GIT Department of Computer Engineering
CSE 222/505 - Spring 2020
Homework #4 Part 1 Report

Murat YILDIZ
1801042004

Q1:

Convert the infix expressions given below to prefix and postfix, then evaluate them. Show your work (how each token is processed and content of the stack afterwards) step by step.

i) A + ((B – C * D) / E) + F – G / H

ii) ! (A && ! ((B < C) || (C > D))) || (C < E)

i)

Infix to postfix

A B C D * – E / + F + G H / –

CONVERSION

Expression	Action	Stack	Effect on postfix
A + ((B – C * D) / E) + F – G / H	It is an operand, Append A to postfix		A
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , Push +	+	A
A + ((B – C * D) / E) + F – G / H	Push (	(+	A
A + ((B – C * D) / E) + F – G / H	Push (	((+	A
A + ((B – C * D) / E) + F – G / H	It is an operand, Append B to postfix	((+	A B
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , Push –	– ((+	A B
A + ((B – C * D) / E) + F – G / H	It is an operand, Append C to postfix	– ((+	A B C
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , Push *	* – (	A B C

		(+	
A + ((B – C * D) / E) + F – G / H	It is an operand, Append D to postfix	* – ((+	A B C D
A + ((B – C * D) / E) + F – G / H	Pop and append to postfix until precedence is equal or less than)	(+	A B C D * –
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , Push /	/ (+	A B C D * –
A + ((B – C * D) / E) + F – G / H	It is an operand, Append E to postfix	/ (+	A B C D * – E
A + ((B – C * D) / E) + F – G / H	Pop and append to postfix until precedence is equal or less than)	+	A B C D * – E /
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , append to postfix	+	A B C D * – E / +
A + ((B – C * D) / E) + F – G / H	It is an operand, Append F to postfix	+	A B C D * – E / + F
A + ((B – C * D) / E) + F – G / H	Pop and append to postfix until precedence is equal or less than – and Push –	–	A B C D * – E / + F +
A + ((B – C * D) / E) + F – G / H	It is an operand, Append G to postfix	–	A B C D * – E / + F + G
A + ((B – C * D) / E) + F – G / H	Pop and append to postfix until precedence is equal or less than / and Push /	/ –	A B C D * – E / + F + G

A + ((B – C * D) / E) + F – G / H	It is an operand, Append H to postfix	/ –	A B C D * – E / + F + G H
A + ((B – C * D) / E) + F – G / H	When it is ended Pop and append to postfix until stack is empty	–	A B C D * – E / + F + G H /
A + ((B – C * D) / E) + F – G / H	When it is ended Pop and append to postfix until stack is empty		A B C D * – E / + F + G H / –

i)

POSTFIX

EVALUATION

SCANNING LEFT TO RIGHT

A = 20, B =15, C = 2, D = 6, E = 3, F = 8, G = 16, H = 4

Expression	Action	Stack
A B C D * – E / + F + G H / –	Push A	20
A B C D * – E / + F + G H / –	Push B	15 20
A B C D * – E / + F + G H / –	Push C	2 15 20
A B C D * – E / + F + G H / –	Push D	6 2 15 20
A B C D * – E / + F + G H / –	Pop twice , Do math and push result to stack	12 15 20
A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	3 20
A B C D * – E / + F + G H / –	Push E	3 3 20

A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	1 20
A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	21
A B C D * – E / + F + G H / –	Push F	8 21
A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	29
A B C D * – E / + F + G H / –	Push G	16 29
A B C D * – E / + F + G H / –	Push H	4 16 29
A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	4 29
A B C D * – E / + F + G H / –	Pop twice, Do math and push result to stack	25

i)

Infix to prefix

+ A + / – B * C D E – F / G H

CONVERSION

SCANNING RIGHT TO LEFT

Expression	Action	Stack	Effect on prefix
A + ((B – C * D) / E) + F – G / H	It is an operand, Append H to prefix		H
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher , Push /	/	H

A + ((B – C * D) / E) + F – G / H	It is an operand, Append G to prefix	/ 	H G
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than – and Push –	– 	H G /
A + ((B – C * D) / E) + F – G / H	It is an operand, Append G to prefix	– 	H G / F
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than + and Push +	+ 	H G / F –
A + ((B – C * D) / E) + F – G / H	Push)	) +	H G / F –
A + ((B – C * D) / E) + F – G / H	It is an operand, Append E to prefix	) +	H G / F – E
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher ,Push /	/) +	H G / F – E
A + ((B – C * D) / E) + F – G / H	Push)	) /) +	H G / F – E
A + ((B – C * D) / E) + F – G / H	It is an operand, Append D to prefix	) /) +	H G / F – E D
A + ((B – C * D) / E) + F – G / H	It is an operator, İts precedence is higher ,Push *	*) /) +	H G / F – E D
A + ((B – C * D) / E) + F – G / H	It is an operand, Append C to prefix	*) /) +	H G / F – E D C
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than – and Push –	-) /) +	H G / F – E D C *

