Data Structure

ASSIGNMENT –I

Name – Aaditya Raj Singh

Enrollment No – AU23E1002

Date of Submission

12/JAN/2024

Contents

Q.1 Write a program to perform following operations on stack.

a. **Create functions for push and pop operations of stack.**

```c
#include <stdio.h>

int top = -1, arr[5], item, ch;
void push()
{
    if (top >= 5)
    {
        printf("Stack is full\n");
    }
    else
    {
        top++;
        printf("Enter Element into stack: ");
        scanf("%d", &item);
        arr[top] = item;
    }
}

void pop()
{
    if (top == -1)
    {
        printf("Stack is empty\n");
    }
    else
    {
        printf("Popped Element: %d\n", arr[top]);
        top--;
    }
}

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

int main()
{
    do
    {
        printf("\n 1. PUSH\t 2. POP\t 3. DISPLAY\t 4. EXIT ");
        printf("\n Enter Your choice: ");
        scanf("%d", &ch);

        switch (ch)
        {
        case 1:
            push();
            break;
        case 2:
            pop();
            break;
        case 3:
            display();
            break;
        case 4:
            return 0;
        default:
            printf("Wrong value!!!\n");
            break;
        }
    } while (ch != 4);

    return 0;
}
```

Output

```
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>q1A

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 1
Enter Element into stack: 25

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 1
Enter Element into stack: 52

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 3
Stack elements: 25 52

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 2
Popped Element: 52

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 2
Popped Element: 25

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 3
Stack is empty

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 2
Stack is empty

 1. PUSH         2. POP  3. DISPLAY      4. EXIT
 Enter Your choice: 4
```

b. Write a function to convert an infix expression to postfix expression. Pass a one dimensional character array P to the function as input (infix exp) and return character array Q (postfix exp). Test your program for following input P : (A – (B / C) * D + E) * F % G

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

#define MAX_SIZE 100

struct Stack
{
    char items[MAX_SIZE];
    int top;
};

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

int is_empty(struct Stack *stack)
{
    return stack->top == -1;
}

int is_full(struct Stack *stack)
{
    return stack->top == MAX_SIZE - 1;
}

void push(struct Stack *stack, char item)
{
    if (is_full(stack))
    {
        printf("Stack is full\n");
        exit(EXIT_FAILURE);
    }
    stack->items[++stack->top] = item;
}

char pop(struct Stack *stack)
{
    if (is_empty(stack))
    {
        printf("Stack is empty\n");
        exit(EXIT_FAILURE);
    }
    return stack->items[stack->top--];
}

char peek(struct Stack *stack)
{
    if (is_empty(stack))
    {
        printf("Stack is empty\n");
        exit(EXIT_FAILURE);
    }
    return stack->items[stack->top];
}

int is_operator(char ch)
{
    return (ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '%' || ch == '^');
}

int precedence(char ch)
{
    if (ch == '^')
        return 1;
    if (ch == '+' || ch == '-')
        return 2;
    else if (ch == '*' || ch == '/' || ch == '%')
        return 3;
    else
        return 0;
}

int is_alphanumeric(char ch)
{
    return ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') || (ch >= '0' && ch <= '9'));
}

```

```c

char *infix_to_postfix(char *infix)
{
    struct Stack stack;
    initialize(&stack);

    int len = strlen(infix);
    char *postfix = (char *)malloc((len + 1) * sizeof(char));
    int p_index = 0;

    for (int i = 0; i < len; i++)
    {
        char ch = infix[i];

        if (is_alphanumeric(ch))
        {
            postfix[p_index++] = ch;
        }
        else if (ch == '(')
        {
            push(&stack, ch);
        }
        else if (ch == ')')
        {
            while (!is_empty(&stack) && peek(&stack) != '(')
            {
                postfix[p_index++] = pop(&stack);
            }
            if (!is_empty(&stack) && peek(&stack) == '(')
            {
                pop(&stack);
            }
        }
        else if (is_operator(ch))
        {
            while (!is_empty(&stack) && precedence(ch) <= precedence(peek(&stack)))
            {
                postfix[p_index++] = pop(&stack);
            }
            push(&stack, ch);
        }
    }

    while (!is_empty(&stack))
    {
        postfix[p_index++] = pop(&stack);
    }

    postfix[p_index] = '\0';

    return postfix;
}

int main()
{
    char iE[MAX_SIZE];
    printf("Enter Infix Expression: ");
    scanf("%s", iE);
    char *pE = infix_to_postfix(iE);

    printf("Infix Expression: %s\n", iE);
    printf("Postfix Expression: %s\n", pE);

    free(pE);

