Oppgave 4 Administrasjon av obligatoriske oppgaver (50 poeng)

Universitetet i Ruritania skal digitalisere sin håndtering av studentenes obligatoriske oppgaver («obliger»), og du skal lage deler av en pilot med et objektorientert design basert på innkapsling, dvs. instansvariabler og non-public metoder (navn starter med **_**) skal kun benyttes i den klassen de er definert. **Det er i mange av deloppgavene nødvendig å bruke metoder fra andre klasser/ deloppgaver. Du kan bruke disse selv om du ikke har skrevet dem.**

Emnene har ulike antall obligatoriske oppgaver som kreves godkjent, og hver oblig i et emne må registreres manuelt hvert semester. I Ruritania må en student ha alle oppgaver gitt i emnet godkjent for å få gå opp til eksamen, men i denne piloten foretas det ikke kontroll av om studenten har fått godkjent forrige oblig før godkjenning av besvarelse for neste oblig. Foruten obliger har et emne registrerte studenter og ansatte rettere. Både rettere og studenter har unike brukernavn.

Innleveringen av besvarelser for en oblig foregår i et annet system som du ikke skal lage. Innleverings-systemet lager en fil for hver oblig, med et entydig navn etterfulgt av «.txt», for eksempel *oblig1.txt*. Filen inneholder en linje per student på følgende format (eksempel med to linjer):
student1 obl1student1.txt
student2

I eksempelet er *student1* et entydig brukernavn og *obl1student1.txt* er navnet på filen der student1s besvarelse ligger. s*tudent2* i eksempelet har ikke levert noen besvarelse på oblig 1.

Hver besvarelse skal vurderes av en retter for emnet. Når fristen for en obligatorisk oppgave er utløpt, fordeles besvarelser på retterne, som vurderer en og en til godkjent (1) eller underkjent (0).

Piloten har funksjonalitet for hente ut en eksamensliste med brukernavn på alle studenter som er kvalifisert for eksamen – dvs har godkjent alle registrerte obliger i emnet.

Oppgave 4a) 5 poeng

Definer en klasse **Emne** (inkludert konstruktør) som administrerer obliger, rettere og studenter i et emne. Konstruktørens parametere gir verdier til emnekode (streng), registrerte studenter (ordbok med Student-objekter) og rettere (liste med Retter-objekter).

Oppgave 4b) 5 poeng

Skriv metoden **administrer** i klassen Emne. Metoden skriver ut emnekoden for emnet og en meny på terminalen før den ber om kommando. Den tar imot følgende lovlige kommandoer fra en bruker:
O: Ny oblig
F: Frist ute, start retting
L: Lag eksamensliste
A: Avslutt

Annen input skal gi feilmelding og nytt forsøk. En kommando bør gjenkjennes selv om det er blanke foran eller bak, og uansett om brukeren skriver liten eller stor bokstav (metodene **strip** og **upper** er nyttige her). For å utføre kommandoene skal metoden kalle på følgende non-public metoder som også ligger i klassen **Emne**:
O -> **_opprettOblig**
F -> **_startRetting**
L -> **_skrivEksamensListe**
Disse metodene skal skrives i senere deloppgaver. De opererer kun på instansvariabler i klassen, og har ingen parametere (annet enn **self**) eller returverdier.

Oppgave 4c) 10 poeng

Skriv en klasse **Student** med instansvariabler for brukernavn og fullt navn (begge får verdi fra parametere til konstruktøren) og en ordbok som skal ta vare på resultater av rettinger, der obligId er nøkkel og resultatet verdien. Klassen har følgende metoder i tillegg til konstruktør:

registrer med parametere **oblig** (en streng som identifiserer en oblig entydig) og **resultat** (et heltall som angir om en besvarelse er vurdert som godkjent). Metoden registrerer hvilket resultat en student har fått på en oblig.

altGodkjent med parameter **antObliger** som angir antall obliger registrert i emnet. Metoden returnerer en boolsk verdi: **True** hvis studenten har fått alle obliger godkjent, **False** om en eller flere obliger mangler eller har et annet resultat enn godkjent (kodet som heltallet 1).

Oppgave 4d) 2 poeng

Skriv klassen **Retter** med instansvariabel retterens brukernavn (parameter til konstruktøren) og metoden:

vurder med en parameter som angir et filnavn for en besvarelse. I denne foreløpige utgaven av piloten ignorerer programmet ditt filen med besvarelsen, og returnerer alltid heltallet 1 (godkjent).

Oppgave 4e) 8 poeng

Skriv en klasse **Oblig** med tre instansvariabler: Entydig id for obligen, en leveringsfrist på formen ååmmdd (f. eks. 190906 for 6. september 2019) og om obligen er rettet eller ikke. De to første initialiseres med parametere til konstruktøren. Skriv følgende metoder i klassen Oblig:

klarForRetting med parameter som angir dagens dato. Metoden returnerer **True** hvis fristen er før dagens dato og besvarelsene for obligen ikke allerede er rettet – ellers **False**.

hentBesvarelser leser en fil med oversikt over studenters besvarelser (navn og format på denne filen er beskrevet i oppgave-innledningen) og returnerer en ordbok der en students brukernavn er nøkkel og filnavn med studentens besvarelse er verdi.

Oppgave 4f) 5 poeng

Utvid klassen **Oblig** med en ny metode

fordelRetting tar som parametere en ordbok med studenters besvarelser og en liste med rettere, og lar retterne vurdere én og én besvarelse (bruk metoden **vurder** i klassen **Retter**) slik at alle retterne ideelt sett vurderer like mange. Altså skal ingen retter vurdere mer enn 1 besvarelse mer enn de andre. Resultatene av retternes vurderinger samles i en ordbok der nøkkelverdiene er studentenes brukernavn. Metoden markerer obligen som rettet og returnerer ordboken med resultater.

Oppgave 4g) 15 poeng

Skriv følgende non-public metoder i klassen **Emne**:

_opprettOblig genererer et unikt oblignavn, og ber bruker på terminalen om å oppgi frist på formen ååmmdd. Deretter legges en ny oblig til emnet.

_startRetting ber om dagens dato og sjekker om noen av obligene i emnet har en utløpt frist uten at besvarelsene er rettet. I så fall henter den alle besvarelser for obligen, fordeler dem på rettere og registrerer resultatet av rettingen i Student-objekter (på riktig oblig) for studenter som har levert.

_skrivEksamensListe bygger opp en liste med brukernavn for alle studenter som har fått godkjent alle obliger registrert for emnet, og skriver til sist ut denne på terminal med en passende overskrift.