A + ((B – C * D) / E) + F – G / H	It is an operand, Append B to prefix	-) /) +	H G / F – E D C * B
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than (	/) +	H G / F – E D C * B -
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than (	+	H G / F – E D C * B - /
A + ((B – C * D) / E) + F – G / H	Pop and append to prefix until precedence is equal or less than + and Push +	+	H G / F – E D C * B - / +
A + ((B – C * D) / E) + F – G / H	It is an operand, Append A to prefix	+	H G / F – E D C * B - / + A
A + ((B – C * D) / E) + F – G / H	When it is ended Pop and append to prefix until stack is empty		H G / F – E D C * B - / + A +

We reverse the output string to get prefix expression so;

+ A + / - B * C D E – F / G H

i)

PREFIX

EVALUATION

SCANNING RIGHT TO LEFT

A = 20, B =15, C = 2, D = 6, E = 3, F = 8, G = 16, H = 4

Expression	Action	Stack
+ A + / – B * C D E – F / G H	Push H	4
+ A + / – B * C D E – F / G H	Push G	16 4
+ A + / – B * C D E – F / G H	Pop twice , Do math and push result to stack	4
+ A + / – B * C D E – F / G H	Push F	8 4

+ A + / – B * C D E – F / G H	Pop twice , Do math and push result to stack	4
+ A + / – B * C D E – F / G H	Push E	3 4
+ A + / – B * C D E – F / G H	Push D	6 3 4
+ A + / – B * C D E – F / G H	Push C	2 6 3 4
+ A + / – B * C D E – F / G H	Pop twice, Do math and push result to stack	12 3 4
+ A + / – B * C D E – F / G H	Push B	15 12 3 4
+ A + / – B * C D E – F / G H	Pop twice, Do math and push result to stack	3 3 4
+ A + / – B * C D E – F / G H	Pop twice, Do math and push result to stack	1 4
+ A + / – B * C D E – F / G H	Pop twice, Do math and push result to stack	5
+ A + / – B * C D E – F / G H	Push A	20 5
+ A + / – B * C D E – F / G H	Pop twice, Do math and push result to stack	25

ii)

Infix to postfix

A B C < C D > | | ! && ! C E < | |

CONVERSION

Expression	Action	Stack	Effect on postfix
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push !	!	
! (A && ! ((B < C) || (C > D))) || (C < E)	Push (	(!	
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append A to postfix	(!	A
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push &&	&& (!	A
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push !	! && (!	A
! (A && ! ((B < C) || (C > D))) || (C < E)	Push (	(! && (!	A

! (A && ! ((B < C) || (C > D))) || (C < E)	Push (	((! && (!	A
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append B to postfix	((! && (!	A B
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push <	< ((! && (!	A B
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append C to postfix	< ((! && (!	A B C
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	(! && (!	A B C <
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push ||	|| (! && (!	A B C <
! (A && ! ((B < C) || (C > D))) || (C < E)	Push (	(|| (! &&	A B C <

		(!	
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append C to postfix	(|| (! && (!	A B C < C
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push >	>(|| (! && (!	A B C < C
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append D to postfix	>(|| (! && (!	A B C < C D
! (A && ! ((B < C) || (C > D)))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	||(! && (!	A B C < C D >
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	!&&(!	A B C < C D > ||
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	!	A B C < C D > || ! &&

! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than || and Push ||	||	A B C < C D > || ! && !
! (A && ! ((B < C) || (C > D))) || (C < E)	Push (	(||	A B C < C D > || ! && !
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append C to postfix	(!	A B C < C D > || ! && ! C
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push <	< (||	A B C < C D > || ! && ! C
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append E to postfix	< (||	A B C < C D > || ! && ! C E
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	||	A B C < C D > || ! && ! C E <
! (A && ! ((B < C) || (C > D))) || (C < E)	When it is ended Pop and append to postfix until stack is empty		A B C < C D > || ! && ! C E < ||

ii)

POSTFIX

EVALUATION

SCANNING LEFT TO RIGHT

A = 20, B =15, C = 2, D = 6, E = 3

Expression	Action	Stack
A B C < C D > | | ! && ! C E < | |	Push A	20
A B C < C D > | | ! && ! C E < | |	Push B	15 20
A B C < C D > | | ! && ! C E < | |	Push C	2 15 20
A B C < C D > | | ! && ! C E < | |	Pop twice , Do math and push result to stack	0 20
A B C < C D > | | ! && ! C E < | |	Push C	2 0 20
A B C < C D > | | ! && ! C E < | |	Push D	6 2 0 20
A B C < C D > | | ! && ! C E < | |	Pop twice, Do math and push result to stack	0 0 20
A B C < C D > | | ! && ! C E < | |	Pop twice, Do math and push result to stack	0 20
A B C < C D > | | ! && ! C E < | |	Pop once, Do math and push result to stack	1 20
A B C < C D > | | ! && ! C E < | |	Pop twice, Do math and push result to stack	1
A B C < C D > | | ! && ! C E < | |	Pop once, Do math and push result to stack	0

