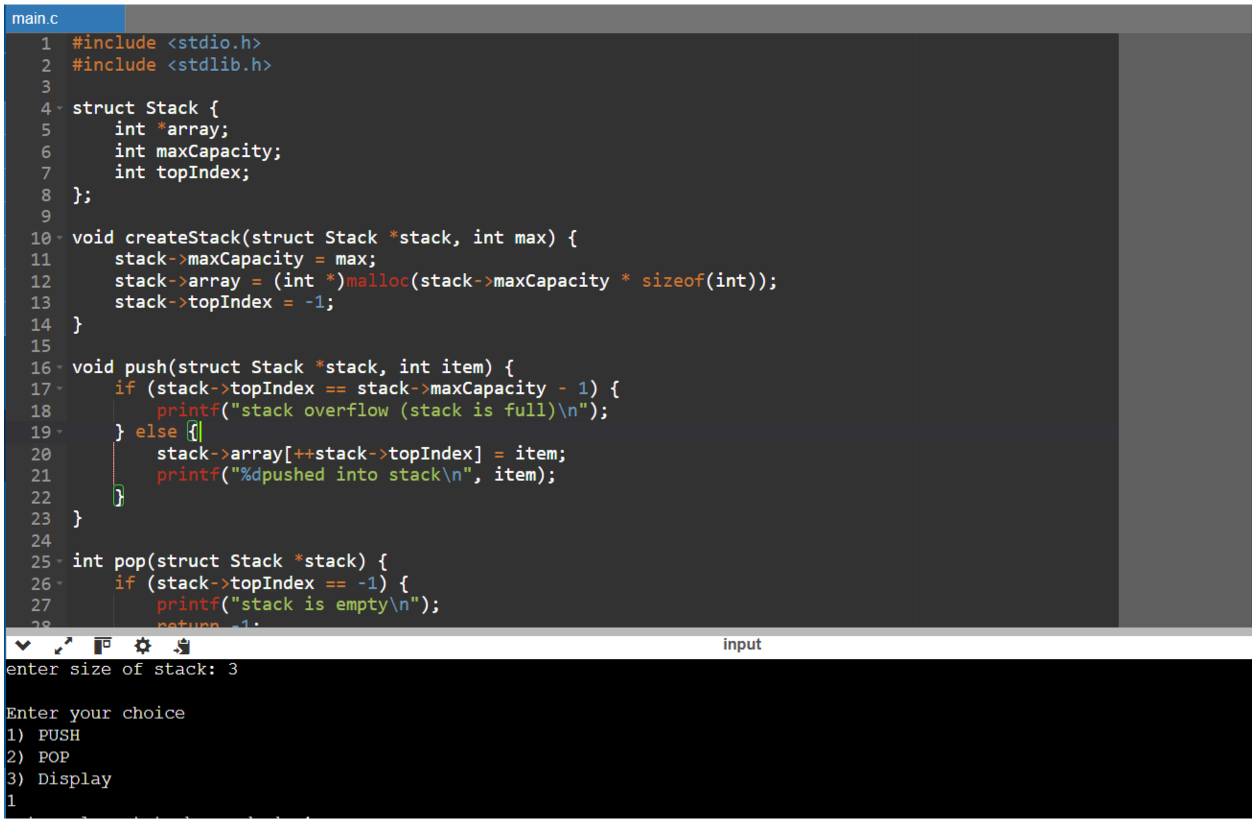
Name: Mithlesh Yeole
Class: B3-B3
Roll no.: 59

DS LAB Practical 2(A)

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

struct Stack {
    int *array;
    int maxCapacity;
    int topIndex;
};

void createStack(struct Stack *stack, int max) {
    stack->maxCapacity = max;
    stack->array = (int *)malloc(stack->maxCapacity * sizeof(int));
    stack->topIndex = -1;
}

void push(struct Stack *stack, int item) {
    if (stack->topIndex == stack->maxCapacity - 1) {
        printf("stack overflow (stack is full)\n");
    } else {
        stack->array[++stack->topIndex] = item;
        printf("%dpushed into stack\n", item);
    }
}

int pop(struct Stack *stack) {
    if (stack->topIndex == -1) {
        printf("stack is empty\n");
        return -1;
    } else {
        return stack->array[stack->topIndex--];
    }
}

void display(struct Stack *stack) {
    printf("stack elements: ");
    for (int i = stack->topIndex; i >= 0; i--) {
        printf("%d ", stack->array[i]);
    }
    printf("\n");
}

int main() {
    struct Stack mystack;
    int stacksize, element, option;

    printf("enter size of stack: ");
    scanf("%d", &stacksize);
    createStack(&mystack, stacksize);

    while (1) {
        printf("\nEnter your choice\n1) PUSH\n2) POP\n3) Display\n");
        scanf("%d", &option);

        switch (option) {
            case 1:
                printf("enter element to be pushed: ");
                scanf("%d", &element);
                push(&mystack, element);
                break;

            case 2:
                element = pop(&mystack);
                if (element != -1) {
                    printf("popped element:%d\n", element);
                }
                break;

            case 3:
                display(&mystack);
                break;
        }
    }

    return 0;
}
```



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

struct Stack {
    int *array;
    int maxCapacity;
    int topIndex;
};

void createStack(struct Stack *stack, int max) {
    stack->maxCapacity = max;
    stack->array = (int *)malloc(stack->maxCapacity * sizeof(int));
    stack->topIndex = -1;
}

void push(struct Stack *stack, int item) {
    if (stack->topIndex == stack->maxCapacity - 1) {
        printf("stack overflow (stack is full)\n");
    } else {
        stack->array[++stack->topIndex] = item;
        printf("%dpushed into stack\n", item);
    }
}

int pop(struct Stack *stack) {
    if (stack->topIndex == -1) {
        printf("stack is empty\n");
        return -1;
```

```
enter size of stack: 3

Enter your choice
1) PUSH
2) POP
3) Display
1
```