Laboration 4

SORTERING

OBJEKTBASERAD PROGRAMMERING I C++
Ver 11

Mål: Laborationen avser att belysa olika sorteringsalgoritmers effektivitet

Redovisning: Koden skall vara lättläst där indragningar, kommentarer och variabelnamn är logiska och konsekventa.

I denna laboration ska du skriva en laborationsrapport. Låt de delar som enligt tidigare laborationer, ska ingå i laborationsbeskrivningen vara en första del av rapporten. Sist i detta dokument beskrivs vad som ska ingå i rapporten.

Skicka in källkod och laborationsrapport via Moodle.
Zippa källkod och rapport i samma fil.

Regler för inlämning:

Genom att du lämnar in detta arbete försäkrar du att alla svar är skapade av dig själv. Du är även ansvarig att se till att det inte finns någon plagierad text i dokumentet. När du refererar och citerar andra verk måste korrekta källhänvisningar finnas och i fallet citering ska den citerade texten vara tydligt markerad. http://www.bib.miun.se/student/skriva/referenser

Om plagierad text finns i dokumentet riskerar du att stängas av från studier.
Om samarbete sker utan att detta har stöd i instruktionen för examinationen utgör det normalt en disciplinförseelse och du som student riskerar att stängas av från dina studier.

Uppgift Sorteringsalgoritmers effektivitet

Uppgift

Du ska genom praktiska tester undersöka fyra sorteringsalgoritmers effektivitet. Genom att mäta tider för sortering av arrayer med olika antal element ska du undersöka hur väl följande angivelser av algoritmernas komplexitet stämmer:

- quicksort är av typen $O(n*\log_2(n))$
- insertionsort är av typen $O(n^2)$
- selectionsort är av typen $O(n^2)$
- bubblesort är av typen $O(n^2)$

O(n) är ett skrivsätt som anger storleken på ansträngningen som algoritmen behöver för att utföra sorteringen. I detta fall är ansträngningen proportionell mot antalet (n) element. O brukar uttalas Big O och kan också översättas med "order of magnitude". Se vidare i läroboken avsnittet om sökning och sortering.

Specifikation av testerna

Antal element

Samtliga algoritmer ska testas med arraystorlekarna 5 000, 10 000, 15 000, 20 000, 25 000, 30 000, 35 000 och 40 000 element.

Mätning av tid

Det visar sig med dagens snabba datorer att sorteringen går för snabbt för att tidmätningen ska bli tillförlitlig vid sortering av heltal och man mäter tiden med standardfunktionen clock().

Vi tillhandahåller därför en klass för tidmätning som kan mäta tider ner till en mikrosekund, se bifogad fil. Bifogat finns också ett testprogram som visar hur klassen används.

Klassen fungerar både i Windowsmiljö och i Linuxmiljö.

Arrayklass

Skapa en arrayklass för att hantera arrayerna som används vid sorteringen. Utgå från arrayklassen i lektion 3 "Överlagring av operatorer", avsnitt 3.17 "En arrayklass". I klassen skapas arrayen dynamiskt med önskad storlek. Destruktorn frigör allokerat minnesutrymme. Lägg till medlemsfunktioner för:

- slumpning av tal till arrayen
- sortering med quicksort
- sortering med insertionsort
- sortering med selectionsort
- sortering med bubblesort

Tester av sorteringsalgoritmerna

Slumpa tal till arrayen, starta tidtagningen, sortera med respektive sorteringsalgoritm och stoppa tiden. Gör 10 mätningar för varje arraystorlek och slumpa nya tal till arrayen för varje mätning. Redovisa medeltalet av de 10 mätningarna.

Slumptalen i arrayen ska ligga i intervallet 0..arraystorlek-1.

Program

Skriv ett program som utför testerna enligt ovan. Resultatet av testerna ska skrivas på skärmen och på fil. Filen ska du sedan använda för att i Excel (eller något annat lämpligt program) rita diagram som visar hur testerna stämmer med de förväntade teoretiska BigO–resultaten. Rita diagram över testernas utfall **och** över de teoretiska värdena. Kurvan för de teoretiska värdena ska vara med i samma diagram som testernas utfall!

Följande diagram ska finnas med:

- Ett diagram för varje sorteringsalgoritm (inklusive den teoretiska kurvan)
- Ett diagram i vilket alla fyra sorteringsalgoritmerna ingår, så att man tydligt kan jämföra utfallen.

Totalt blir det 5 diagram. Kom ihåg att gradera axlarna!

På skärmen

Skriv ut resultat på skärmen eftersom de produceras av programmet. Gör en snygg layout i vilken man ser vilken sorteringsalgoritm som jobbar, arraystorlek och sedan dess sorteringstid.

På fil

Spara resultatet av testerna på en textfil. Använd följande format så fungerar det bra att öppna textfilen i Excel:

algoritmens namn <tabb> arraystorlek <tabb> tid.

<tabb> fås med chr(9).

Redovisning

Laborationen redovisas med källkod och rapport. I rapporten ska förutom en laborationsbeskrivning, resultatet av testerna ingå. Använd tabeller och diagram för att presentera resultaten. Lägg in diagram från Excel (motsvarande) i rapporten. Kommentera resultaten! Hur väl stämmer testresultaten med teorin?