A B C < C D > | | ! && ! C E < | |	Push C	2 0
A B C < C D > | | ! && ! C E < | |	Push E	3 2 0
A B C < C D > | | ! && ! C E < | |	Pop twice, Do math and push result to stack	1 0
A B C < C D > | | ! && ! C E < | |	Pop twice, Do math and push result to stack	1

ii)

Infix to prefix

! | | && A ! | | < B C > C D < C E

CONVERSION

SCANNING RIGHT TO LEFT

Expression	Action	Stack	Effect on postfix
! (A && ! ((B < C) | | (C > D))) | | (C < E)	Push)	)	
! (A && ! ((B < C) | | (C > D))) | | (C < E)	It is an operand, Append E to prefix	)	E
! (A && ! ((B < C) | | (C > D))) | | (C < E)	It is an operator, İts precedence is higher , Push <	<)	E
! (A && ! ((B < C) | | (C > D))) | | (C < E)	It is an operand, Append C to prefix	<)	E C

! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to postfix until precedence is equal or less than)	||	E C <
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push ||	)	E C <
! (A && ! ((B < C) || (C > D))) || (C < E)	Push)	||	E C <
! (A && ! ((B < C) || (C > D)) || (C < E)	Push)	)	E C <
! (A && ! ((B < C) || (C > D)) || (C < E)	Push)	)	E C <
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append D to prefix	)	E C < D
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push >	>	E C < D
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append C to prefix	>	E C < D C
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to prefix until precedence is equal or less than (	)	E C < D C >
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push ||	||	E C < D C >
! (A && ! ((B < C) || (C > D))) || (C < E)	Push)	)	E C < D C >
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append C to prefix	)	E C < D C > C

		||)) ||	
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push <	<) ||)) ||	E C < D C > C
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append B to prefix	<) ||)) ||	E C < D C > C B
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to prefix until precedence is equal or less than (	||)) ||	E C < D C > C B <
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to prefix until precedence is equal or less than (	) ||	E C < D C > C B < ||
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push !	!) ||	E C < D C > C B < ||
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is lower , pop and append to prefix , then push !	&&) ||	E C < D C > C B < || !
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operand, Append A to prefix	&&) ||	E C < D C > C B < || ! A
! (A && ! ((B < C) || (C > D))) || (C < E)	Pop and append to prefix until precedence is equal or less than (	||	E C < D C > C B < || ! A &&
! (A && ! ((B < C) || (C > D))) || (C < E)	It is an operator, İts precedence is higher , Push !	! ||	E C < D C > C B < || ! A &&
! (A && ! ((B < C) || (C > D))) || (C < E)	When it is ended	||	E C < D C > C B < || ! A && !

	Pop and append to prefix until stack is empty		
! (A && ! ((B < C) || (C > D))) || (C < E)	When it is ended Pop and append to prefix until stack is empty		E C < D C > C B < || **!** A && ! ||

We reverse the output string to get prefix expression so;

|| ! && A ! || < B C > C D < C E

ii)

PREFIX

EVALUATION

SCANNING RIGHT TO LEFT

A = 20, B =15, C = 2, D = 6, E = 3

Expression	Action	Stack
|| ! && A ! || < B C > C D < C E	Push E	3
|| ! && A ! || < B C > C D < C E	Push C	2 3
|| ! && A ! || < B C > C D < C E	Pop twice , Do math and push result to stack	1
|| ! && A ! || < B C > C D < C E	Push D	6 1
|| ! && A ! || < B C > C D < C E	Push C	2 6 1
|| ! && A ! || < B C > C D < C E	Pop twice , Do math and push result to stack	0 1
|| ! && A ! || < B C > C D < C E	Push C	2 0 1
|| ! && A ! || < B C > C D < C E	Push B	15 2 0 1

|| ! && A ! || < B C > C D < C E	Pop twice, Do math and push result to stack	0 0 1
|| ! && A ! || < B C > C D < C E	Pop twice, Do math and push result to stack	0 1
|| ! && A ! || < B C > C D < C E	Pop once, Do math and push result to stack	1 1
|| ! && A ! || < B C > C D < C E	Push A	20 1 1
|| ! && A ! || < B C > C D < C E	Pop twice, Do math and push result to stack	1 1
|| ! && A ! || < B C > C D < C E	Pop once, Do math and push result to stack	0 1
|| ! && A ! || < B C > C D < C E	Pop once, Do math and push result to stack	1