INF-1400, eksempel på eksamensoppgave.

2010-05-03

Dette er et utdrag (del A) av eksamen i INF1010 Objektorientert programmering ved Universitetet i Oslo (2009-06-03). Oppgaven er noe omskrevet for å passe litt mer til kurset i Tromsø, og er ment å være et eksempel på en mulig deloppgave på eksamen.

Den originale oppgaven ser ut til å være basert på Java (Java er nevnt i oppgavesettet), og er lagt opp til å vare i 3 timer.

Et program for å finne beste sammensetning av et stafettlag

I denne oppgaven skal du skrive deler av et program som skal brukes for å finne hvilke utøvere (lagsammensetning eller stafettlag) som skal settes på de ulike etappene i en stafett. Problemet som skal løses, er å finne det beste laget for stafetten ved å velge en utøver til hver etappe slik at lagets totaltid blir minst mulig. En stafett har et bestemt antall etapper. Alle utøvere prøver en eller flere etapper, og den tiden de bruker på etappen noteres (i objekter av klassen Etappetid, se nedenfor). En utøver kan bare brukes til en etappe, og kun til etapper der det finnes etappetider for utøveren. I del A løses oppgaven ved hjelp av en metode som prøver alle mulige lagsammensetninger. I del B gjøres det samme ved hjelp av tråder. (Det finnes lurere måter å gjøre dette på, men det bryr vi oss ikke om i denne oppgaven.)

Del A (utgjorde totalt 60%)

Vi definerer fire klasser:

- Et objekt av klassen **Utøver** inneholder navnet til en utøver.
- Et objekt av klassen **Etappetid** har en referanse til en utøver og en etappetid for denne utøveren på den etappen som objektet tilhører.
- Et objekt av klassen **Etappe** inneholder et etappenummer og en liste med etappetider for denne etappen.
- Et objekt av klassen **Stafett** har en liste med alle etappene i stafetten, sortert etter etappenummer (dere trenger ikke sørge for at denne blir sortert - vi antar at den er sortert).

Oppgave 1 - Tegning av datastrukturen (vekt 5%)

Tegn datastrukturen til et stafettobjekt med 5 etapper, og for hver etappe et antall etappetidobjekter. Tegn også noen utøverobjekter, og gjør det klart hvordan disse er forbundet med de andre objektene i figuren. Legg inn en utøver som går igjen i minst 2 etapper.

Oppgave 2 - Klassedefinisjonene (vekt 5%)

Skriv programkoden som definerer de fire klassene **Stafett**, **Etappe**, **Etappetid** og **Utøver** med variablene du trenger. Metoder kommer i oppgave 3-6 og trenger ikke å være med nå.

Bakgrunn for oppgave 3-6

Metoden **finnBesteStafettlag** skal fungere som hovedmetode ved å kalle på metoden **stafettid**. **stafettid** genererer alle mulige sammensetninger av lag (ved rekursivt å kalle på seg selv i neste etappe). For hver gang **stafettid** har et forslag til et nytt lag (alle etapper har fått en utøver og en tid) kalles metoden **nyttMuligStafettlag** som sørger for å bevare det hittil beste laget med tilhørende totaltid. Til slutt skriver **finnBesteStafettlag** ut resultatet.

stafettid lager forslag til stafettlag ved (rekursivt) å prøve alle mulige kombinasjoner av utøvere for etappene. Metoden har to parametre. Den første parameteren er lagkombinasjonen dere bygger opp nå. Denne bør angi etappetidene og utøver som er valgt ut for hver etappe samt totaltiden for etappene som er valgt ut til nå. Den andre parameteren kan være en liste over etappene som det gjenstår å plassere utøvere i.

Anta at **stafettid** er kalt etter å ha plassert de første tre etappene. Første parameter angir et stafettlag med utøvere plassert på etappe 1-3, samt totaltiden til de plasserte utøverne. Den andre parameteren angir at etappe 4 og 5 er igjen (f.eks. som en liste med de to etappeobjektene). Metoden skal etter tur velge alle utøvere fra listen over etappetidobjekter for etappe 4, legge dem til laget og kalle **stafettid** med det nye laget samt at etappe 5 er igjen. Når **stafettid** blir kalt med en tom liste over gjenstående lag skal den kalle metoden **nyttMuligStafettlag** som sørger for å ta vare på totaltiden og utøverlisten hvis dette er beste tid til nå.

Hvordan datastrukturen initialiseres (input), skal ikke være med i programmene du lager. Du trenger ikke lage «set»- eller «get»-metoder i klassene.

Oppgave 3 - Datastruktur for valgte utøvere (vekt 5%)

I programmet må det være en datastruktur for å holde orden på utøvere som er med i (deler av) et stafettlag og som ikke kan brukes i senere etapper. Hittil beste lagsammensetning kan tas vare på med den samme strukturen. Beskriv kort (med ord, ikke kode) hvordan du velger å ta vare på denne informasjonen. Beskriv metodene du bruker til dette ved å forklare hvordan de virker og ved å beskrive parametre og eventuell returverdi.

Oppgave 4 Metoden stafettid (vekt 30%)

Skriv den rekursive metoden **stafettid** definert i klassen **Stafett**.

Oppgave 5 Metoden nyttMuligStafettlag (vekt 10%)

Skriv metoden **nyttMuligStafettlag** i klassen **Stafett. nyttMuligStafettlag** blir kalt (i **stafettid**) når en ny kombinasjon av utøvere for alle etappene er funnet.

Oppgave 6 Metoden finnBesteStafettlag (vekt 5%)

Skriv metoden **finnBesteStafettlag**. Metoden skal til slutt skrive ut navnene på utøverne på det beste laget i «etappeorden», samt den tilhørende stafettiden (totaltid).

I del B skulle de løse noen av de samme problemene, men ved hjelp av tråder. Dette har vi ikke gjort noe med i INF-1400, så dette er ikke tatt med her.