**Oppgave 1**

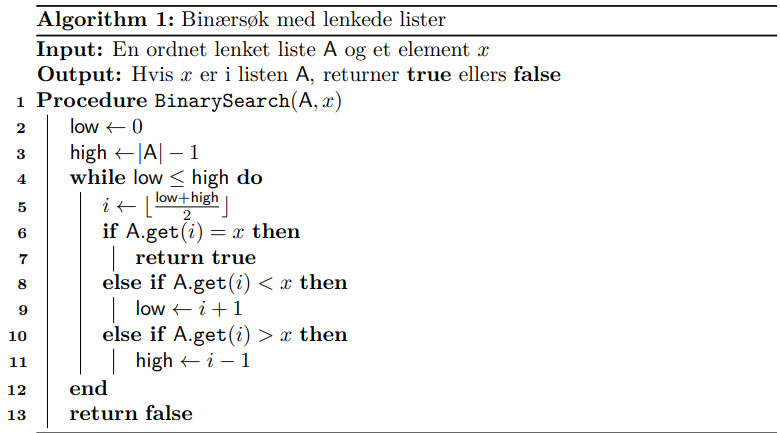
b)

* Push\_back
  + Benytter innebygd metode i ArrayList, elementet legges sist i listen ved hjelp av metoden add(x), der x er elementet som skal legges inn. Push\_back vil ha en konstant kjøretid på O(1), ettersom den alltid gjør den samme operasjonen.
* Push\_front
  + Benytter en innebygd metode i ArrayList som i Push\_back, men denne kan legge inn elementer på spesifikke indekser, syntaks; add(indeksen, x), der x er elementet. Alle elementer flyttes bakover, og derfor vil operasjonen ha en lineær kjøretid på O(n), der tiden blir større jo flere elementer det er i listen.
* Push\_middle
  + Vil ha en lineær kjøretid på O(n), og har samme begrunnelse som i Push\_front.
* Get
  + Metoden get(indeks), henter elementet på indeksen spesifisert. Operasjonen har en konstant kjøretid på O(1), ettersom den alltid gjør den samme operasjonen.

c)

Hvis N er begrenset og ikke vilkårlig stor, så skal det teoretisk sett ikke ha noe å si. O-notasjoner har ingen konstanter, og vil etter en stund hvor N blir stor nok, der konstanter ikke betyr noe. For eksempel vil en algoritme 1 som benytter N2 tid og en algoritme 2 som benytter 10 \* N2 + N tid, ha samme tid: O(N2). Men algoritme 1 vil være raskere, ettersom den er enklere og da vil konstanter bety noe.

**Oppgave 2**

Vi har denne algoritmen:

Operasjonen *A.get(i) = x* kan benyttes som en elementær operasjon. I denne algoritmen er antallet av sammenligninger ikke bare avhengig av antall elementer n i listen, men også av verdien av elementet x. Det vil si at:

* Hvis x ikke er lik et element i A, så gjør operasjonen O(log n) sammenligninger.
* Men hvis x er lik et element i A, så gjøres det en sammenligning.

Dermed kan vi gi et worst case estimat;

* Hvis elementet x ikke finnes i listen A. Det gir [log2n + 1] iterasjoner hvis den når enden av listen. Det er absolutt worst case.
* Søket kan også nå nest siste del av listen 🡪 [log2n]

Lenkede lister er også en datastruktur som er bevisst tregere enn for eksempel array og arraylist. For eksempel er tilgang til et tilfeldig element i et array med lengden N lik O(1), noe som betyr at det i verste fall bare er ett trinn. Dette fordi man benytter en indeks for å finne elementet. For en lenket liste over lengde N, er det verste tilfellet N, så det er O(N). Det betyr at du må gjennom alle elementene, og i verste fall kan det være det siste elementet i listen.