**Batch: B3 Roll No.: 121**

**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,

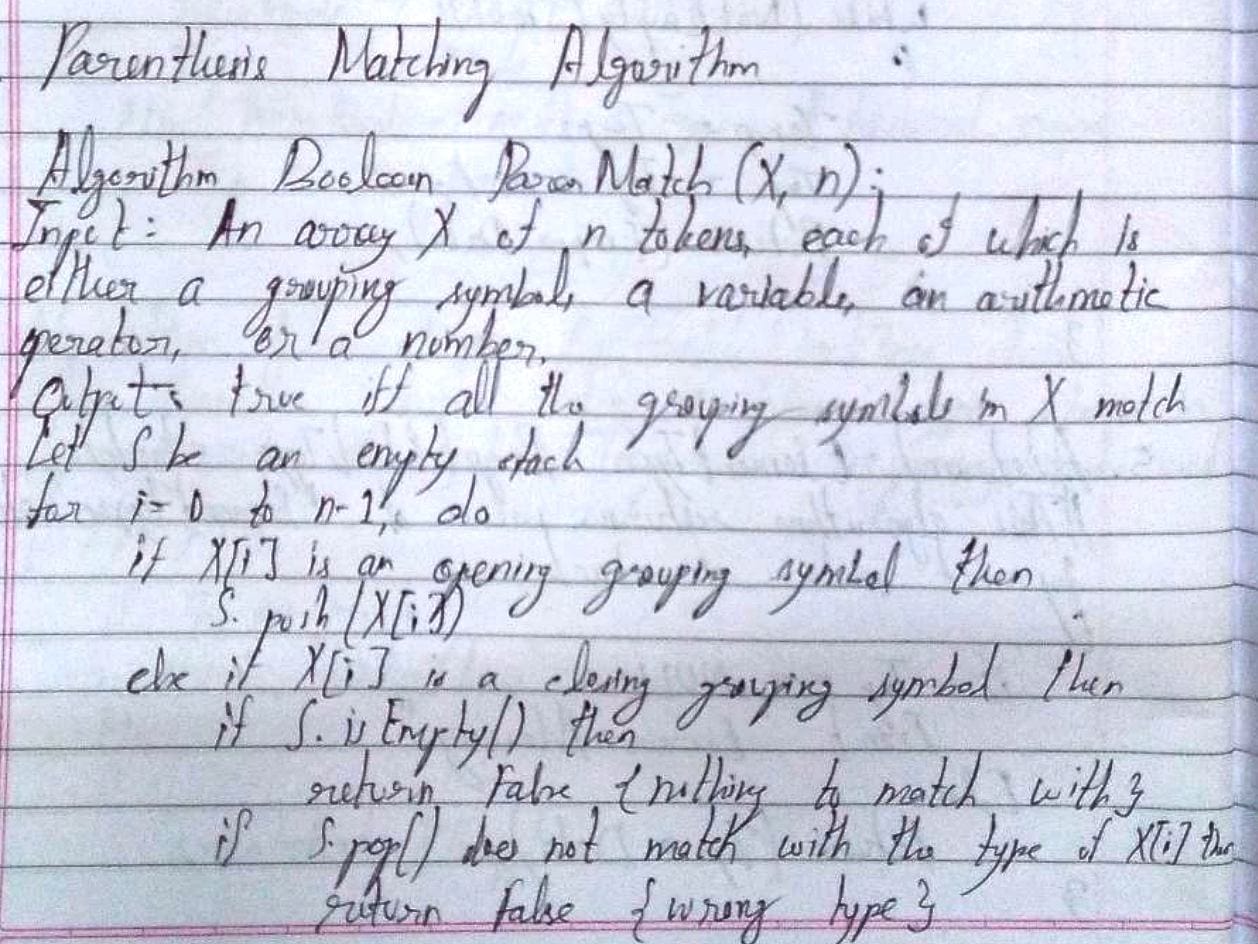
Paul G. Sorenson

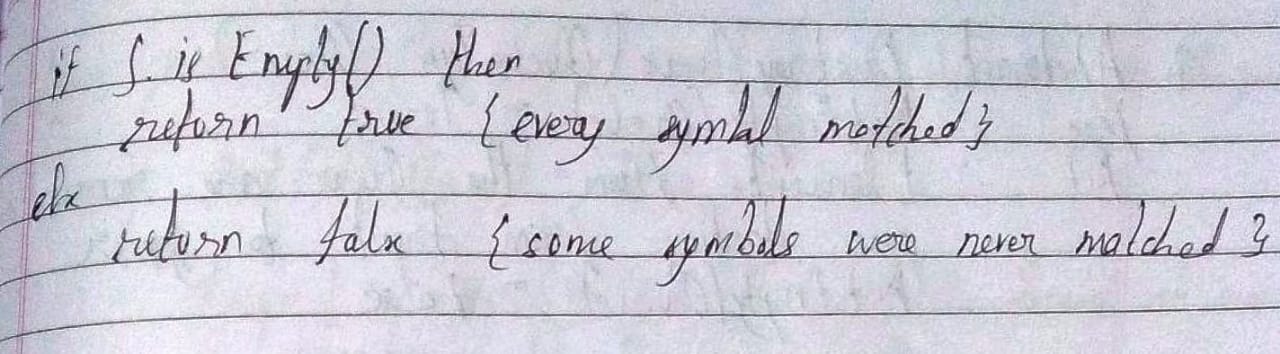
1. *Data Structures A Pseudo Approach with C –* Richard F. Gilberg & Behrouz A. Forouzan
2. [*https://www.cprogramming.com/tutorial/computersciencetheory/stack.html*](https://www.cprogramming.com/tutorial/computersciencetheory/stack.html)
