```
#include <stdio.h>
#define SIZE 100

char stack[SIZE];
int top = -1;

void push(char ch)
{
if (top == SIZE-1)
  printf("Stack Overflow");
else
{
top++;
stack[top] = ch;
}
}

char pop()
{
char item;
if (top == -1)
  printf("Stack underflow");
else
{
item = stack[top];
top--;
return item;
}
}

int stackempty()
{
if (top == -1)
  return 1;
```

```
else
return 0;
}

char stacktop()
{
if (top == -1)
printf("Underflow");
else
return stack[top];
}

int priority (char ch)
{
switch (ch)
{
case '+':
case '-': return (1);
case '*':
case '/': return(2);
case '^': return(3);
default : return (0);
}
}
int main (int argc, char *argv)
{
char infix[100];
int i, item;
Printf("Enter a valid infix exp: ");
Scanf("%s - - - - ---------- %s");
for (int i=0; i < strlen(infix); i++)
{
```

```
if (infix[i]=='*' || infix[i]=='+' || infix[i]==
    || infix[i+1]=='*' || inf[i+]=='+' || infix[
    || infix[i]=='-' || infix[i]=='^' || infix[
    infix[i+1]=='-' || infix[i+1]=='^' || ifix
{
printf("INVALID EXP");
exit(1);
}
}

printf("Entered infix exp is: %s", infix);
printf("\n The generated Postfix Exp is: ");
i=0;
while (infix[i]!='\0')
{
switch (infix[i])
{
case '(': push (infix[i]);
          break;
case ')': while (item = pop()) != '(')
              printf("%c", item);
          break;
case '+':
case '-':
case '*':
case '/':
case '^':
    while (!stackempty() && priority(infix[i])
                <= priority(stack[
    {
    item = pop();
    printf("%c", item);
    }
    push (infix[i]);
    break;
```

```
        default : printf("%c", infix[i]);
                  break;
        }
    i++;
    }

while ( !stackempty() )
{
  char item;
  item = pop();
  printf("%c", item);
}
printf("\n");
return 0;
}
```