```
a) infix to postfix using operand +, -, +, /, (,)
    int index = 0 , pos = 0 , top = -1 , length;
   Char Symbol, temp, infix [20], postix [20], stack [20],
    Void infix to postbix ();
    Void push (charsymbol);
     than pop ();
      int pred (than symb);
    Void main ()
      discret;
      pf (" enter infix expression: \n').

S[ (" infix)
      infinto postfix ()
       Pf Linfix expression is)
       PL L postfix expression is)
     Void main & infinto postfix()
        length = strlen linfix);
push ((#));
         while (index < length)
         2 symbol - in fix [indev];
        Switch (symbol) {
          Care ((1: push (symbol);
          breake:
          proush ( symboth; temp = pop ();
           Break;
```

```
while (temp! = 'c')
  post fix [pos ++] = +emp;
 poss ++;
 lemp = pop();
break;
cau '+':
coul 1-1:
Case ( *):
case 11:
(are 'n': while (pred (Stack Etop ] >= pred (Symbol))
cours a
        temp= pop();
         postfix [postt ]=temp;
         push Csymbol)
  default : post fi x Cpos + + J=Symbol;
 index ++;
```

```
store
67
while (top > 0)
temp = pop ();
postfix [pos ++ ]=temp
void push (char symbol)
top = top + 1;
Stack[top] = Symbol;
Char symb;
Heturn Stack Ctop-- ].
int pred (charsymbol)
Switch (symbol)
Case ' 1'.
 case ' * ';
 case 1 1 :
case ' n':
  while I prece (Stack [top]) ) = p~ (symbol)){
```

```
temp = Pop ();
  postfix [pos++] = temp.
 push (symbol);
 break,
 defaut:
 post fix [post+]=symbol;
 index ++;
while (top> 0) }
temp = pop(); }
postfix [pos]=1101.
char pop () {
  char symb=stack [ to p];
  top - - 6
return Symb;
int pre (char symb) {
intp
Switch (Symbol) {
 care ' 1'
 p = 3 :/
  break;
  Case 1 $1.
  case X':
  p = 2;
   break;
```

```
# include < stdio.b>
# include < conio. b>
# include < string.h>
int index=0, pos=0, top=-1, length;
char Symbol, temp, infix [36], postfix [30], stack[30].
 Void infix To postbix ();
 Void push (than symbol);
  Char pop ();
  int precedence (Char Symb);
 int main () }
  print (" Enter infix expression: \n")
  Scant ( " " s", 4 infix);
 infix To postfix ();
  perint ("In infix expression: In y.s", infix);
  pount ("in post fix expression in its" postfir);
   Meturn O:
 Void infix To postfix () {
    length = str len (in fix)
    push = (+#1);
    while (index < length) {
       symbol = infix [index];
        switch (Symbol) {
     Case (1):
           push (symbol);
            break:
       Case ()':
           temp = pop ();
             while (temp 1=1(1) f
                postfrix [post+]=temp;
                  temp=pop();
```

```
break;
case (+1:
Case (-1.
case ( * ):
care (11:
care (1).
 while (precedence (stack [top]) >= procedence (symbol));
 temp= popes;
  postfix [pos+1] = temp;
 push Cymbol);
 becoalci
 defaut!
  pastfix [pos++] = terosp;
 postfix Cpos ] = 101.
void push (char Symbol ) {
   top=top+1;
    Stack [ top] = Symbol;
char pop () {
    char symb = Stack [top];
    top--
   hetion Symb:
```

```
int precedence (char symbol) {
int P:
 Switch (symbol) {
    care (A):
      p = 3;
      break;
     case ( * " :
     case 11:
     p= 2.
     break;
     case 41:
     case (-1:
     P=1;
     break;
     course ( c):
     p=0;
    break;
   Care 'H 'A
    p =/-1
     break;
  return P;
o Output
Venter infix expression:
a be-d+elf 1cg+h)
postbix code
abe d+eb/ght/+.
```