---------------------------------------------------------------------------------------------------------------------

Algorithm: Push front(x) – spesifikasjon

Input: et element X som skal settes inn fremst i en kø (køen er tiltenkt å ha struktur som en indeksert lenket liste

Output: Ikke nødvendig, eventuelt en melding som informerer om vellykket innsetting

* Algoritmen sjekker om antall elementer i køen > 0, hvis dette ikke er tilfellet settes element x inn i køen, og det behøver ikke gjøres noen andre endringer.  
  Dersom det finnes elementer i køen fra før vil element x settes inn fremst i køen og kobles til elementet i køen som originalt sto på index 0.

Implementasjon:

Input: et element x

Output: ikke nødvendig

1 Procedure pushFront(x)

2 | if (queue.length == 0)

3 | | queue.first 🡨 x, queue.last 🡨 x

4 | | return

5 | else

6 | | x.next 🡨 queue.first

7 | | queue.first 🡨 x

8 | | return

---------------------------------------------------------------------------------------------------------------------

Algorithm: Push back(x) – spesifikasjon

Input: et element X som skal settes inn bakerst i en kø (køen er tiltenkt å ha struktur som en indeksert lenket liste

Output: Ikke nødvendig, eventuelt en melding som informerer om vellykket innsetting

* Algoritmen sjekker om antall elementer i køen > 0, hvis dette ikke er tilfellet vil køens første og siste-referanser bli satt til å være x.
* Dersom det finnes elementer i køen fra før vil køens siste referanse settes til å være lik x, og elementet som var på køens index -1 vil kobles til x

Implementasjon:

Input: et element x

Output: ikke nødvendig

1 Procedure PushBack(x)

2 | if (queue.length == 0)

3 | | queue.first 🡨 x, queue.last 🡨 x

4 | | return

5 | else

6 | queue.last.next 🡨 x

7 | queue.last 🡨 x

8 | return

---------------------------------------------------------------------------------------------------------------------

Algorithm: Push Middle(x) – spesifikasjon

Input: et element X som skal settes inn i midten av en kø (køen er tiltenkt å ha struktur som en indeksert lenket liste

Output: Ikke nødvendig, eventuelt en melding som informerer om vellykket innsetting

* Algoritmen sjekker om antall elementer i køen > 0, hvis dette ikke er tilfellet settes element x inn i køen, og det behøver ikke gjøres noen andre endringer.  
  Dersom det finnes elementer i køen fra før vil element x settes inn miderst i køen og kobles til elementene som er på elementets index -1 og +1

Implementasjon:

1 Procedure pushMiddle(x)

2 | if (queue.length == 0)

3 | | queue.first 🡨 x, queue.last 🡨 x

4 | | return

5 | else if (queue.length == 1)

6 | | pushback(x)

7| | return

8 | else

9 | | container temp 🡨 (queue.length + 1 / 2)

10 | | x.previous 🡨 temp.previous

11 | | x.next 🡨 temp

12 | | x.previous.next 🡨 x

13 | | x.next.previous 🡨 x

14 | | return

---------------------------------------------------------------------------------------------------------------------

Algorithm: Get(i) – spesifikasjon

Input: Et heltall som korresponderer til en index i køen

Output: Elementet som befinner seg på gitt index i køen

* Algoritmen starter fremst i køen, og traverserer elementene via deres next referanser i -1 ganger, slik at den kan hente ut det aktuelle elementet via elementet på i -1 sin next referanse og returnerer elementet.

Implementasjon:

1 Procedure get(i)

2 | pointer 🡨 queue.first

3 | for J 🡨 0 to I -1 do

4 | | pointer 🡨 pointer.next

5| return pointer.next