Innlevering 1- IN2010

Oppgave : Teque

a)

```
/* Jeg har valgt å bruke en lenkeliste for å løse problemet.
Dette er en enkel tilnærming, men det vil ikke være mulig
å oppnå konstant tid O(1) for push_middle og get(i) med denne måte å gjøre det på
siden vi må traverse i lenkelisten.
 */

Procedure: push_back(x):
    // Legger til bakerst.
    list.push_last(x)

Procedure: push_front(x):
    // Legger til fremst.
    list.push_first(x)

Procedure: push_middle(x):
    //Finner midt-indeksen.
    mid_index <-(|list| + 1) / 2
    // Setter inn element i midten.
    list.insert(mid_index, x)

Procedure: get(i):
    // Henter element på indeks i.
    return list[i]
```

b) Se vedlagt javafil.

c) **Verste-tilfelle kjøretidsanalyse**

- push_back(x): Denne opperasjonen utføres i O(1) tid, side det bare legges til et elemnt på slutten av lenkelisten.

- push_front(x): Utføres også i O(1) tid, siden det bare legges til et element på starten av lenkelisten.

- push_middle(x): Forst må vi finne midten for så å sette inn elementet. Det å fine midten tar O(N) tid side vi må gå gjennom N/2 i verste fall. Videre vil det å sette inn et element ta O(N) tid i verste tilfelle fordi vi må flytte elementer for å sette inn et nytt element i midten. Operasjonen tar derfor i verste tilfellet O(N) tid.

- get(i): Denne opperasjonen tar O(N) tid i verste tilfellet siden vi må traverse gjennom lenkelisten for å fine indeksen til ‘i’.

d) Det kan være vikitg å ta bort begreninsingen på N for at vi kan gi en genrell beskrivelse av hvordan algoritmen oppfører seg ved store datamengder. Det her kan hjelpe oss med å skjønne hvordan algoritmen skalerer og om den er robust. N er en konstant og i en kjøretidsanalyse (O-notasjon) ønkser vi å se hvordan algortimen oppfører seg når N nærmer seg uendleig.

Oppgave: Binærsøk med lenkede lister

Verste kjøretiden for binært søk over lenked lister er O(n * log(n)). Valget å bruke lenkeliste som datastruktur kan være med på å øke kjøretidskomplekisteten siden vi må traversere gjennom den lenkede listen for å fine frem til rett element. Selv binæresøket har en en kompleksitet på O(log(n), men fordi hver gang man skal ha tilgang til et element “i” for den lenkede lista så tar det O(n) tid i verste fall, vil altså den sammlede kompelksiteten for binærsøket i værste fall være O(n *log(n)).