Uke 13:
Prøveeksamen og eksamensinfo

Siri Moe Jensen

Geir Kjetil Sandve

Dagens forelesning

- Hva skjer fremover?
- Informasjon om eksamen
- Gjennomgang av (deler av) prøveeksamen

IN1000 tilbud frem til eksamen (1.12)

- Undervisning - se beskjed på semestersiden
  - prøveeksamen fra i morgen til 29.11
  - repetisjonskurs og lab (nye tider) fra i morgen, se semestersiden
  - Pragmatisk eksamensforberedelse 24.11; kl 14-18
    Detaljer: Beskjed kommer ila uken på semestersiden

- Andre ressurser
  - pensumbok
  - lysark og opptak fra forelesningene
  - innleveringsoppgaver (og tilbakemeldinger)
  - tidligere eksamensoppgaver med løsningsforslag (lenke fra semestersiden)
  - Trix-oppgaver

Formell informasjon om eksamen

- Om IN1000:
  - Emnesiden: Eksamen
  - Semestersiden: Eksamen: Tid og sted

- Eksamenssettet i Inspera:
  - Informasjon (første side) – les denne!
  - Kandidatinstruks (vedlegg)

- Teknisk/ praktisk
  - Eksamen løses i Inspera, uten nett, på Windows PC med norsk tastatur
  - Du kan veksle mellom norsk og engelsk oppgavetekst (øverst til høyre)

Tips til eksamen: Vis at du kan pensum!

- Poeng sier noe om vektingen av (del)oppgavene. Ikke bruk uforholdsmessig mye tid på oppgaver med få poeng.

- Vi kjører ikke programmene deres!
- Svar på det det spørres om. Ikke gjenta oppgaven.
- Bruk navn som oppgis i oppgaven (variabler, klasser, ...)
- Norske bokstaver er ok.
- Vi trekker ikke for opplagte skrivefeil, men gjennomgående feil kan gi inntrykk av manglende forståelse/ kunnskaper. *Lesbar kode* kan bidra til at sensor forstår hva du mener.

- Vi krever ikke kommentarer på eksamen. Kan brukes om du ønsker å klargjøre noe i løsningsvalget.
- Er du usikker på tolkning av oppgaven, vurder alternativer og gjør et valg. Kommenter om du er usikker.

- Du trenger ikke teste input og håndtere feilsituasjoner om ikke oppgaven ber om det.
- Vi vurderer det du viser at du kan fra IN1000 pensum. Bruk fortrinnsvis python som undervist.

Etter eksamen

Vurdering

- Poengene viser hvordan den enkelte oppgave vektes
- Bokstavkarakterer (A-F, der F er ikke bestått)
- Oppgaven, løsningsforslag og sensorveiledning legges ut på semestersiden senest ved sensurtidspunkt (frist 22. desember)
- Alle vil dessuten få tilsendt en tekstlig, automatisk generert tilbakemelding (uio-epost)

Ny/ utsatt eksamen ("konte")

- arrangeres ca midten av januar
- påmeldte får beskjed om dag etter nyttår

Prøveeksamen del 1-3

Prøveeksamen del 4

Kandidat ID
? gjenstår

Oppgave 4. Reservasjonssystem (50 poeng)

I denne oppgaven skal du skrive en første versjon av et program for reservasjon av rom på ett eller flere hoteller. I denne versjonen håndterer programmet kun reservasjoner for kommende natt. Programmet består av fire klasser du skal skrive (du trenger ikke skrive noe hovedprogram):

Reservasjonssystem. Systemet håndterer ett eller flere hoteller og støtter to typer reservasjoner:

- Reservasjon av et rom med kjent rom-nummer på et navngitt hotell
- Gruppereservasjoner der reservasjonssystemet selv finner og reserverer *passende rom*[1] til så mange som mulig i gruppen, uavhengig av hotell.

Gruppe. En gruppe består av en eller flere personer som ønsker å bestille sammen, og som har felles krav til fasiliteter på rommet/ rommene de ser etter. Personer i en gruppe bryr seg ikke om hvem de blir plassert på rom med så lenge de er i samme gruppe.

Hotell. Et hotell har et navn og en samling gjesterom av klassen **Rom**.

Rom. Hvert rom har et romnummer og et antall senger. Rommet har null eller flere fasiliteter (for eksempel bad eller utsikt). Om rommet er utleid har det en eller flere gjester.

Du velger selv hensiktsmessige instansvariabler utover de som er oppgitt. Instansvariabler skal ikke aksesseres utenfra klassen (de er non-public). Personer representeres kun med navn (en streng som kan inneholde blanke). Fasiliteter og krav til fasiliteter representeres med en tekstlig beskrivelse (en streng uten blanke, for eksempel "utsikt" eller "peis").

Husk å bruke metoder i andre klasser der dette er hensiktsmessig. Du kan anta at returverdier og argumenter til metoder er gyldige, og trenger ikke fange opp eller sjekke for feilsituasjoner.