3. [*https://www.geeksforgeeks.org/stack-data-structure-introduction-program/*](https://www.geeksforgeeks.org/stack-data-stru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

**Assigned Stack application**:

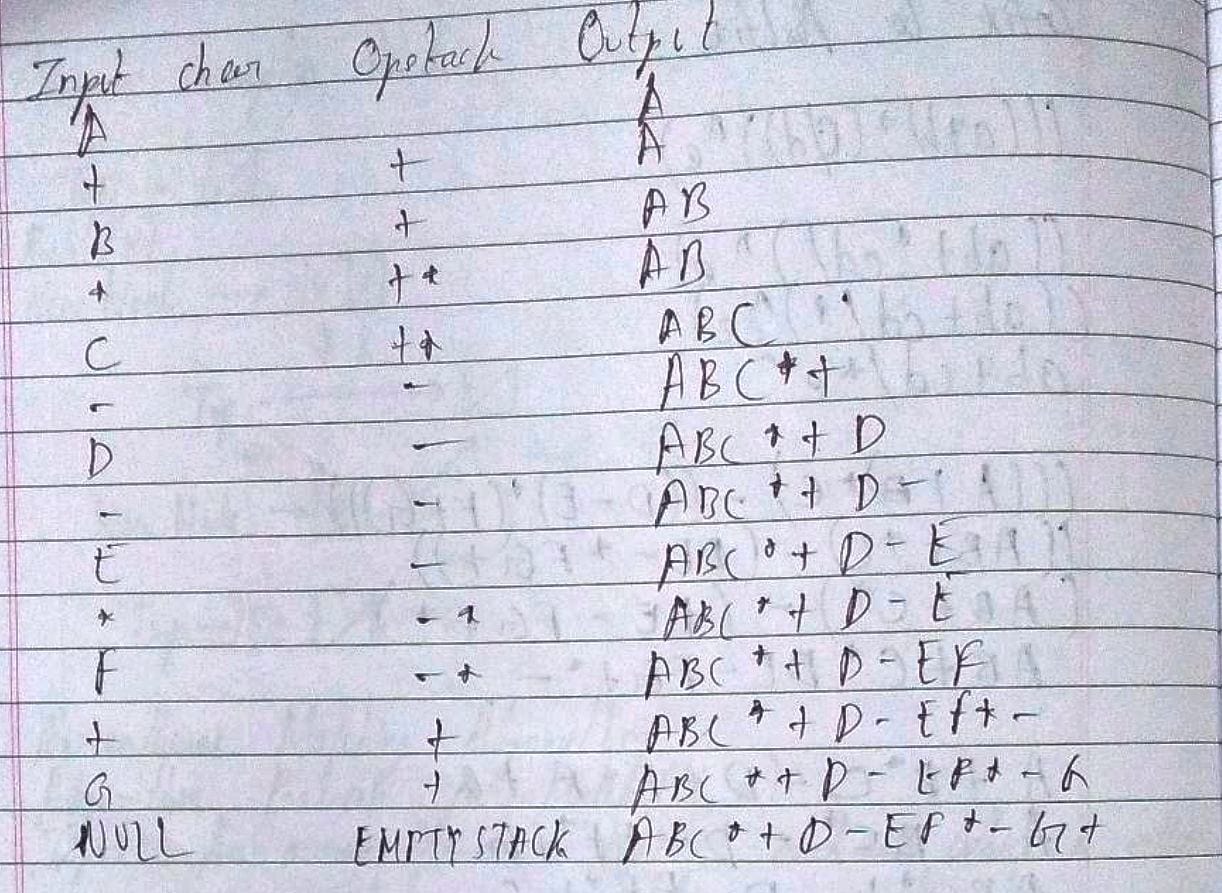
1. Parenthesis Matching Algorithm
2. Infix to Postfix Conversion

