

# **Shri G.S. Institute of technology and Science**

(An Autonomous Institute, Established in 1952)

## **LABORATORY ASSIGNMENT**

**CT10212: Data Structure**



**Department of Information Technology and Application**

**2024-26**

**MCA I YEAR**

**Semester – I**

**SUBMITTED TO:**

**Mr. Deepesh Agarwal  
Ms. Parul Saxena**

**SUBMITTED BY:**

**SHIV ARORA**

# Assignment 2



# **Shri G.S Institute of Technology & Science**

## **Data Structure**

### **Assignment 2 – INDEX**

Sr. No.	Program	P. No.	Remarks
1	Write the algorithm for push and pop operation.	1	
2	Write a C program for push pop and display the stack elements using function.	1-4	
3	Write a program to show the ADT of stack using structure in C.	4-7	
4	Write a program to reverse string using stack.	7-9	
5	Write a program to convert infix into postfix using stack.	9-10	

Que1) Write the algorithm for push and pop operation.

Ans1)

### Push Operation Algorithm

1. Check for Overflow: (top == MAX - 1), display an error message and exit.
2. Input an Item.
3. Increment the Top: top++
4. Insert the Element: Assign the new element to stack[top].
5. Exit

### Pop Operation Algorithm

1. Check for Underflow: (top == -1), display an error message and exit.
2. Retrieve the Element: Store the value of stack[top] in a temporary variable.
3. Decrement the Top: top--
4. Return the Popped Value: Return the value stored in the temporary variable.
5. Exit

Que2) Write a C program for push pop and display the stack elements using function.

Ans2)

```
#include <stdio.h>
#include <stdbool.h>
#define MAX 1000

int stack[MAX];
int top = -1;
bool push(int value) {
    if (top == MAX - 1) {
        printf("Stack overflow\n");
        return false;
    }
    stack[++top] = value;
    return true;
}
int pop() {
```

```
if (top == -1) {
    printf("Stack underflow\n");
    return -1;
}
return stack[top--];
}

void display() {
    if (top == -1) {
        printf("Stack is empty\n");
        return;
    }
    printf("Stack elements: ");
    printf("{");
    for (int i = top; i >= 0; i--) {
        printf("%d ", stack[i]);
    }
    printf("}\n");
}

void main() {
    printf("SHIV ARORA\n");
    int choice, value;
    while (true) {
        printf("\nChoose an operation:\n");
        printf("1. Push\t2. Pop\t3. Display 4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                scanf("%d", &value);
                push(value);
                break;
            case 2:
```

```
    value = pop();
    if (value != -1) {
        printf("Popped: %d\n", value);
    }
    break;
case 3:
    display();
    break;
case 4:
    return 0;
default:
    printf("Invalid choice. Please try again.\n");
    break;
}
}
}
```

OUTPUT:

```
SHIV ARORA

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
34

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
56

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
56

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
45

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
78

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 2
Popped: 78

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 3
Stack elements: {45 56 56 34 }

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 4
PS D:\C Assignments\Run C Programs> |
```

Que3) Write a program to show the ADT of stack using structure in C.

Ans3)

```
#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>
```

```
#define MAX 1000

typedef struct Stack {
    int arr[MAX];
    int top;
} Stack;

void initStack(Stack* stack) {
    stack->top = -1;
}

bool push(Stack* stack, int value) {
    if (stack->top == MAX - 1) {
        printf("Stack overflow\n");
        return false;
    }
    stack->arr[++stack->top] = value;
    return true;
}

int pop(Stack* stack) {
    if (stack->top == -1) {
        printf("Stack underflow\n");
        return -1;
    }
    return stack->arr[stack->top--];
}

void display(Stack* stack) {
    if (stack->top == -1) {
        printf("Stack is empty\n");
        return;
    }
    printf("{ ");
    for (int i = stack->top; i >= 0; i--) {
```



```

        printf("%d ", stack->arr[i]);
    }
    printf("{}\n");
}

int main() {
    printf("SHIV ARORA");

    Stack stack;
    initStack(&stack);
    int choice, value;
    while (true) {
        printf("\nChoose an operation:\n");
        printf("1. Push\t2. Pop\t3. Display\t4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter value to push: ");
                scanf("%d", &value);
                push(&stack, value);
                break;
            case 2:
                value = pop(&stack);
                if (value != -1) {
                    printf("Popped: %d\n", value);
                }
                break;
            case 3:
                display(&stack);
                break;
            case 4:
                return 0;
            default:

