1.
The code is shown in the java file

2.
The code is shown in the java file
Sample data: “It was – the best - - of times –”
(1) After 2 enqueues

It	was	null	null	null	null	null	null	null	null
front	last								

Output:
(2) After first “–“, dequeue and print

null	was	null	null	null	null	null	null	null	null
	front/last								

Output: It
(3) After 2 enqueues

null	was	the	best	null	null	null	null	null	null
	front		last						

Output: It
(4) After 2 “-“, dequeue and print

null	null	null	best	null	null	null	null	null	null
			front/last						

Output: It was the
(5) After 2 enqueues

null	null	null	best	of	times	null	null	null	null
			front		last				

Output: It was the
(6) After “-“, dequeue and print

null	null	null	null	of	times	null	null	null	null
				front	last				

Output: It was the best

3.
(1) After 2 enqueues

The	temperature	null	null	null	null	null	null	null	null
front	last								

Output:
(2) After 2 “-“, dequeue and print

null	null	null	null	null	null	null	null	null	null
	last	front							

Output: The temperature
(3) After 4 enqueues

null	null	degrees	today	and	it	null	null	null	null
		front			last				

Output: The temperature

(4) After 3 "-", dequeue and print

null	null	null	null	null	it	null	null	null	null
					front/last				

Output: The temperature degrees today and

(5) After 1 enqueue

null	null	null	null	null	it	tomorrow	null	null	null
					front	last			

Output: The temperature degrees today and

4.

A) 10 + 2 * 8 – 3

(1)

First number is 10, output it

Output: 10

(2)

+

Then comes the first operator "+", push it into the stack

Output: 10

(3)

+

The next number is 2, output it

Output: 10 2

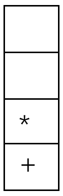
(4)

*
+

The next operator is "*", since the top operator in the stack "+" has lower priority then "*", push "*" into the stack

Output: 10 2

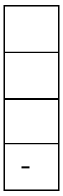
(5)



The next number is 8, output it

Output: 10 2 8

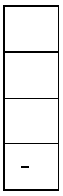
(6)



The next operator is "-", which has a lower priority then "*" which is at the top of the stack. So pop. Then "+" which is now at the top of the stack has the same priority as "-", but it's on the left of "-", so pop.

Output: 10 2 8 * +

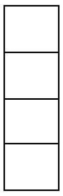
(7)



The next number is 3, output it

Output: 10 2 8 * + 3

(8)



Then pop all the elements from the stack

Output: 10 2 8 * + 3 –

B) The code is shown in the java file

5.

h) the time-complexity is O(1) for the methods "isEmpty()", "enqueue" and "dequeue"

j) the time-complexity is O(1) for the methods "isEmpty()", "enqueue" and "dequeue"

k) for fixed array size, when oversizing, use resizing array for array implementation. When undersizing, throw exception if dequeue from an empty queue.

6.

A)

(A+B)*C+D/(E+F*G)-H

token: (

operand								
operator	(							

token: A

operand	A							
operator	(							

token: +

operand	A							
operator	(	+						

token: B

operand	A	B						
operator	(	+						

token:)

operand	A+B							
operator								

token: *

operand	A+B							
operator	*							

token: C

operand	A+B	C						
operator	*							

token: +

operand	A+B	C						
operator	*	+						

token: D

operand	A+B	C	D					
operator	*	+						

token: /

operand	A+B	C	D					
operator	*	+	/					

token: (

operand	A+B	C	D					
operator	*	+	/	(				

token: E

operand	A+B	C	D	E				
operator	*	+	/	(				

token: +

operand	A+B	C	D	E				
operator	*	+	/	(	+			

token: F

operand	A+B	C	D	E	F			
operator	*	+	/	(	+			

token: *

operand	A+B	C	D	E	F			
operator	*	+	/	(	+	*		

token: G

operand	A+B	C	D	E	F	G		
operator	*	+	/	(	+	*		

token:) (processing 1)

operand	A+B	C	D	E	F*G			
operator	*	+	/	(	+			

(processing 2)

operand	A+B	C	D	E+F*G				
operator	*	+	/					

token: -

operand	A+B	C	D	E+F*G				
operator	*	+	/	-				

token: H

operand	A+B	C	D	E+F*G	H			
operator	*	+	/	-				

processing 1

operand	A+B	C	D/(E+F*G)	H				
operator	*	+	-					

processing 2

operand	(A+B)*C	D/(E+F*G)	H					
operator	+	-						

processing 3

operand	(A+B)*C+ D/(E+F*G)	H						
operator	-							

processing 4 (operator stack is empty)

operand	(A+B)*C+ D/(E+F*G)-H							
operator								

(300+23)*(43-21)/(84+7)