Oppgave 4a) (3 poeng)

Skriv klassen **Gruppe**. Konstruktøren skal ha én parameter; en liste med krav til fasiliteter i rom. Klassen skal tilby følgende metoder i tillegg til konstruktør:

- **legg_til_personer().** Parameter er en liste med personnavn som legges til gruppen (i tillegg til de som eventuelt var med i gruppen fra før)
- **hent_personer().** Returnerer navnene i gruppen
- **hent_krav()**. Returnerer fasilitetene gruppen krever i sine rom.

Oppgave 4b) (8 poeng)

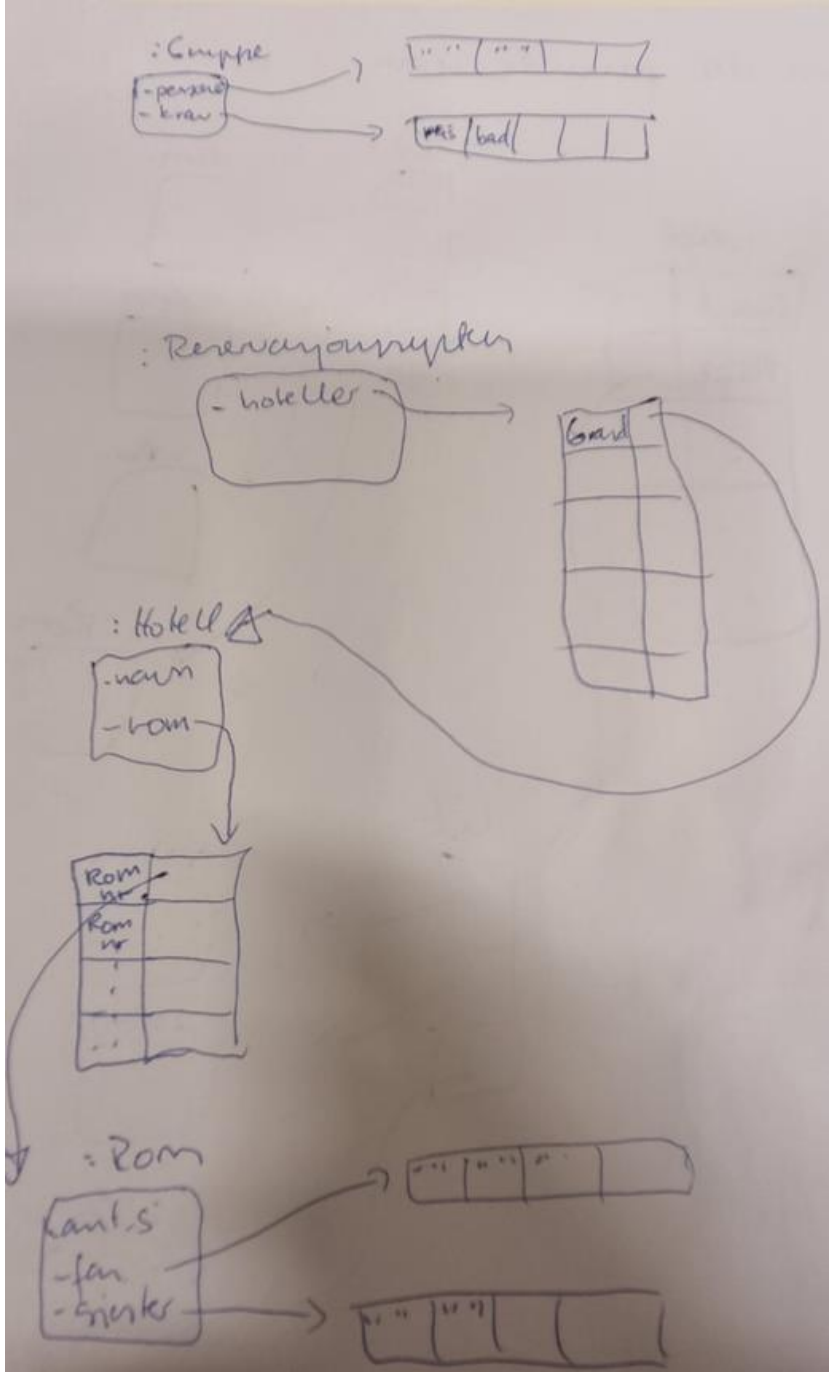
Skriv klassen **Rom**. Romnummer, antall senger og fasiliteter er parametere til konstruktøren. Klassen skal tilby følgende metoder i tillegg til konstruktør:

- **reserver().** Metoden har en parameter som angir navn på en eller flere gjester som skal bo på rommet. Metoden registrerer at rommet er opptatt og navn på gjesten(e).
- **hent_ant_senger().** Returnerer antall senger i rommet.
- **passer().** Metoden har en parameter for krav til fasiliteter og returnerer True hvis rommet *passer*, ellers False. Metoden tar ikke hensyn til antall senger.
- **__str__().** Returnerer en streng med romnummer og antall senger, samt fasiliteter i rommet.

