EUROPEAN UNIVERSITY OF LEFKE
Faculty of Engineering
Department of Software Engineering

COMP 217

DATA STRUCTURES

Lab Work No. 3

Prepared by Abdelrahman Mohamed Radwan Mostafa

Student Number : 21140036

Submitted to Dr. Ferhun Yorgancıoğlu

Code :

```c
#include <stdio.h>
#include <string.h>
#define SIZE 10
typedef struct {
char items[SIZE];
int top;
} stack;
void init_stack(stack * s){
s->top = -1;
}
int is_empty(stack *s){
return s->top == -1;
}
int is_full(stack *s)
{
return s->top == SIZE-1;
}
char top_of_stack(stack *s){
return is_empty(s) ? '\0' : s->items[s->top];
}
void push(stack *s, char c){
if(is_full(s)) {
printf("Overflow: stack is full!\n");
return;
}
s->top++;
s->items[s->top] = c;
}
char pop(stack *s){
if(is_empty(s)) {
printf("Underflow: stack is empty!\n");
return '\0';
}
char temp = s->items[s->top];
s->top--;
return temp;
}
int is_balanced(char * exp, stack * s){
```

```
    for(size_t i = 0; exp[i] != '\0'; i++){
        if(exp[i] == '(')
        push(s,exp[i]);
        if(exp[i] == ')')
        pop(s);
    }
    return is_empty(s);
}
void str_rev(char * str){
    stack s;
    init_stack(&s);
    int len=strlen(str);
    for (int i = 0; i < len; i++)
    push(&s, str[i]);
    while (!is_empty(&s))
    printf("%c",pop(&s));
}
int main(){
    stack s, min, max;
    init_stack(&s);
    init_stack(&min);
    init_stack(&max);
    char exp[SIZE], str[SIZE];
    printf("Enter the expression to check if its brackets are
balanced : ");
    scanf("%s",exp);
    if(is_balanced(&exp,&s))
    printf("Your expression is balanced\n");
    else printf("Your expression isn\'t balanced\n");
    printf("Enter a string to reverse : ");
    scanf("%s", str);
    str_rev(&str);
    printf("\n");
    int i = 0, x, arr[SIZE];
    printf("Enter numbers to get the max and mininum or enter 0 to
stop and get the results.\n");
    do{
        printf("Enter a number : ");
        scanf("%d", &x);
```

```
        arr[i] = x;
        if(is_empty(&min)) push(&min, arr[i]);
        else if(arr[i] < top_of_stack(&min)) push(&min, arr[i]);
        if(is_empty(&max)) push(&max, arr[i]);
        else if(arr[i] > top_of_stack(&max)) push(&max, arr[i]);
        i++;
    } while (x != 0);
    printf("Minimum element in array: %d\nMaximum element in array:
%d\n", top_of_stack(&min), top_of_stack(&max));
    return 0;
}
```

Result :