LinkedList

Operation	Hvad sker der?
Find på index	Listen starter forfra og tæller sig frem med ét skridt ad gangen.
Find et objekt	Vi starter forfra og kigger på hver enkelt, til vi finder det.
Indsæt først eller sidst	Den nye element kobles bare på og ingen andre flyttes.
Indsæt midt i	Listen skal først tælle sig frem til indexet, men selve indsættelsen er nem: koblinger omlægges.
Fjern først eller sidst	Koblingen omlægges og ingen andre flyttes.
Fjern midt i	Listen skal først tælle sig frem til indexet, men selve fjernelsen nem.

ERHVERVSAKADEMI KØBENHAVN

Datastrukturer

Performance

Tech uden Tech –forskellige datastrukturer

- Vi skal se hvordan forskellige datastrukturer opfører sig ved opslag, søgning, indsættelse og fjernelse af elementer
- Bagefter skal vi teste performance

Array

Operation	Hvad sker der?
Find på index	Direkte opslag. Arravet ved præcis hvor det er. Som at slå op på side 47 i en bog.
Find et objekt	Vi kigger fra første plads og frem til vi finder det eller når enden.
Indsæt	Vi overskriver den værdi der allerede står på pladsen.
Fjern	Pladsen bliver tom, men den er stadig der. Værdien er nu null eller lignende.

ArrayList

Operation	Hvad sker der?
Find på index	Direkte opslag. Listen ved præcis hvor det er.
Find et objekt	Vi kigger fra første plads og frem til vi finder det.
Indsæt sidst	Den nye bare stiller sig bagerst. Ingen andre påvirkes.
Indsæt først eller midt i	Alle elementer der står efter det nye element rykker én plads tilbage.
Fjern først eller midt i	Alle der står efter det fjernede element rykker én plads frem.
Fjern sidst	Den bagerste går bare. Ingen andre påvirkes.

LinkedList

Operation	Hvad sker der?
Find på index	Listen starter forfra og tæller sig frem med ét skridt ad gangen.
Find et objekt	Vi starter forfra og kigger på hver enkelt, til vi finder det.
Indsæt først eller sidst	Den nye element kobles bare på og ingen andre flyttes.
Indsæt midt i	Listen skal først tælle sig frem til indexet, men selve indsættelsen er nem: koblinger omlægges.
Fjern først eller sidst	Koblingen omlægges og ingen andre flyttes.
Fjern midt i	Listen skal først tælle sig frem til indexet, men selve fjernelsen nem.

HashSet

Operation	Hvad sker der?
Find på index	Ikke muligt. HashSet kender ikke til rækkefølge eller numre.
Find et objekt	HashSet beregner hashkoden og finder elementet direkte. Lidt ligesom et indexopslag, bare uden at vi selv har valgt nummeret.
Indsæt	HashSet beregner hashkoden og placerer elementet det rigtige sted med det samme.
Fjern	HashSet beregner hashkoden, finder elementet direkte og fjerner det.

TreeSet

Operation	Hvad sker der?
Find på index	Ikke muligt. TreeSet kender ikke til index.
Find et objekt	TreeSet starter i roden og spørger ved hvert knudepunkt: er det større eller mindre? Søgefeltet halveres ved hvert trin.
Indsæt	TreeSet finder den rigtige plads i træet ved at sammenligne undervejs og placerer elementet der.
Fjern	TreeSet finder objektet som ved søgning og omstrukturerer grenene bagefter.

Test det af

- Kig på metoden `searchTimes()` og se hvordan man kan måle tid i java
- Lav nogle eksperimenter med henholdsvis `ArrayList` og `LinkedList` som du fylder med objekter
- Undersøg hvordan de to lister performer når du indsætter, sletter eller søger. Prøv med forskellige antal elementer i listerne
- Vær systematisk og skriv resultaterne op i en tabel fx i Excel