```

```

        printf("Invalid choice. Please try again.\n");
        break;
    }
}
return 0;
}

```

OUTPUT:

```

SHIV ARORA
Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
Enter value to push: 34

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
Enter value to push: 234

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
Enter value to push: 23

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
Enter value to push: 243

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 1
Enter value to push: 67

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 3
{ 67 243 23 234 34 }

Choose an operation:
1. Push 2. Pop 3. Display 4. Exit
Enter your choice: 3
{ 67 243 23 234 34 }

```

Que4) Write a program to reverse string using stack.

Ans4)

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

```
#include <string.h>

#define MAX 100

typedef struct Stack {
    char arr[MAX];
    int top;
} Stack;

void initStack(Stack* s) {s->top = -1;}

int isFull(Stack* s) {return s->top == MAX - 1;}

int isEmpty(Stack* s) {return s->top == -1;}

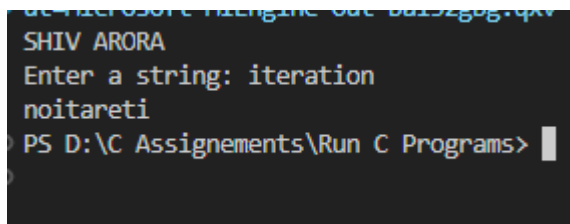
void push(Stack* s, char ch) {
    if (!isFull(s)) s->arr[++s->top] = ch;
}

char pop(Stack* s) {
    return !isEmpty(s) ? s->arr[s->top--] : '\0';
}

void reverseString(char* str) {
    Stack s; initStack(&s);
    for (int i = 0; str[i]; i++) push(&s, str[i]);
    for (int i = 0; str[i]; i++) str[i] = pop(&s);
}

int main() {
    printf("SHIV ARORA\n");
    char str[MAX];
    printf("Enter a string: ");
    fgets(str, MAX, stdin);
    str[strcspn(str, "\n")] = 0;
    reverseString(str);
    printf("%s\n", str);
    return 0;
}
```

OUTPUT:



```
SHIV ARORA
Enter a string: iteration
noitareti
PS D:\C Assignments\Run C Programs>
```

Que5) Write a program to convert infix into postfix using stack.

Ans5)

```
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include <string.h>
#define MAX 1000

typedef struct Stack {char arr[MAX]; int top;} Stack;

void push(Stack* s, char ch) {s->arr[++s->top] = ch;}

char pop(Stack* s) {return s->top == -1 ? '\0' : s->arr[s->top--];}

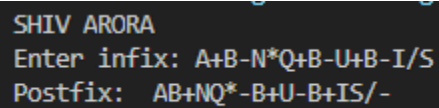
char peek(Stack* s) {return s->top == -1 ? '\0' : s->arr[s->top];}

int precedence(char op) {return op == '+' || op == '-' ? 1 : op == '*' || op == '/' ? 2 : 0;}

int main() {
    printf("SHIV ARORA\n");
    Stack s; s.top = -1;
    char infix[MAX], postfix[MAX];
    printf("Enter infix: ");
    fgets(infix, MAX, stdin);
    int j = 0;
    for (int i = 0; infix[i]; i++) {
        if (isalnum(infix[i])) postfix[j++] = infix[i];
        else if (infix[i] == '(') push(&s, infix[i]);
        else if (infix[i] == ')') {
            while (peek(&s) != '(') postfix[j++] = pop(&s);
            pop(&s);
        }
    }
    postfix[j] = '\0';
    printf("Postfix: %s\n", postfix);
}
```

```
    } else {  
        while (s.top != -1 && precedence(peek(&s)) >= precedence(infix[i]))  
            postfix[j++] = pop(&s);  
        push(&s, infix[i]);  
    }  
}  
while (s.top != -1) postfix[j++] = pop(&s);  
postfix[j] = '\\0';  
printf("Postfix: %s\\n", postfix);  
return 0;  
}
```

OUTPUT:



```
SHIV ARORA  
Enter infix: A+B-N*Q+U-B-I/S  
Postfix: AB+NQ*-B+U-B+IS/-
```