

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;  
}
```

```

main.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct Stack {
5      int *array;
6      int maxCapacity;
7      int topIndex;
8  };
9
10 void createStack(struct Stack *stack, int max) {
11     stack->maxCapacity = max;
12     stack->array = (int *)malloc(stack->maxCapacity * sizeof(int));
13     stack->topIndex = -1;
14 }
15
16 void push(struct Stack *stack, int item) {
17     if (stack->topIndex == stack->maxCapacity - 1) {
18         printf("stack overflow (stack is full)\n");
19     } else {
20         stack->array[++stack->topIndex] = item;
21         printf("%d pushed into stack\n", item);
22     }
23 }
24
25 int pop(struct Stack *stack) {
26     if (stack->topIndex == -1) {
27         printf("stack is empty\n");
28         return -1;

```

input

enter size of stack: 3

Enter your choice

- 1) PUSH
- 2) POP
- 3) Display

1