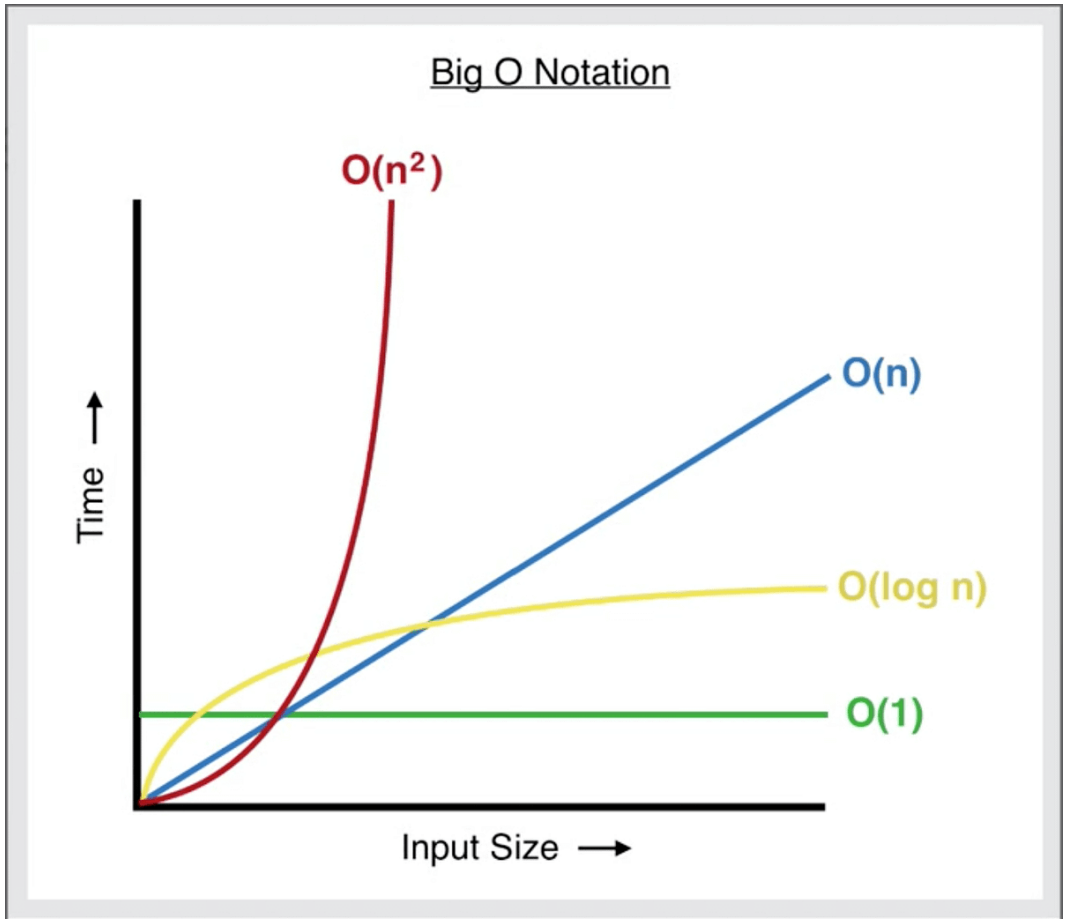
Test det af

- Lav nogle eksperimenter med henholdsvis `TreeSet` og `HashSet` som du fylder med objekter
- Undersøg hvordan de to set performer når du indsætter, sletter eller søger. Prøv med forskellige antal elementer i settene
- Vær systematisk og skriv resultaterne op i en tabel
- Sammenlign med resultaterne for lister

Opsamling

- Hvilken datastruktur skal vi bruge?
  - Kig først på egenskaber og vælg interface
  - Kig dernæst på hvad du skal bruge det til og vælge konkret klasse

O-notationen bruges til at angive kompleksitet



ArrayList og LinkedList

	ArrayList	LinkedList
Indsæt i slutning af liste (`add(element)`):	O(1)	O(1)
Indsæt et vilkårligt sted (`add(index, element)`)	O(n)	O(1)
Indsæt i starten af liste (`add(0, element)`)	O(n)	O(1)
Søgning (`contains(element)`)	O(n)	O(n)
Opslag på index (`get(index)`)	O(1)	O(n)
Fjerne et vilkårligt sted (`remove(index)`)	O(n)	O(n)
Fjerne første objekt (`remove(0)`)	O(n)	O(1)
Fjerne sidste objekt (`remove(list.size()-1)`)	O(1)	O(1)

Tree og Hash

	TreeSet/TreeMap	HashSet/HashMap
Indsæt (`add(element)`)	O(logn)	O(1)
Søge (`contains(element)`)	O(logn)	O(1)
Fjerne (`remove(element)`)	O(logn)	O(1)