Løsningsforslag teoriøving 2

Du måtte klare 12/19 oppgaver for å få øvingen godkjent. Det beste av to forsøk er tellende. Finner du feil, mangler eller forbedringer, ta gjerne kontakt!

Oppgave 1
Kø 1: I en kø vil man legge til elementer til slutten av lista. Så etter å la lagt til tallet 1 i køen vil lista se ut som $Q = \langle 2, 5, 12, 52, 1, 1\rangle$.

Oppgave 2
Kø 2: I en kø vil dequeue ta det første elementet i lista. Svaret vil da bli $Q = \langle 5, 12, 52, 1\rangle$.

Oppgave 3
Kø 3: For å kunne ha 1 først i lista, må man først dequeue 4 ganger. Så kan man kjøre enqueue 5 ganger for å legge til de resterende tallene. Svaret blir da at man må gjøre 9 opperasjoner.

Oppgave 4
Stakk 1: Push legger til i slutten av stakken, så etter å ha kjørt $Push(S, 1)$ vil stakken se ut som $\langle 2, 5, 12, 52, 1, 1\rangle$.

Oppgave 5
Stakk 2: Pop på en stack tar det siste elementet i lista. Her vil stakken se ut som $\langle 2, 5, 12, 52\rangle$.

Oppgave 6
Stakk 3: For å få tallet 1 som første element (nederst i stakken), må man først kjøre Pop 5 ganger for å få en tom stakk. Etter det kan man så pushe tallene som trengs tilbake, som tar 6 opperasjoner. Totalt vil det da trenges 11 opperasjoner.

Oppgave 7
Dobbel-lenket liste 1: List-Search modifiserer ikke listen, men bare returnerer noden som matcher det du søker etter. Listen vil da være uendret.

Oppgave 8
Dobbel-lenket liste 2: List-Insert vil putte inn en ny node, og sette denne som hodet i lista. En animasjon av dette kan bli sett i Hetland sine slides om datastrukturer.

Oppgave 9
Dobbel-lenket liste 3: List-Delete vil ta bort den første noden som matcher med nøkkelen du vil slette. Den vil så sy sammen nodene før og etter der du sletta. I dette tilfellet var den første noden som hadde nøkklen lik 2, så denne vil bli tatt bort.

Oppgave 10
Implementering av pekere og objekter 1: Hvis du skriver opp alternativene slik som vist i figur 10.5 i læreboka, og så følger strukturen, vil du få at alternativet $I = 7$, $N = \langle 3, 0, 4, /, 0, 1, 6, 0\rangle$,
$K = \langle 12, 0, 52, 1, 0, 5, 2, 0\rangle$, $P = \langle 6, 0, 1, 3, 0, 7, /, 0\rangle$ gir listen vi ønsker.

Oppgave 11
Implementering av pekere og objekter 2: Hvis du skriver opp alternativene slik som vist i figur 10.6 i læreboka, og så følger strukturen, vil du få at alternativet I = 4, A = $\langle 52, 13, 10, 2, 7, /, 5, 10, 4, 12, 1, 7, 1, /, 1\rangle$ gir listen vi ønsker.

Oppgave 12
Hashfunksjon 1: $x.key = m$, $h(m) = j$. Her er x elementet, m nøkkelen og j hashen.

Oppgave 13
Hashfunksjon 2: En kollisjon i hashtabeller betyr at 2 ulike nøkler blir hashet til samme verdi. Et eksempel her er at hashfunksjonen $h(k) = k \mod 10$ vil gi samme svar for $h(1)$ og $h(11)$.

Oppgave 14
Hashfunksjon 3: Hvis man ikke allerede vet dataene en hashfunksjon skal få, kan man ikke kunne garantere null kollisjoner selv om at antall nummere du skal hashe er mindre enn mulige hashverdier.

Oppgave 15
Hashfunksjon 4: En hashfunksjon som gir samme hash for alle mulige nøkler er ikke en god hashfunksjon, men kan brukes, bare du vil få veldig mange kollisjoner. De eneste i denne lista som ikke kan brukes som en hashfunksjon er den som baserer seg på sekundet nå, eller noe tilfeldighet. Her vil da samme nøkkel gi flere forskjellige hasher, som gjør at den da ikke kan brukes som en hashfunksjon.

Oppgave 16
Kjedet hashtabell 1: Tingen med denne oppgaven er at her sletter du basert på en node. I en dobbel lenket liste, vil du bare trenge å fikse referansen til node.prev og node.next. Dette vil ta O(1) arbeid. For en enkel-lenket liste, har den bare node.next feltet, så for å linke sammen noden før og noden etter, må du først søke gjennom hele listen etter noden x. Dette vil da ta O(n) tid.

Oppgave 17
Kjedet hashtabell 2: I worst-case har alle elementene samme hashverdi og vi har derfor en lenket liste med n elementer. Det tar O(n) tid å søke gjennom listen.

Oppgave 18
Amortisert analyse 1: Her kan du se på eksempelet regnet i forelesningen. For hver allokering gjør du gjøre 2^i arbeid, hver gang man øker må man også kopiere de forrige, som tar $\frac{2^i}{4}$ arbeid. Som så gir $\sum_{i=0}^{h-1} 2^i + \sum_{i=0}^{h} \frac{2^i}{4}$. Dette er lik $2^{h-1} + 2^h - \frac{5}{4}$ arbeid hver gang vi må allokere mer ressurser. Bytter vi ut $h = \log_2(n \cdot 4)$, får vi $2n + 4n - \frac{5}{4}$. Arbeidet per innsetting blir da $\frac{2n+4n-\frac{5}{4}}{n} = \Theta(1)$

Oppgave 19
Amortisert analyse 2: Noen ganger vil det være interessant å se på gjennomsnittlig kost av en funksjon. Dette er veldig relevant for tabell-insetting (som vist i forelesning). Så et eksempel er at en operasjon vil generelt ta O(1) tid, men så worst-case ta O(n) tid, bare at worst-case skjer i gjevne mellomrom, vil det å bare bruke worst-caset været for pessimistisk.