token	action	operand stack	operator stack	note
(	push it to operator stack		(	
300	push it to operand stack	300	(	
+	push it to operator stack	300	+ (	
23	push it to operand stack	23 300	+ (	
)	pop 23 and 300 from operand stack		+ (	Do process until (is popped from operator stack
	pop + from operator stack		(	
	do 23 + 300 = 323		(	
	push 323 to operand stack	323	(	
	pop (from operator stack	323		
*	push it to operator stack	323	*	
(	push it to operator stack	323	(*	
43	push it to operand stack	43 323	(*	
-	push it to operator stack	43 323	- (*	
21	push it to operand stack	21 43 323	- (*	
)	pop 21 and 43 from operand stack	323	- (*	Do process until (is popped from operator stack
	pop – from operator stack	323	(*	
	do 43 – 21 = 22	323	(*	
	push 22 to operand stack	22 323	(*	
	pop (from operator stack	22 323	*	
/	pop 22 and 323 from operand stack		*	
	pop * from operator stack			
	do 22 * 323 = 7106			
	push 7106 to operand stack	7106		
	push / to operator stack	7106	/	
(	push it to operator stack	7106	(/	
84	push it to operand stack	84 7106	(/	
+	push it to operator stack	84 7106	+ (/	
7	push it to operand stack	7 84 7106	+ (/	
)	pop 7 and 84 from operand stack	7106	+ (/	Do process until (is popped from operator stack
	pop + from operator stack	7106	(/	
	do 7 + 84 = 91	7106	(/	
	push 91 to operand stack	91 7106	(/	
	pop (from operator stack	91 7106	/	
	pop 91 and 7106 from operand stack		/	
	pop / from operator stack			
	do 7106 / 91 = 78			

	push 78 to operand stack	78		

(4+8)*(6-5)/((3-2)*(2+2))

token	action	operand stack	operator stack	note
(	push it to operator stack		(	
4	push it to operand stack	4	(	
+	push it to operator stack	4	+ (	
8	push it to operand stack	8 4	+ (	
)	pop 4 and 8 from operand stack		+ (	Do process until (is popped from operator stack
	pop + from operator stack		(	
	do 4 + 8 = 12		(	
	push 12 to operand stack	12	(	
	pop (from operator stack	12		
*	push it to operator stack	12	*	
(	push it to operator stack	12	(*	
6	push it to operand stack	6 12	(*	
-	push it to operator stack	6 12	- (*	
5	push it to operand stack	5 6 12	- (*	
)	pop 5 and 6 from operand stack	12	- (*	Do process until (is popped from operator stack
	pop - from operator stack	12	(*	
	do 6 – 5 = 1	12	(*	
	push 1 to operand stack	1 12	(*	
	pop (from operator stack	1 12	*	
/	pop 1 and 12 from operand stack		*	
	pop * from operator stack			
	do 1 * 12 = 12			
	push 12 to operand stack	12		
	push / to operator stack	12	/	
(	push it to operator stack	12	(/	
(	push it to operator stack	12	((/	
3	push it to operand stack	3 12	((/	
-	push it to operator stack	3 12	- ((/	
2	push it to operand stack	2 3 12	- ((/	
)	pop 2 and 3 from operand stack	12	- ((/	Do process until (is popped from operator stack
	pop - from operator stack	12	((/	
	do 3 - 2 = 1	12	((/	
	push 1 to operand stack	1 12	((/	
	pop (from operator stack	1 12	(/	

*	push it to operator stack	1 12	* (/	
(	push it to operator stack	1 12	(* (/	
2	push it to operand stack	2 1 12	(* (/	
+	push it to operator stack	2 1 12	+ (* (/	
2	push it to operand stack	2 2 1 12	+ (* (/	
)	pop 2 and 2 from operand stack	1 12	+ (* (/	Do process until (is popped from operator stack
	pop + from operator stack	1 12	(* (/	
	do 2 + 2 = 4	1 12	(* (/	
	push 4 to operand stack	4 1 12	(* (/	
	pop (from operator stack	4 1 12	* (/	
)	pop 4 and 1 from operand stack	12	* (/	Do process until (is popped from operator stack
	pop * from operator stack	12	(/	
	do 4 * 1 = 4	12	(/	
	push 4 to operand stack	4 12	(/	
	pop (from operator stack	4 12	/	
	pop 4 and 12 from operand stack		/	
	pop / from operator stack			
	do 12 / 4 = 3			
	push 3 to operand stack	3		

7.

A*B/C+(D+E-(F*(G/H)))

Symbol	Scanned	Stack	Postfix Expression	Description
1		(		Start
2	A	(	A	
3	*	(*	A	
4	B	(*	AB	
5	/	(/	AB*	
6	C	(/	AB*C	
7	+	(+	AB*C/	
8	(	(+(	AB*C/	
9	D	(+(	AB*C/D	
10	+	(+(+	AB*C/D	
11	E	(+(+	AB*C/DE	
12	-	(+(-	AB*C/DE+	
13	(	(+(-(	AB*C/DE+	
14	F	(+(-(	AB*C/DE+F	
15	*	(+(-(*	AB*C/DE+F	
16	(	(+(-(*(	AB*C/DE+F	
17	G	(+(-(*(	AB*C/DE+FG	

18	/	(+(-(*(/	AB*C/DE+FG	
19	H	(+(-(*(/	AB*C/DE+FGH	
20	)	(+(-(*	AB*C/DE+FGH/	
21	)	(+(-	AB*C/DE+FGH/*	
22	)	(+	AB*C/DE+FGH/*-	
23	)		AB*C/DE+FGH/*-+	END