    return 0;
}
```

Output

```
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>q1B
Enter Infix Expression: (A-(B/C)*D+E)*F%G
Infix Expression: (A-(B/C)*D+E)*F%G
Postfix Expression: ABC/D*-E+F*G%
```

Q.2 List the various operations that can be performed on data structure.

Q.2 List the various operations that can be performed on data structure.

Ans

(1) Traversing :- Traversing a data structure means to visit the element stored in it. it visits data in a systematic manner.

(2) Searching :- Searching mean to find a particular element in a given data-structure it considered as successful when the required element is found.

(3) Insertion :- it is operation which we apply on all the data-structures It's means to add an element in a given data structure.

(4) Deletion : It is the operation which we apply on all the data-structure Deletion means to delete an element in the given data structure.

(5) Sort :- Sorting data in a particular order, we can take the help of many sorting algorithms to sort data in less time.

Q.3 What is abstract data type? What are all not concerned in an ADT?

Q. 3 What is Abstract data type? What are all not Concerned in an ADT?

Ans An abstract data type is a concept of a model of a data type that defines the behavior of a data type in terms & associated operations, without revealing the implementation details. The User only need to know what operation can be performed on the data type.

Some Key point about ADTS:-

→ ADT are a way of defining data type in terms of their behavior,

→ It's provide a level of abstraction that allows Users to work with data type without needing to know how they are implemented.

Q.4 Write the postfix form for the expression -A+B-C+D?

Q.4] Write the postfix form for the expression - A+B-C+D

0		(	
1	-	(-	
2	A	(-	A
3	+	(+	A-
4	B	(+	A-B
5	-	(-	A-B+
6	C	(-	A-B+C
7	+	(+	A-B+C-
8	D	(+	A-B+C-D
9	)		A-B+C-D+

Answer:- A-B+C-D+

Q.5 What are the postfix and prefix forms of the expression? A+B*(C-D)/(P-R)

Q.5 What are the postfix & prefix forms of the expression ? A+B*(C-D)/(P-R)

00	A	(	A
01	+	(+	A
02	B	(+	AB
03	*	(+*	AB
04	(	(+*(	AB
05	C	(+*(	ABC
06	-	(+*(-	ABC
07	D	(+*(-	ABCD
08	)	(+*	ABCD-
09	/	(+/	ABCD-*
10	(	(+/(	ABCD-*
11	P	(+/(	ABCD-*P
12	-	(+/(-	ABCD-*P
13	R	(+/(-	ABCD-*PR
14	)	(+/	ABCD-*PR-
15	)		ABCD-*PR-/+

Q.6 Write program to generate Fibonacci series. The user enter the limit of series.

```c
#include <stdio.h>

int main()
{
    int i, n;
    int t1 = 0, t2 = 1;
    int nt = t1 + t2;

    printf("ENTER THE NUMBER : ");
    scanf("%d", &n);

    printf("Fibonacci Series : %d, %d", t1, t2);

    for (i = 3; i <= n; i++)
    {
        t1 = t2;
        t2 = nt;
        nt = t1 + t2;
        printf(", %d", nt);
    }

    return 0;
}
```

Output

```
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>q6
ENTER THE NUMBER : 10
Fibonacci Series : 0, 1, 2, 3, 5, 8, 13, 21, 34, 55
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>
```

Q.7 Write program to calculate factorial of number.

Number being entered by user.

```c
#include <stdio.h>
int fact(int n)
{
    if (n >= 1)
        return n * fact(n - 1);
    else
        return 1;
}
void main()
{
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    printf("Factorial of %d = %d", n, fact(n));
}
```

Output

