ADK - Konkordans: teoriuppgifter 2009-02-04

Name: _____

1. (a) **Bör positionerna lagras som text eller binärt?**

   Med avseende på storleken på indexinformationen kan man avväga för- och nackdelar. Om positionerna blir relativt stora kommer man spara plats på att lagra dem binärt. Filen var ca: 20MB stor. Ett tecken kräver 8 bitar, eller en byte, och vi behöver 8 bytes för att skriva 20 000 000. Att uttrycka 20M binärt kräver endast 4 bytes, som får plats i en 32-bitars int. Däremot blir det krångligare att hantera indexinformationen och felsöka om den lagras binärt, och med tanke på storleken på moderna diskar är lagringsutrymme kanske inte att prioritera.

   (b) **Bör indexinformationen lagras tillsammans med själva ordet eller på ett separat ställe?**

   Om man lagrar indexinformationen tillsammans med orden så underlättar man hanteringen. Dessutom slipper man att söka efter indexinformationen annorstädes.

Datastruktur	Snabbhet	Utrymme skivminne	Utrymme primärminne	Enkelhet
Binärt sökträd	Sökning är effektivt om trädet är sorterat, O(log n). Blir långsamt om trädet är högt	Ganska stort eftersom vi måste ha pekare vid varje nod till barnen.	Eftersom sökningen ska utföras på skivminnet, kommer ingen väsentlig mängd primärminne användas	Jobbigt att balansera trädet. Borde inte vara så svår att konstruera dock.
Sorterad Array	Sökning är effektivt med binärsökning, tidskomplexiteten O(log n).	Tar ytterst lite plats eftersom vi inte behöver spara mer information än ordet och dess index.	Eftersom sökningen ska utföras på sknetivmin, kommer ingen väsentlig mängd primärminne användas	Enkelt att generera och lagra på fil. I princip en lista med ord och index.
Hashtabell	Om hashfunktionen är bra får vi en mycket effektiv sökning, tidskomplexitet O(1). Däremot kommer det ta tid att läsa in hashtabellen i minnet.	Tar ungefär lika mycket skivutrymme som en sorterad array, här med ett hashvärde i stället för index.	Eftersom sökningen ska utföras på skivminnet, kommer ingen väsentlig mängd primärminne användas	Tar tid att generera. Borde inte vara svårt att lagra. Kan bli komplicerat om man skriver hashfunktionen.
Trie	Borde vara ineffektiv eftersom det blir många sökningar på varje bokstav. Att hitta en nyckel har tidskomplexiteten O(n), där n är längden på ordet.	Torde bli stor eftersom vi lagrar alla bokstäver i ett ord för sig.	Eftersom sökningen ska utföras på filen, kommer ingen väsentlig mängd primärminne användas.	Blir knepig att lagra med alla bokstäver och pekare.
Latmanshashning	Vi söker på konstant tid fram ett index som är en del av ordet vi söker. Därefter utför vi binärsökning på en mycket liten delmängd av alla ord.	Tar ungefär lika mycket skivutrymme som en sorterad array, här med ett hashvärde i stället för index.	Se ovan, men man borde kunna läsa in prefixinformationen (aaa, aab osv) i minnet för att på konstant tid kunna ta reda på ett intervall mellan två prefix.	Tar tid att generera. Borde inte vara svårt att lagra. Kan bli komplicerat om man skriver hashfunktionen.