c) Kjøretidskompleksitetsanalyse:

* PushFront(x)
* Prosedyren gjør 1 av 2 ting avhengig av if-testen, i alle disse tilfellene gjør prosedyren konstanttidsoperasjoner, disse kan vi tillate å sette til 1, så kompleksiteten til pushFront vil være O(1)
* PushBack(x)
* Oppfører seg tilsvarende som pushFront(x) og vil også ha kjøretidskompleksitet like O(1)
* PushMiddle (x)

Metoden gjør et iterativt kall for å hente relevant index, og vil i verste tilfelle iterere over hele køen(Dette er mest aktuelt hvis det er få elementer i køen), i tillegg gjør den noen konstanttidsoperasjoner som if-tester og tilordninger av verdier til variabler, disse kan settes til 1. Worst-case vil prosedyren ha kjøretidskompleksitet på O(n). I store køer vil kjøretidskompleksiteten likne mer på O(n/2), fordi den alltid kun vil trenge å iterere over halve køen, i praksis så er dette en forskjell fra O(n), men jeg vet ikke hvor viktig det er å tenke over

* Get(i)

Vil også ha worst-case kjøretidskompleksitet på O(n) fordi den gjør i iterasjoner for å komme frem til riktig element i køen, som i verste tilfelle kan være det siste elementet.

d) Det er viktig å fjerne begrensningen på n nettopp fordi vi tillater oss å sette verdien til alle konstanter til 1. Dersom vi setter en begrensning på n vil et worst case iterativt tilfelle være f.eks 10^6 iterasjoner, men fordi dette er en konstant vil dette fortsatt bare telle som 1, og alle prosedyrene ville fått konstant-tid kjøretidskompleksitet.