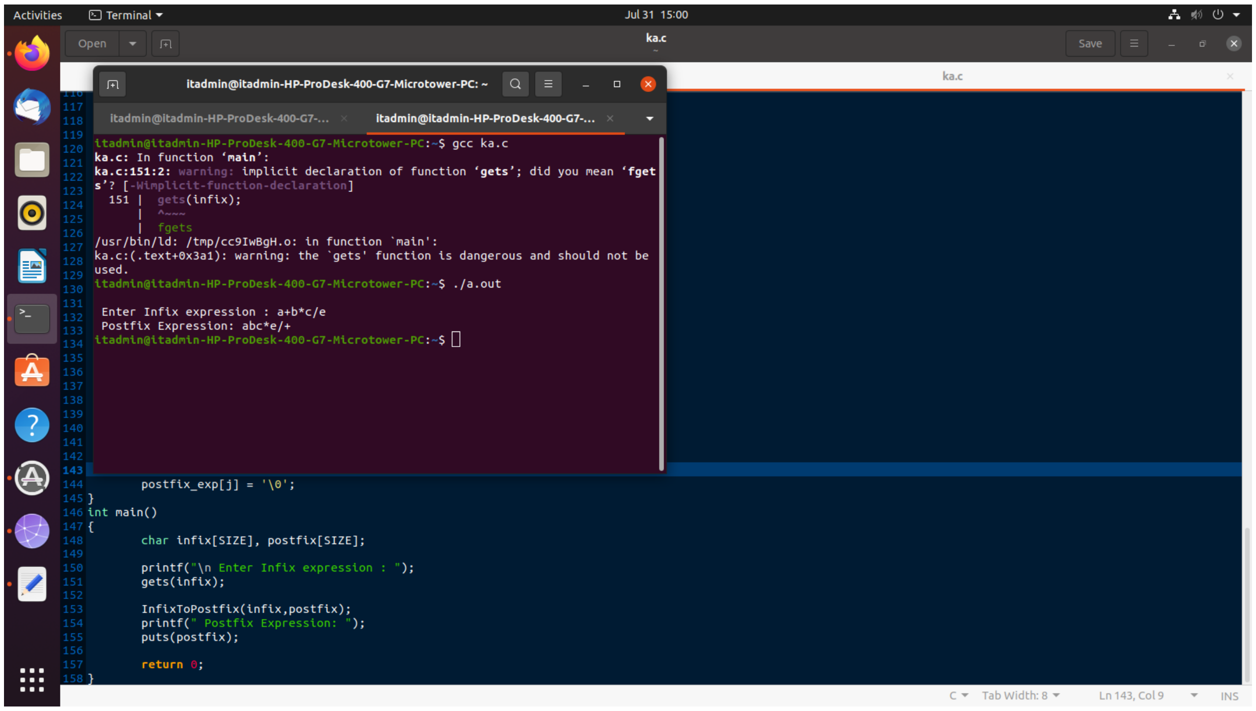
Karan shah
Roll number - 52

```
code-#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>

#define SIZE 100
char stack[SIZE];
int top = -1;

/* === define push operation === */
void push(char item)
{
   if(top >= SIZE-1)
   {
          printf("\n Stack Overflow.");
   }
   else
   {
          top = top+1;
          stack[top] = item;
   }
}

/* === define pop operation === */
char pop()
{
   char item ;

   if(top <0)
   {
          printf("stack under flow: invalid infix expression");
          getchar();
          /* underflow may occur for invalid expression */
          /* where ( and ) are not matched */
          exit(1);
   }
   else
   {
          item = stack[top];
          top = top-1;
          return(item);
```

```
    }
}


int is_operator(char symbol)
{
    if(symbol == '^' || symbol == '*' || symbol == '/' || symbol == '+' || symbol =='-')
    {
            return 1;
    }
    else
    {
    return 0;
    }
}


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

void InfixToPostfix(char infix_exp[], char postfix_exp[])
{
    int i, j;
    char item;
    char x;

    push('(');
    strcat(infix_exp,")");

    i=0;
    j=0;
    item=infix_exp[i];

    while(item != '\0')
    {
            if(item == '(')
            {
                    push(item);
            }
            else if( isdigit(item) || isalpha(item))
            {
                    postfix_exp[j] = item;
                    j++;
            }
            else if(is_operator(item) == 1)
            {
                    x=pop();
                    while(is_operator(x) == 1 && precedence(x)>= precedence(item))
                    {
                            postfix_exp[j] = x;
                            j++;
                            x = pop();
                    }
                    push(x);

                    push(item);
            }
            else if(item == ')')
            {
                    x = pop();
                    while(x != '(')
                    {
                            postfix_exp[j] = x;
                            j++;
                            x = pop();
                    }
            }
            else
            {
                    printf("\nInvalid infix Expression.\n");
                    getchar();
```

```
                        exit(1);
                }
                i++;


                item = infix_exp[i];
        }
        if(top>0)
        {
                printf("\nInvalid infix Expression.\n");
                getchar();
                exit(1);
        }

        postfix_exp[j] = '\0';
}
int main()
{
        char infix[SIZE], postfix[SIZE];

        printf("\n Enter Infix expression : ");
        gets(infix);

        InfixToPostfix(infix,postfix);
        printf(" Postfix Expression: ");
        puts(postfix);

        return 0;
}
```



```
itadmin@itadmin-HP-ProDesk-400-G7-Microtower-PC:~$ gcc ka.c
ka.c: In function ‘main’:
ka.c:151:2: warning: implicit declaration of function ‘gets’; did you mean ‘fget
s’? [-Wimplicit-function-declaration]
  151 |  gets(infix);
      |  ^~~~
      |  fgets
/usr/bin/ld: /tmp/cc9IwBgH.o: in function `main':
ka.c:(.text+0x3a1): warning: the `gets' function is dangerous and should not be
used.
itadmin@itadmin-HP-ProDesk-400-G7-Microtower-PC:~$ ./a.out

 Enter Infix expression : a+b*c/e
 Postfix Expression: abc*e/+
itadmin@itadmin-HP-ProDesk-400-G7-Microtower-PC:~$ 
```

```
144         postfix_exp[j] = '\0';
145 }
146 int main()
147 {
148         char infix[SIZE], postfix[SIZE];
149 
150         printf("\n Enter Infix expression : ");
151         gets(infix);
152 
153         InfixToPostfix(infix,postfix);
154         printf(" Postfix Expression: ");
155         puts(postfix);
156 
157         return 0;
158 }
```