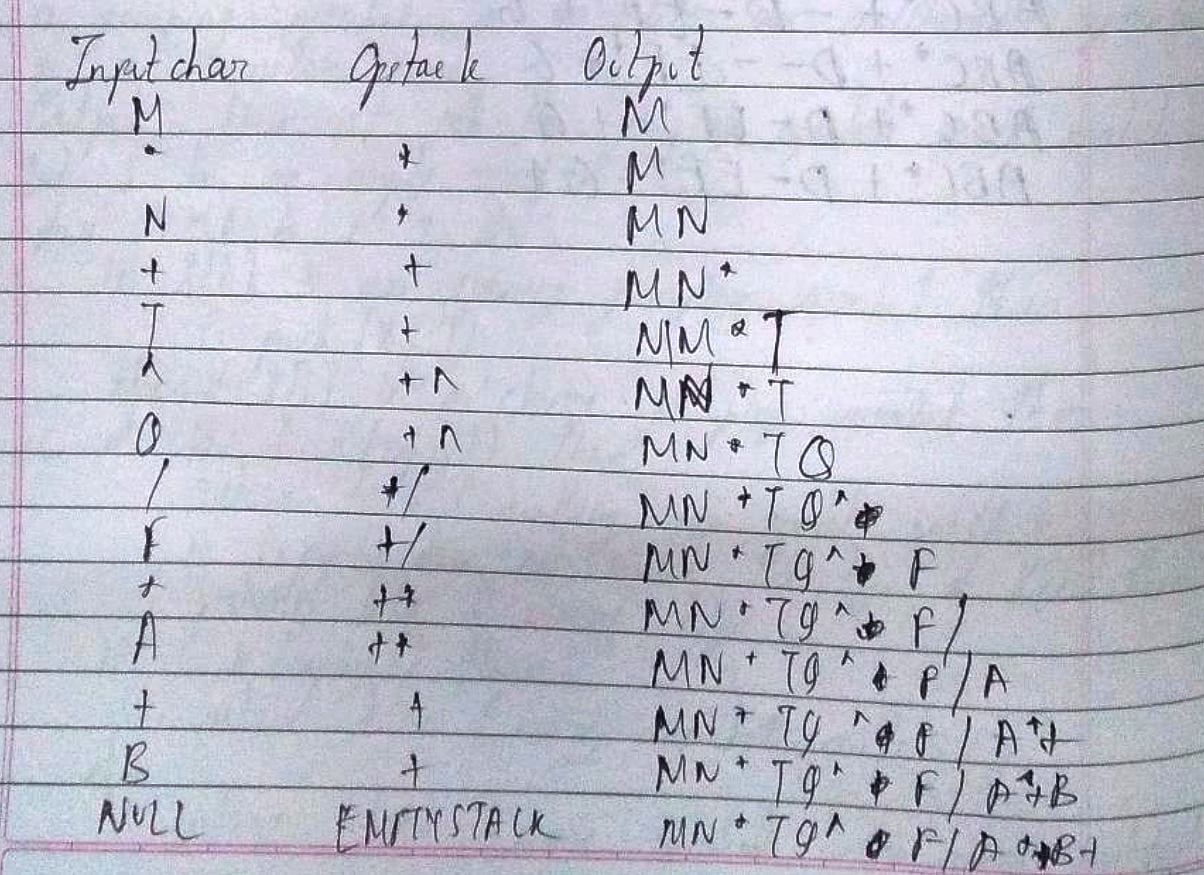
**Algorithm:**

****

****

**Examples:**

****

****

**Sourcecode:**

**Code 1:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define MAX 100

struct stack

{

char st[MAX];

int top;

}st\_v;

void push(char elem)

{

if(st\_v.top == (MAX - 1))

printf("\nStack Overflow!");

else

{

st\_v.top = st\_v.top + 1;

st\_v.st[st\_v.top] = elem;

}

}

void pop()

{

if(st\_v.top == -1)

printf("\nStack Underflow!");

else

st\_v.top = st\_v.top - 1;

}

void main()

