**Batch: B-4 Roll No.:16010122221**

**Experiment / assignment / tutorial No.**

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of the Staff In-charge with date**

|  |
| --- |
| **Title:**  Implementation of Stack applications. |

**Objective:** To implement applications of stack

**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **CO** | **Outcome** |
| 1 | Explain the different data structures used in problem solving |

**Books/ Journals/ Websites referred:**

1. *Fundamentals Of Data Structures In C –* Ellis Horowitz, Satraj Sahni, Susan Anderson-Fred
2. *An Introduction to data structures with applications –* Jean Paul Tremblay,

Paul G. Sorenson

1. *Data Structures A Pseudo Approach with C –* Richard F. Gilberg & Behrouz A. Forouzan
2. *<https://www.cprogramming.com/tutorial/computersciencetheory/stack.html>*
3. *<https://www.geeksforgeeks.org/stack-data-structure-introduction-program/>*
4. *<https://www.thecrazyprogrammer.com/2013/12/c-program-for-array-representation-of-stack-push-pop-display.html>*

**Assigned Stack application**:

**Algorithm:**

* Start
* Initialize two stacks, say Stack and Backup-stack.
* Traverse the array of strings, Q, and perform the following operations:
* If Add data is selected then, push the character to Undo stack
* If Undo data is selected then, pop the top element from main stack and push it to backup stack.
* If Redo data is selected then, pop the top element of backup stack and push it into the main stack.
* If display data is selected then, print all the elements of the main stack.

**Example:**

* Perform Write P in the data. Therefore, the data contains only “P”.
* Perform Write A on the data. Therefore, the data contains “PA”.
* Perform Write R on the data. Therefore, the data contains “PAR”.
* Perform Undo on the data. Therefore, the data contains “PA”.
* Print the contents of the data, i.e. “PA”
* Perform Redo on the data. Therefore, the data contains “PAR”.
* Print the contents of the data, i.e. “PAR”

**Sourcecode:**

#include<stdio.h>

#include<ctype.h>

char stack[100];

int top = -1;

void push(char x)

{

    stack[++top] = x;

}

char pop()

{

    if(top == -1)

        return -1;

    else

        return stack[top--];

}

int priority(char x)

{

    if(x == '(')

        return 0;

    if(x == '+' || x == '-')

        return 1;

    if(x == '\*' || x == '/')

        return 2;

    return 0;

}

int main()

{

    char exp[100];

    char \*e, x;

    printf("Enter the expression : ");

    scanf("%s",exp);

    printf("\n");

    e = exp;

    while(\*e != '\0')

    {

        if(isalnum(\*e))

            printf("%c ",\*e);

        else if(\*e == '(')

            push(\*e);

        else if(\*e == ')')

        {

            while((x = pop()) != '(')

                printf("%c ", x);

        }

        else

        {

            while(priority(stack[top]) >= priority(\*e))

                printf("%c ",pop());

            push(\*e);

        }

        e++;

    }

    while(top != -1)

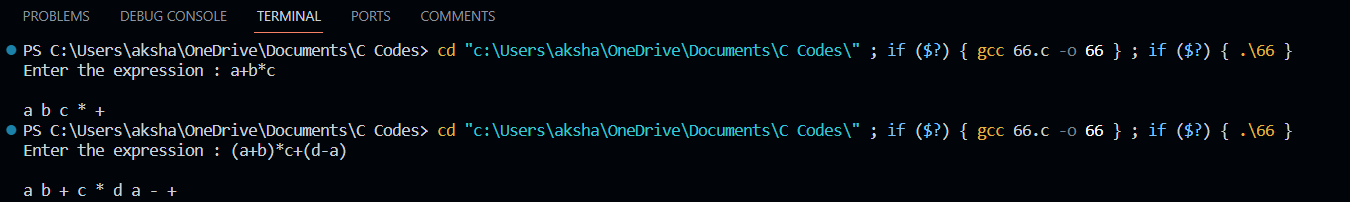
    {

        printf("%c ",pop());

    }return 0;

}

**Output Screenshots:**



**Conclusion:**Successfully implemented Undo-Redo operation implementation using static stack.