```
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>q7
Enter a number: 5
Factorial of 5 = 120
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>
```

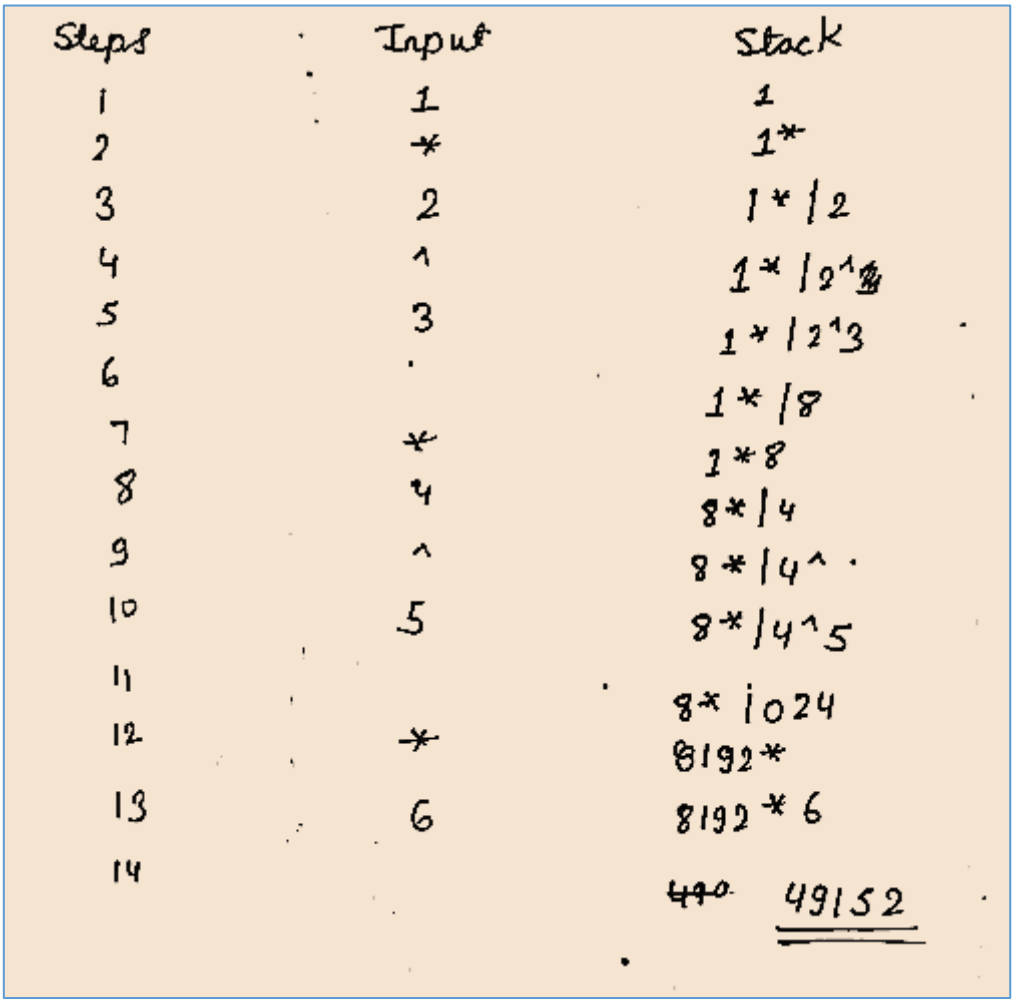
Q. 8 Write program to read two matrix and then print a matrix which is multiplication of these two matrix.

```c
#include <stdio.h>

int main() {
    int mat1[3][3], mat2[3][3], result[3][3];
    printf("Enter elements for Matrix 1 (3x3):\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("Enter element at position (%d, %d): ", i + 1, j + 1);
            scanf("%d", &mat1[i][j]);
        }
    }
    printf("Enter elements for Matrix 2 (3x3):\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("Enter element at position (%d, %d): ", i + 1, j + 1);
            scanf("%d", &mat2[i][j]);
        }
    }
    printf("\nMatrix 1:\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d\t", mat1[i][j]);
        }
        printf("\n");
    }
    printf("\nMatrix 2:\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d\t", mat2[i][j]);
        }
        printf("\n");
    }
    printf("\nMultiplication Result:\n");
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            result[i][j] = 0;
            for (int k = 0; k < 3; k++) {
                result[i][j] += mat1[i][k] * mat2[k][j];
            }
            printf("%d\t", result[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Output

```
C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>q8
Enter elements for Matrix 1 (3x3):
Enter element at position (1, 1): 1
Enter element at position (1, 2): 2
Enter element at position (1, 3): 3
Enter element at position (2, 1): 4
Enter element at position (2, 2): 5
Enter element at position (2, 3): 6
Enter element at position (3, 1): 7
Enter element at position (3, 2): 8
Enter element at position (3, 3): 9
Enter elements for Matrix 2 (3x3):
Enter element at position (1, 1): 7
Enter element at position (1, 2): 8
Enter element at position (1, 3): 9
Enter element at position (2, 1): 4
Enter element at position (2, 2): 5
Enter element at position (2, 3): 6
Enter element at position (3, 1): 1
Enter element at position (3, 2): 2
Enter element at position (3, 3): 3

Matrix 1:
1       2       3
4       5       6
7       8       9

Matrix 2:
7       8       9
4       5       6
1       2       3

Multiplication Result:
18      24      30
54      69      84
90      114     138

C:\Users\Aaditya Raj Singh\Desktop\2ndsem\2ndsem>
```

Q.9 The expression 1*2^3*4^5*6 will be evaluated as?



Steps	Input	Stack
1	1	1
2	*	1*
3	2	1*|2
4	^	1*|2^
5	3	1*|2^3
6		1*|8
7	*	1*8
8	4	8*|4
9	^	8*|4^
10	5	8*|4^5
11		8*|024
12	*	8192*
13	6	8192*6
14		~~440~~ 49152

Q.10 The postfix expression for the infix expression: A+B*(C+D)/(F+D)/F+D*E is?