Assignments

Question-1

The stock span problem is a financial problem where we have a series of n daily price quotes for a stock and we need to calculate span of stock's price for all n days. The span Si of the stock's price on a given day i is defined as the maximum number of consecutive days just before the given day, for which the price of the stock on the current day is less than or equal to its price on the given day.

Sample Input
{100, 80, 60, 70, 60, 75, 85}
An array of 7 days prices is given as

Sample Output
{1, 1, 1, 2, 1, 4, 6}
The span values for corresponding 7 days

Sample Input
{10, 4, 5, 90, 120, 80}
An array of 6 days prices is given as

Sample Output
{1, 1, 2, 4, 5, 1}
The span values for corresponding 6 days

Question-2

Write a Program to convert infix expression to prefix

Sample input
A+B-C

Sample output
-+ABC

Algorithm

Let, X is an arithmetic expression written in infix notation. This algorithm finds the equivalent postfix expression Y.

1. Push "(" onto Stack, and add ")" to the end of X.
2. Scan X from left to right and repeat Step 3 to 6 for each element of X until the Stack is empty.
3. If an operand is encountered, add it to Y.
4. If a left parenthesis is encountered, push it onto Stack.
5. If an operator is encountered ,then:
   Repeatedly pop from Stack and add to Y each operator (on the top of Stack) which has the same precedence as or higher precedence than operator.
   Add operator to Stack.
6. If a right parenthesis is encountered ,then:
   Repeatedly pop from Stack and add to Y each operator (on the top of Stack) until a left parenthesis is encountered.
   Remove the left Parenthesis.
   [End of If]
7. END.

An Example
Infix Expression: A+ (B*C-(D/E^F)*G)*H, where ^ is an exponential operator.

Symbol	Scanned	STACK	Postfix Expression	Description
1.		(		Start
2.	A	(	A	
3.	+	(+	A	
4.	(	(+(	A	
5.	B	(+(	AB	
6.	*	(+(*	AB	
7.	C	(+(*	ABC	
8.	-	(+(-	ABC*	'*' is at higher precedence than '-'
9.	(	(+(-(	ABC*	
10.	D	(+(-(	ABC*D	
11.	/	(+(-(/	ABC*D	
12.	E	(+(-(/	ABC*DE	
13.	^	(+(-(/^	ABC*DE	
14.	F	(+(-(/^	ABC*DEF	
15.	)	(+(-	ABC*DEF^/	Pop from top on Stack , that's why '^' Come first
16.	*	(+(-*	ABC*DEF^/	
17.	G	(+(-*	ABC*DEF^/G	
18.	)	(+	ABC*DEF^/G*-	Pop from top on Stack , that's why '^' Come first
19.	*	(+*	ABC*DEF^/G*-	
20.	H	(+*	ABC*DEF^/G*-H	
21.	)	Empty	ABC*DEF^/G*-H*+	END

Resultant Postfix Expression: ABC*DEF^/G*-H*+