Innlevering 1

Kladd for plan skriving av il1

Lage Teque, både psuedokode og python/java program for Tequq, merk at oppgaven er hentet fra Kattis

Oppgaveoversikt:

**1a)** Skriv pseudokode for hver av operasjonene

• push\_back

• push\_front

• push\_middle

• get

**1b)** Skriv et Java eller Python-program som leser input fra stdin og printer

output nøyaktig slik som beskrevet ovenfor. Vi stiller ingen strenge krav

til kjøretid.

**1c)** Oppgi en verste-tilfelle kjøretidsanalyse av samtlige operasjoner med Onotasjon. I analysen fjerner vi begrensningen på N, altså kan N være vilkårlig stor

**1d)** Hvis vi vet at N er begrenset, hvordan påvirker det kompleksiteten i Onotasjon? Formulert annerledes: Hvorfor er det viktig at vi fjerner begrensningen på N i forrige deloppgave? (Hint: 106 er en konstant).

**Oppgave 2 – sortering**

I denne oppgaven skal du implementere flere sorteringsalgoritmer!

Disse to sorteringsalgoritmene må implementeres:

• Insertion sort

• Merge sort

Du kan implementere flere sorteringsalgoritmer dersom du ønsker.

**2a)**

(som æ forsto det) skal man lage et program som tar input som er gitt i oppgaven og implementere minst to algoritmer(gitt over) som skal sortere denne inputen og gi output i en ny fil som har akkurat samme format som input filen.

**2b)**

Du skal redgjøre for hvilke sorteringsalgoritmer som er implementert, og hva som er testet for å sjekke at alle algoritmene gir rimelige svar.

**Oppgave 3 – sammenligninger, bytter og tid**

**3a)**

Hver av sorteringsalgoritmene skal kunne telle antall sammenligninger og antall bytter som gjøres i løpet av en sortering. Det kan være lurt å skrive egne metoder for sammenligning og bytter, slik at det blir enkelt å telle. I tillegg skal den totale tiden brukt på sortering oppgis i mikrosekunder

Du skal kun produsere én outputfil for en gitt inputfil, som en CSV-fil. Dersom

inputfilen heter example, skal outputfilen hete example\_results.csv.

Hvis inputfilen inneholder n heltall, skal du gjøre n+1 sorteringer for hver sorteringsalgoritme du tester. Første sortering gjøres på det tomme arrayet, andre

sortering gjøres på et array som inneholder det første elementet, tredje sortering gjøres på et array som inneholder de to første elementene, og så videre,

helt opp til n.

Den første linjen i output skal se slik ut:

n, alg1\_cmp, alg1\_swaps, alg1\_time, alg2\_cmp, alg2\_swaps, alg2\_time

Det samme mønsteret skal følges for alle algoritmene du tester. De etterfølgende linjene skal bestå av kommaseparerte heltall.

• Første tall skal angi hvor mange elementer som ble sortert

• Andre tall skal angi hvor mange sammenligninger alg1 brukte

• Tredje tall skal angi hvor mange bytter alg1 gjorde

• Fjerde tall skal angi hvor lang tid alg1 brukte (i mikrosekunder)

• Femte tall skal angi hvor mange sammenligninger alg2 brukte

• ...

Det er ikke forventet at deres algoritmer viser nøyaktig samme tall. Tallene

er både avhengig av implementasjonen av sorteringsalgoritmen, og hvordan

bytter og sammenligninger måles.

**3b)**

Det skal skrives kort om hvordan bytter og sammenligninger er målt.

**Oppgave 4 – Eksperimenter**

Her skal du kun bruke programmet fra forrige deloppgave for å gjøre noen refleksjoner.

Vi ønsker at du gjør noen egne refleksjoner. Du må ta stilling til disse spørsmålene:

• I hvilken grad stemmer kjøretiden overens med kjøretidsanalysene (store

O) for de ulike algoritmene?

• Hvordan er antall sammenligninger og antall bytter korrelert med kjøretiden?

• Hvilke sorteringsalgoritmer utmerker seg positivt når n er veldig liten?

Og når n er veldig stor?

• Hvilke sorteringsalgoritmer utmerker seg positivt for de ulike inputfilene?

• Har du noen overraskende funn å rapportere?

Alle svar må begrunnes. Legg gjerne ved skjermbilder eller lignende som gir grunnlag for påstandendene dine. Vær tydelig på hvordan eksperimentet ble utført, slik at det lar seg reprodusere av andre.

Siste to do:

Gjør oppgave 1c) og 1d)

Finn ut hvordan merge swaps skal forklares til retter

Forklare hvordan bytter og samlingene er målt

«Det skal skrives kort om hvordan bytter og sammenligninger er målt.»

Eksperimenter: tips lag excel ark og plott data inn der.

Se sammenhenger og dokumentere de

Sammenligning, bytter og tid heng sammen

Hvor effektiv en algoritme er

Gjør egne refleksjoner under eksperiment

Gjøre rede for disse spørsmålene:

I hvilken grad stemmer kjøretiden overens med kjøretidsanalysene (store

O) for de ulike algoritmene?

• Hvordan er antall sammenligninger og antall bytter korrelert med kjøretiden?

• Hvilke sorteringsalgoritmer utmerker seg positivt når n er veldig liten?

Og når n er veldig stor?

• Hvilke sorteringsalgoritmer utmerker seg positivt for de ulike inputfilene?

• Har du noen overraskende funn å rapportere?