{

char stk[MAX], ch;

int i = 0;

st\_v.top = -1;

printf("\nPARENTHESIS\tMATCHING\tALGORITHM");

do

{

printf("\nNote: The expression must be a maximum of 100 characters including whitespace ");

printf("\nEnter the expression: ");

scanf("%s", stk);

for(i = 0; i < strlen(stk); i++)

{

if(stk[i] == '(' || stk[i] == '[' || stk[i] == '{')

{

push(stk[i]);

continue;

}

else if(stk[i] == ')' || stk[i] == ']' || stk[i] == '}')

{

if(stk[i] == ')')

{

if(st\_v.st[st\_v.top] == '(')

{

pop();

}

else

{

printf("\nThe expression is unbalanced.");

break;

}

}

if(stk[i] == ']')

{

if(st\_v.st[st\_v.top] == '[')

{

pop();

}

else

{

printf("\nThe expression is unbalanced.");

break;

}

}

if(stk[i] == '}')

{

if(st\_v.st[st\_v.top] == '{')

{

pop();

}

else

{

printf("The expression is unbalanced.");

break;

}

}

}

}

if(st\_v.top == -1)

printf("\nThe expression is balanced.");

printf("\nDo you want to try for another expression?\nIn that case, enter 'Y', otherwise enter\nany other character: ");

scanf("%s", &ch);

}while(ch=='Y');

}

Code 2:

#include<stdio.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 elem)

{

if(elem == '(')

return 0;

else if(elem == '+' || elem == '-')

return 1;

else if(elem == '\*' || elem == '/')

return 2;

return 0;

}

int main()

{

char stk[100], ch;

char \*e, x;

printf("\nINFIX\tTO\tPOSTFIX\tALGORITHM");

do

{

printf("\nNote: The maximum number of characters in the expression\nincluding whitespace, cannot be greater than 100.");

printf("\nEnter the infix expression: ");

scanf("%s", stk);

printf("\n");

e = stk;

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());

printf("\nDo you want to try for another expression?\nIn that case, enter 'Y', otherwise enter\nany other character: ");

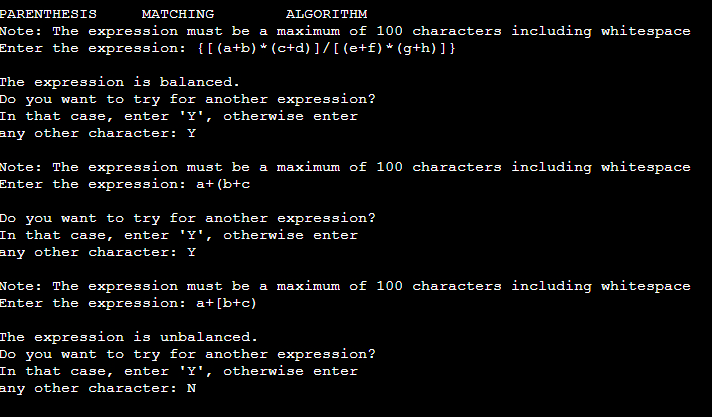
scanf("%s", &ch);

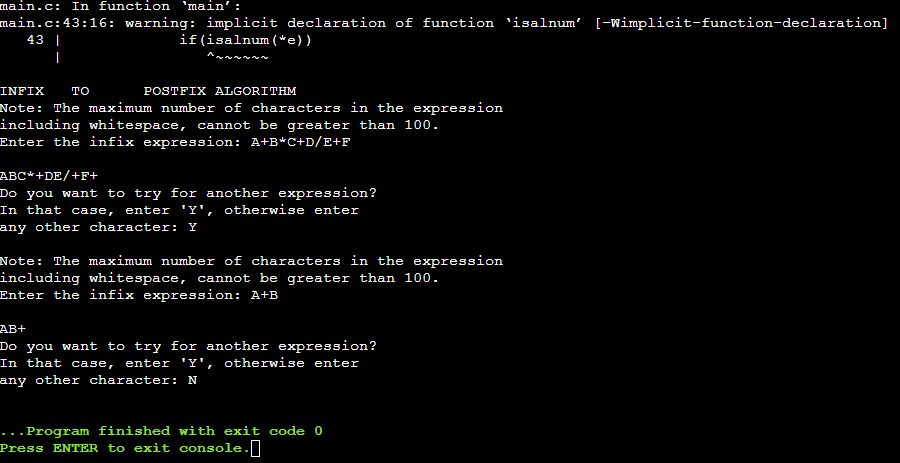
}while(ch=='Y');

return 0;

}

**Output Screenshots:**

****

****

**Conclusion:**

Thus, in this experiment, the applications of stack using Data Structures has been learnt and implemented.