1. **Algorithm**:

**Step 1** : Scan the Infix Expression from left to right.

* **Step 2** : If the scanned character is an operand, append it with final Infix to Postfix string.
* **Step 3** : Else,
  + **Step 3.1** : If the precedence order of the scanned(incoming) operator is greater than the precedence order of the operator in the stack (or the stack is empty or the stack contains a ‘(‘ or ‘[‘ or ‘{‘), push it on stack.
* **Step 3.2** : Else, Pop all the operators from the stack which are greater than or equal to in precedence than that of the scanned operator. After doing that Push the scanned operator to the stack. (If you encounter parenthesis while popping then stop there and push the scanned operator in the stack.)
* **Step 4** : If the scanned character is an ‘(‘ or ‘[‘ or ‘{‘, push it to the stack.
* **Step 5** : If the scanned character is an ‘)’or ‘]’ or ‘}’, pop the stack and and output it until a ‘(‘ or ‘[‘ or ‘{‘ respectively is encountered, and discard both the parenthesis.
* **Step 6** : Repeat steps 2-6 until infix expression is scanned.
* **Step 7** : Print the output
* **Step 8** : Pop and output from the stack until it is not empty.