```
Write a program to simulate the working of stack using an array
with the following.
 al push
 8) bob
 c) Display
  The program should print appropriate message for stock overflow
 and, stack underflow.
 # deline sizes;
  int top = - 1, Stack [size];
  void push (int e)
         if (top == size)
               print(" stack overflow");
          else
               top = top+1;
                stack[top]=e;
                printf (" The insertion operation is complete");
            3
      }
  raid bob()
       if (top = = -1)
            printf(" stack underflow, stack is empty!);
         }
       else
          1 prints (" the deleted element: (: (d" stack (top));
             top = top-1;
     z
  void display co
              printf (" ctade: 1 empty");
```

```
dse
   {
       for (1=top; 1=0; 1
     >
   }
tothe mains
 mt value, choice
 Indix to postfix
    int index 6=0, pos=0, top=-1, length;
    char symbol, temp, infix [20], postfix [20], stack(20);
    bid
     void push (char symbol)
      گر
          top= top+1;
           Stack[top] = symbol;
       z
    char pop ()
      J
         char val;
         val = stack [top];
         top 0 -- ;
          return (gral);
        }
    int presidence (char symbol)
    £
       int p;
        Switch (symbol)
            case ' *1:
            case 11, $ 6= 5;
             cate - 1 :
```

```
Case 'c' = p = 0;
               break;
     case # : p=-1;
               breaks;
    return (p);
void intia to pattix ();
{
   legright = strlen ("infix);
     perh (4+1)
     while (length > index)
          Symbol = infix (index);
           Switch (symbol)
            {
               case '(": pash(symbol);
                          break;
               case ')' = temp = pop()
                       mp://e(femp/z(c))
                         postfix [poi] = temp;
                          POT ++ ,
                          temp = popci;
               cale it.
                cale'~.
                 case +1:
                 (ale 1/1:
                        while (peridance (stacke (top) = prelidance (symbol)
                          temp=popc);
                          postfix [post+] = temps
                           puh (1ymbod;
                     default : politica (politi) = symbol;
                     index ++s
```

```
while (top ro)
        portfix (port +) = temp;
out put ?
1. Stock:
Menu -
      1 such:
       2. 109
       3. display
       4. exit
        1 - antered 2 - anter a value
                              02
                          I hear kion openation is complete
        2. - ) ontorned 2 - deleted value 50
        3. - contrad 3 - grade is empty
        4. of it entered - exits the stack.
 a. Infix to postfix
  O enter a infix expression:
           a+6+ (c^d-1)^(ftg*h)-;
     postfix: abcdne-fgh ++ N++i-
  (1)
         intix: ((AFO)-C4(0/E))+F
         politix: ABICDE 1 *- FI
```

```
#include<stdio.h>
 1
      #include<stdlib.h>
 2
 3
 4
      int size=5;
      int top=-1,stack[5];
 5
 6
      void push(int a)
 7
 8
          if(top==size)
 9
10
              printf("stack overflow\n");
11
12
          else{
13
              top=top+1;
14
              stack[top]=a;
15
              printf("insertion operation is complete\n");
16
17
18
19
      void pop()
20
21
          if (top==-1)
22
                                               I
23
              printf("stack is empty\n");
24
25
          else{
26
              top--;
27
28
29
30
     void display()
31
32
          if (top==-1)
33
          {
34
              printf("stack is empty\n");
35
36
          else{
37
```

```
- stack i.e / Williami
          CTOCI
38
              for(int i=top;i>=0;i--)
39
                  printf("%d\n", stack[i]);
40
41
42
43
44
45
      int main()
46
      {
          int value, choice, t=0;
47
          while(1)
48
          {
49
50
              printf("-
                                                           --\n");
                                 -----MENU-----
              printf("1.push\n 2.pop\n 3.display\n 4.exit\n");
51
              scanf("%d",&choice);
52
              switch (choice)
53
54
              case 1: printf("enter a value:\n");
55
                       scanf("%d",&value);
56
                       push(value);
57
                   break;
58
                                              I
59
              case 2: pop();
60
61
                  break;
62
              case 3:display();
63
                  break;
64
65
66
              case 4:exit(0);
                  break;
67
68
              default:printf("wrong input!\n");
69
70
                  break;
71
72
73
```

```
-----MENU-----
 1. push
 2.pop
 3.display
  4.exit
 1
 enter a value:
 50
 insertion operation is complete
 -----MENU-----
1.push
 2.pop
 3.display
 4.exit
3
50
1. push
 2.pop
 3. display
 4.exit
2
             --MENU--
1. push
 2.pop
 3.display
 4.exit
stack is empty
          ----MENU-----
1. push
 2.pop
 3.display
4.exit
4
Process returned 0 (0x0)
```

execution time : 20.880 s Press any key to continue.

```
3rd_sem > C learning > C infix.c > 分 infixtopostfix()
       #include<stdio.h>
  1
  2
       #include<stdlib.h>
       #include<string.h>
  3
  4
       int top=-1,pos=0;
  5
       char temp, stack[25], infix[25], postfix[25];
  6
  7
       void push(char s)
  8
  9
       {
            stack[++top]=s;
 10
 11
 12
       int precedence(char s)
 13
       {
 14
            switch(s)
 15
            {
 16
                 case '^':
 17
 18
                     return 3;
                 case '+':
 19
                 case
 20
                     return 1;
 21
                case
 22
                      '/':
                case
 23
                     return 2;
 24
                case '(':
  25
                     return 0;
  26
  27
  28
  29
       char pop()
  30
        {
  31
            char symb=stack[top];
  32
            top--;
  33
            return symb;
  34
  35
  36
       void infixtopostfix()
  37
```

```
AOTO THITYTOPOSTITY()
38
          int len=strlen(infix);
39
          int i=0;
40
          char symbol;
41
          while(i<len)
42
43
              symbol=infix[i];
44
45
              switch(symbol)
46
47
                   case '(':
48
                            push(symbol);
49
                            break;
50
51
                   case ')':
52
                       temp=pop();
53
                       while(temp!='(')
54
                       {
55
                            postfix[pos++]=temp;
56
                            temp=pop();
57
58
                       break;
59
60
                   case
61
                   case
62
                   case
63
                   case '/':
64
                                                                       I
                   case '^':
65
                       while(precedence(stack[top])>=precedence(symbol))
66
67
                           postfix[pos++]=pop();
68
69
                       push(symbol);
70
                       break;
71
                   default:postfix[pos++]=symbol;
72
73
```

Clearing

```
75
          while(top!=-1)
76
77
78
               postfix[pos++]=stack[top];
79
               top--;
80
81
          return;
82
83
84
      int main()
85
           printf("enter a infix problem:\n");
86
           scanf("%s",infix);
87
           infixtopostfix();
88
           printf("infix :%s\n",infix);
 89
           printf("postfix :%s\n",postfix);
 90
 91
 92
 93
```

```
enter a infix problem:
a+b(c^d-e)^(f+g*h)-i
infix :a+b(c^d-e)^(f+g*h)-i
postfix :abcd^e-fgh*+^+i-
Process returned 0 (0x0) exec
```