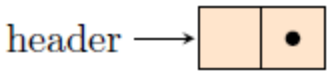
CS-2134

Spring 2015

HW5 Written Solutions

1. Empty Linked List Header:



2. Diagrams for Programming Question 1

push_front()

Before: |H| -> |A| -> …

After: |H| -> |B| -> |A| -> …

pop_front()

Before: |H| -> |A| -> |B| -> …

After: |H| -> |B| -> …

erase_after(A)

Before: |H| -> |A| -> |B| -> …

After: |H| -> |A| -> …

a. push_front()

-Create a new node that points to what the header node points to
-Point the header node at this new node

b. pop_front()

-Create a temp node pointer that points at the node after the node header points at
-Delete the node after header
-Point the header node at the node temp points at

c. erase_after()

-If the position passed in is null or the next node is null, return a null LListItr
-Otherwise, keep a temporary iterator pointing to the Node after position's node, set position's next pointer to the temporary iterator's next pointer then delete the node pointed to by the temporary iterator.
-Decrement the size and return an iterator to the node after the one pointed to by the position iterator.

3.

a. *Valid*: push_back() and push_front() are defined for lists such that they do not invalidate iterators

b. *Invalid:* "mid" is a forward iterator, and you cannot perform addition/subtraction arithmetic directly on forward iterators.
c. *Valid:* "mid" is now a random access iterator, and addition/subtraction is defined.

4. The modulo operator has the same precedence as the division & multiplication operators. Therefore the modification is:

```
enum TokenType { EOL, VALUE, OPAREN, CPAREN, EXP,
MULT, DIV, MOD, PLUS, MINUS };
struct Precedence
{
int inputSymbol;
int topOfStack;
} PREC_TABLE [ ] =
{
{ 0, -1 }, { 0, 0 }, // EOL, VALUE
{ 100, 0 }, { 0, 99 }, // OPAREN, CPAREN
{ 6, 5 }, // EXP
{ 3, 4 }, { 3, 4 }, {3,4} // MULT, DIV, MOD
{ 1, 2 }, { 1, 2 } // PLUS, MINUS
};
```

5. EXP
   EXP
   DIV

   Stack contents (from bottom to top): EOL, PLUS, MULT

   (Note that the letter representation of the output & stack content is shown above. The actual output will consist of the corresponding number values however both answers will be accepted)

6. This code will not work properly! The problem is we are deleting the Old Node *before* keeping track of its next node. Thus we will lose track of the portion of the list after the node we are deleting.

7.

a.

Input	stack
8	8
4	8 4
2	8 4 2
/	8 2
3	8 2 3
3	8 2 3 3
^	8 2 27
-	8 -25
+	-17

b.

Input	stack
2	2
4	2 4
^	16
2	16 2
6	16 2 6
*	16 12
-	4

c.

Input	stack
4	4
2	4 2
3	4 2 3
*	4 6
-	-2
3	-2 3
2	-2 3 2
^	-2 9
-	-11
6	-11 6
+	-5

d.

Input	stack
9	9
3	9 3
/	3
2	3 2
*	6
1	6 1
-	5

e.

Input	stack
2	2
5	2 5
/	0.4
8	0.4 8
+	8.4
7	8.4 7
*	58.8
4	58.8 4
+	62.8

8.

a.

Input	Stack	Output Stream
(	(	
5	(	5
-	(-	5
2	(-	5 2
)		5 2 -
/	/	5 2 -
3	/	5 2 - 3
+	+	5 2 - 3 /
3	+	5 2 - 3 / 3
^	+ ^	5 2 - 3 / 3
2	+ ^	5 2 - 3 / 3 2 ^ +
eof		

b.

Input	Stack	Output Stream
8		8
+	+	8
(	+ (	8
2	+ (	8 2
^	+ (^	8 2
3	+ (^	8 2 3
)	+	8 2 3 ^
^	+^	8 2 3 ^
2	+ ^	8 2 3 ^ 2
eof		8 2 3 ^ 2 ^ +

c.

Input	Stack	Output Stream
2		2
^	^	2
3	^	2 3
^	^ ^	2 3
2	^ ^	2 3 2
+	+	2 3 2 ^ ^
8	+	2 3 2 ^ ^ 8
eof		2 3 2 ^ ^ 8 +

d.

Input	Stack	Output Stream
(	(	
2	(	2
+	(+	2
6	(+	2 6
)		2 6 +
/	/	2 6 +
3	/	2 6 + 3
-	-	2 6 + 3 /
(	- (	2 6 + 3 /
(	-((	2 6 + 3 /
32	- ((	2 6 + 3 / 32
+	- ((+	2 6 + 3 / 32
4	- ((+	2 6 + 3 / 32 4
)	- (	2 6 + 3 / 32 4 +
*	- (*	2 6 + 3 / 32 4 +
7	- (*	2 6 + 3 / 32 4 + 7
)	-	2 6 + 3 / 32 4 + 7 * +
*	- *	2 6 + 3 / 32 4 + 7 * +
2	- *	2 6 + 3 / 32 4 + 7 * + 2
eof		2 6 + 3 / 32 4 + 7 * + 2 * -

e.

Input	Stack	Output Stream
3		3
*	*	3
2	*	3 2
/	/	3 2 *
4	/	3 2 * 4
*	*	3 2 * 4 /
5	*	3 2 * 4 / 5
eof		3 2 * 4 / 5 *