---

[1] *passende rom er i denne oppgaven definert som ledige rom som har alle fasiliteter som kreves av en gruppe*

Skisser oppgave 4a-e)



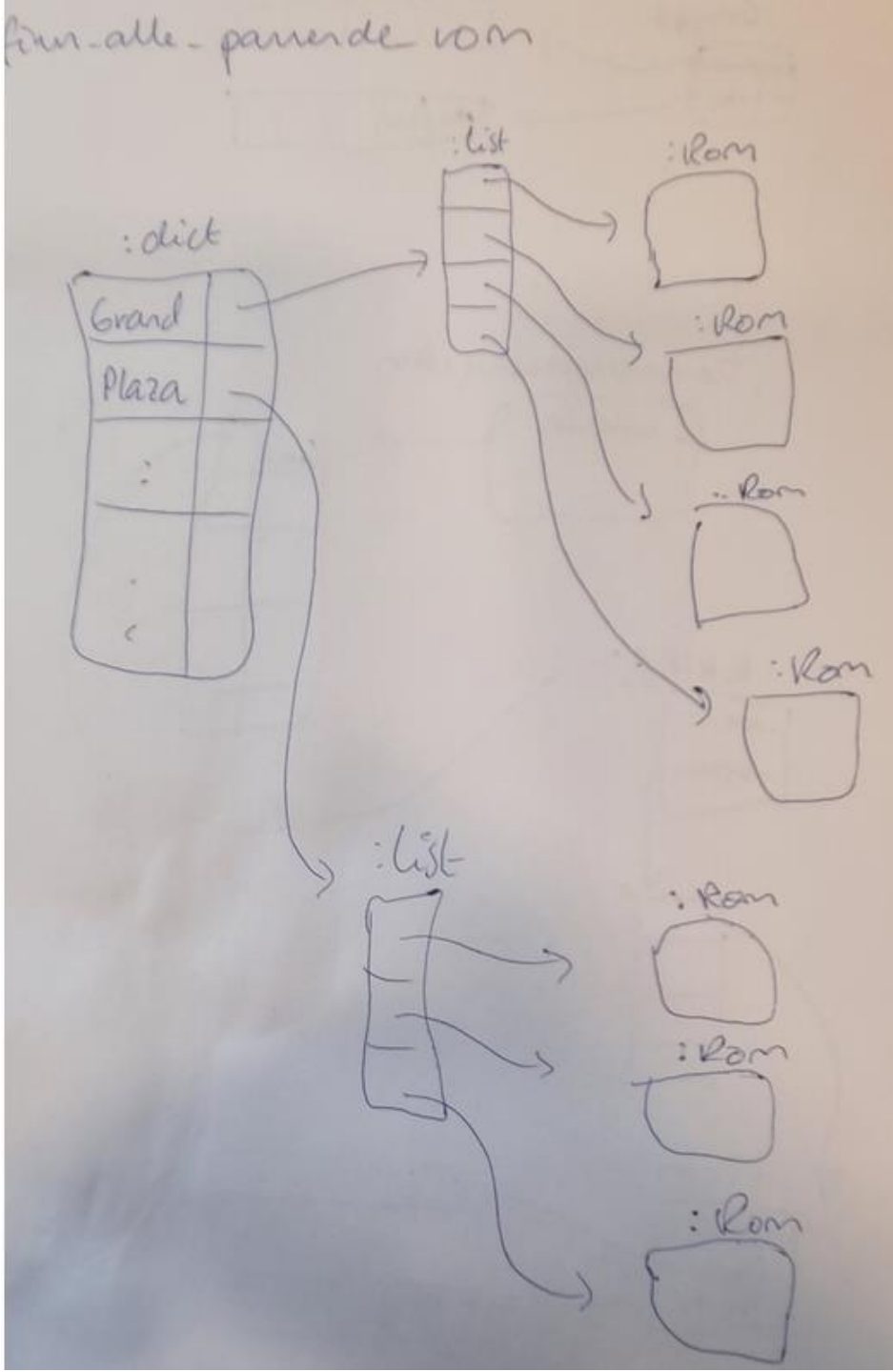
Skisse oppgave 4e)

Metoden har en parameter for fasiliteter som kreves og

returnerer en ordbok med alle passende rom i alle hoteller.

Ordboken har hotellnavn som nøkler og en liste med ledige, passende Rom-objekter som verdi for hver nøkkel.

Hoteller uten passende rom skal ikke være med i ordboken



Skisse oppgave 4f) Gruppereservasjon

gruppe_reservasjon().
Metoden har en parameter for gruppen det skal reserveres rom til, og

returnerer en bekreftelse på resultatet.

Bekreftelsen er en liste med en eller flere strenger

