DS LAB

Name: B RUCHITHA

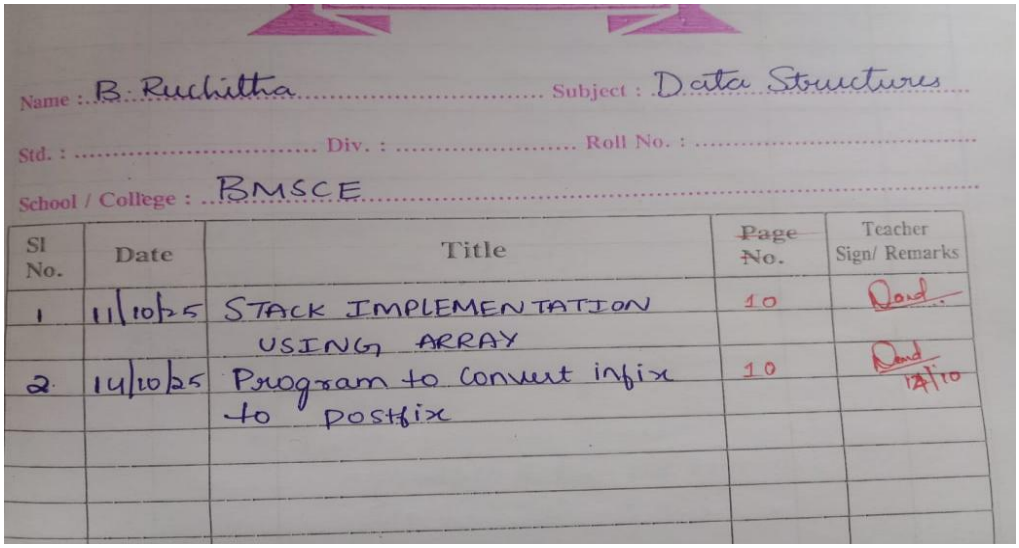
USN:DIP

Program 2: Infix To Postfix:

Output:

```
PS C:\Users\student\Desktop\BRuchitha_3rd_Dsec> ./a.exe
Enter the infix expresssion: (A*B)+C*D/E
The Postfix Expression: AB*CD*E/+
PS C:\Users\student\Desktop\BRuchitha_3rd_Dsec> ./a.exe
Enter the infix expresssion: 1+2*(3+4)
The Postfix Expression: 1234+*+
PS C:\Users\student\Desktop\BRuchitha_3rd_Dsec>
```

Observation:



Name : B. Ruchitha Subject : Data Structures

Std. : Div. : Roll No. :

School / College : BMSCE

Sl No.	Date	Title	Page No.	Teacher Sign/ Remarks
1	11/10/25	STACK IMPLEMENTATION USING ARRAY	10	Read.
2	14/10/25	Program to Convert infix to postfix	10	Read 14/10

Date :- 14/10/25

Program 3 :- Program to Convert Infix expression to postfix

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define N 100
char stack[N];
int top=-1;
void push(char x){
    if(top==N-1){
        Printf("Stack Overflow\n");
    }else{
        stack[++top]=x;
}
char
void pop(){
    if(top==-1){
        Printf("Stack Underflow\n");
    }else{
        return stack[top--];
}
int precendence(char a){
    if(a=='^'){
        return 3;
    }else if(a=='*' || a=='/' || a=='%'){
        return 2;
    }else if(a=='+' || a='-'){
        return 1;
    }else{
        return 0;
    }
}
int main(){
    char infix[N], postfix[N];
    int i; int k=0;
    char x;
```

```
Printf("Enter a Infix expression: ");
gets(infix);
for(i=0; i<strlen(infix); i++){
        char ch = infix[i];
        if(isalnum(ch)){
                Postfix[k++] = ch;
        }else if(ch=='('){
                Push(ch);
        }else if(ch==')'){
                while((x=pop())!='('){
                        Postfix[k++] = x;
                }
        }else{
                while(Precendence(Stack[top])>=
                        Precendence(ch)){
                        Postfix(k++) = pop();
                }
                Push(ch);
        }
}
while(top!=-1){
        Postfix[k++] = pop();
}
Postfix[k] = '\0';
Printf("Postfix Expression: %s", postfix);
return 0;
}.
```

Output :-

Enter the infix expression : (A * B) + C

The Postfix Expression : AB* C+

Good .. 14/10