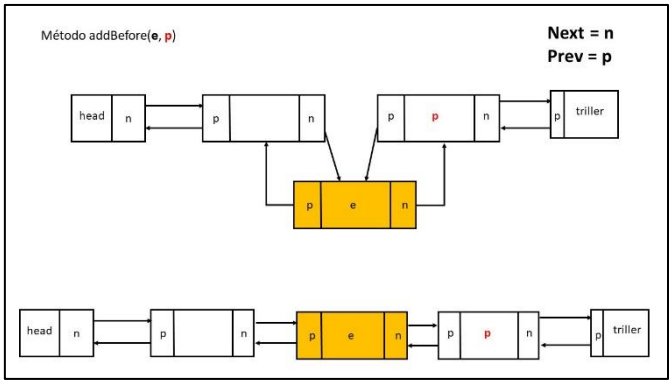
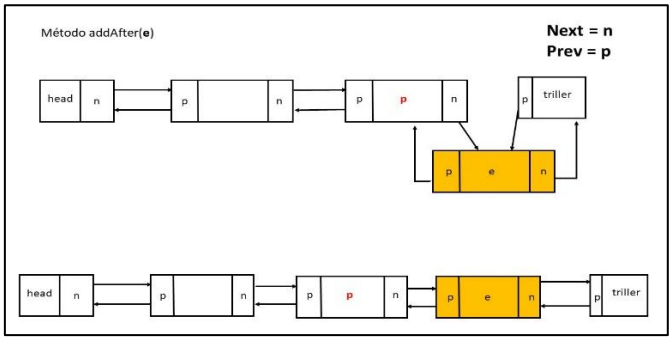
1) Desenvolva um projeto no GitHub que Implemente os testes o TAD Lista Arranjo.
   ------OK-------
2) Desenvolva um projeto no GitHub que estenda a classe de testes do TAD Pilha implementado usando array que teste uma pilha de String.
   -------OK-------
3) Desenvolva um projeto no GitHub que implemente os testes do TAD Pilha usando LSE.
   ------OK-------

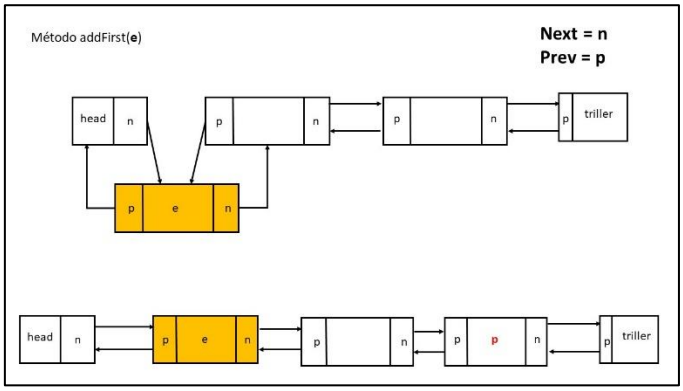
4) Exercícios: ------OK----------
   a. Crie testes e programas Java que:
   b. Inverta os dados de um arranjo usando o TAD Pilha usando LSE.
   c. Verifique se parênteses, colchetes e chaves estão corretos numa expressão matemática, por exemplo: [(5 + x)/4 – 2*(y + z)]
      i. Correto: ()(( )){([[ ]])}
      ii. Correto: ((( )(( )){([[ ]])}))
      iii. Incorreto:)(( )){([[ ]])}
      iv. Incorreto: ({[])}
      v. Incorreto: (
5) Suponha que uma lista inicialmente vazia S tenha executado um total de 25 operações push, 12 operações top e 10 operações pop, 3 das quais geraram StackEmptyExceptions, que foram capturadas e ignoradas. Qual é o tamanho corrente de S? O tamanho é de 15.
6) Se implementarmos a pilha S do problema anterior usando um arranjo, então qual será o valor corrente da variável de instância top? O tamanho é de 15.
7) Descreva a saída resultante da seguinte série de operações de pilha: push(5), push(3), pop( ),push(2), push(8), pop( ), pop( ), push(9), push(1), pop( ), push(7), push(6), pop(), pop(), push(4), pop(), pop( ).

Operação	Saída	Conteúdo da pilha
push(5)	-	"(5)"
push(3)	-	"(5,3)"
pop()	3	"(5)"
push(2)	-	"(5,2)"
push(8)	-	"(5,2,8)"
pop()	8	"(5,2)"
push(9)	-	"(5,2,9)"
push(1)	-	"(5,2,9,1)"
pop()	1	"(5,2,9)"
push(7)	-	"(5,2,9,7)"
push(6)	-	"(5,2,9,7,6)"
pop()	6	"(5,2,9,7)"
pop()	7	"(5,2,9)"
push(4)	-	"(5,2,9,4)"
pop()	4	"(5,2,9)"
pop()	9	"(5,2)"

8) Crie os testes e implemente o TAD Fila. Use implementação do TAD Pilha como exemplo.--------OK----------
9) Implemente o TAD Fila com base nos testes e no fragmento de implementação de duas operações apresentados a seguir (Tarefa 13 - TAD-Fila.pptx, slidese 19, 20 e 21). --------OK-----------
10) Desenhe figuras demonstrando cada um dos passos principais dos métodos addBefore(p, e), addFirst(e) e addLast(e) do TAD lista de nodos.







11) Implemente um método não recursivo para inverter uma lista de nodos.
12) Implemente um novo método, makeFirst(p), que move o elemento na posição p para a primeira posição, mantendo a ordem relativa dos demais elementos inalterada.
    --------OK-------
13) A implementação de NodePositionList não faz verificações de erro para testar se uma dada posição p é realmente membro dessa lista em particular.
    a. Por exemplo, se p é uma posição da lista S, a execução T.addAfter(p,e) deveria lançar a exceção InvalidPositionException pois p não é uma posição de T.
    b. Descreva como alterar a implementação de NodePositionList de uma forma eficiente que impeça esses maus usos: Para isso será criado o método "checkposition(Position<E> p)" que checa se a posição é válida para a lista, e cria um DNode se a posição for válida.