Infix to Postfix

Ex: A+B*C

Step	Input	Stack	Output
1.	A+B*C	-	
2.	+B*C	-	A
3.	B*C	+	A
4.	*C	+	AB
5.	C	+*	AB
6.		+*	ABC
7.		+	ABC*
8.			ABC*+

Final Post fix Expression : ABC*+

---

(2) (A+B)*(C-D)

Step	Input	Stack	Output.
1.	(A+B)*(C-D)	-	-
2.	(A+B)*(C-D)	(	-
3.	+B)*(C-D)	(	A
4.	B)*(C-D)	(+	A
5.	)*(C-D)	(+	A

6.	*(C-D)		AB+
7.	(C-D)	*	AB+
8.	(-D)	*(	AB+
9.	-D)	*(-	AB+C.
10.	D)	*(-	AB+C.
11.	)	*	AB+CD-
12.	-	-	AB+CD-*

Postfix to infix

① AB+C*

Step	Postfix	stack:
1.	AB+C*	[ ]
2.	B+C*	[A]
3.	+C*	[AB]
4.	C*	[(A+B)]
5.	*	[(A+B).C]
6.	—	[(A+B)*C]

Infix expression : [(A+B)*C]

Balancing parenthesis

① [A+B) * (C-D)

Step	Read char	stack.
1.	(	[(]
2.	A	[(]
3.	+	[(]
4.	B	[(]
5.	)	[ ]
6.	*	[ ]
7.	(	[(]
8.	C	[(]
9.	-	[(]
10.	D	[(]
11.	)	[ ]

stack is Empty

② ({A+(B*C)-D})

Step	Read char	Stack.
1.	{	[{]
2.	A	[{]
3.	+	[{]
4.	(	[{,(]
5.	B	[{,(]
6.	*	[{,(]
7.	C	[{,(]
8.	)	[{,(]
9.	-	[{]
10.	D	[{]
11.	}	[ ]
12.	)	[)]

↓

Stack not empty.