Hotell

Sensurveiledning (med oppgaveteksten)

- *Skrivefeil ignoreres vanligvis, men feil som kan tyde på manglende forståelse (for eksempel kopiering av kode som ikke passer inn eller som ikke svarer på oppgaven) vil gi trekk. Norske tegn er helt OK*
- *Gjennomgående litt større feil/ mangler vil det bli trukket for (selv om de er små hver for seg)*
- *Eksempler på feil/unøyaktigheter som ignoreres:*
  - *Mangler (noen få) ';' , '}' eller '{' om koden ellers gir mening*
  - *Småfeil i (gjenkjennbare) navn*
  - *Import-setninger mangler*
- *Det legges lite vekt på riktige aksessmodifikatorer (public og private)*

Du har søkt om jobb i dataavdelingen til en stor hotellkjede, og som en del av jobbintervjuet blir du bedt om å lage noen programdeler. Hotellkjedens datasystemer blir programmert etter gode objektorienterte prinsipper, slik som i kurset IN1010 på Universitetet i Oslo, og du skal vise at du behersker slik programmering. Det du skal programmere utgjør ikke et fullstendig program. Husk derfor å se nøye etter hva oppgavene ber om og programmer bare dette. Svar utover det som det blir bedt om, vil ikke telle med i beregningen av karakteren din. Hvis noe er uklart i oppgaven, så gjør, og skriv ned i besvarelsen din, dine egne fornuftige forutsetninger.

Alle hotellrom er subklasser av klassen `Rom` men det skal ikke finnes objekter av klassen `Rom`. Det er tre typer rom: enkle rom, vanlige rom og suiter.

I tillegg har noen vanlige rom og noen suiter kjøkken, og da må du vite antall kvadratmeter på dette kjøkkenet (et heltall). Ingen enkle rom har kjøkken. Implementer det å ha kjøkken som et interface.

Oppgave 1. 5 poeng
Tegn opp klassehierarkiet. Ta med interfacet.

Læringsmål oppgave 1.
Her skal du vise at du vet hva et klassehierarki er og at du kan tegne opp et slikt. I tillegg skal du vise at du kan tegne interface (f.eks. kalt Kjøkken) i et slikt hierarki. Rom skal helst være merket som abstract. Det skal være tre nivåer, Rom øverst, de tre klassenen f.eks. EnkleRom, VanligRom og Suite på nivå to, og nederst (på nivå tre) to klasser som også implementerer interface-et Kjøkken.

Alle rom har et romnummer, et antall kvadratmeter (inklusive kjøkkenet hvis rommet har kjøkken), et antall sengeplasser og en etasjeangivelse. Alt dette er konstanter som er heltall og som skal gis verdier av konstruktørene. I tillegg har alle rom en boolsk variabel som sier om rommet er opptatt eller ledig og en referanse til et annet rom (slik at rom kan lenkes sammen i en liste). Bortsett fra navnene på klassene skal enkle rom, vanlige rom og suiter ha de samme egenskapene i programmet ditt.

Alle klassene skal ha en `toString()`-metode. Den skal gjøre det klart om det er et enkelt rom, et vanlig rom eller en suite, og den skal ta med alle verdier til alle instansvariabler og konstanter unntatt referansevariabelen til neste rom i listen. Ikke legg vekt på at resultatet skal bli pent hvis det skrives ut.

Oppgave 2. 8 poeng
Programmer alle klassene og interfacet.

Læringsmål oppgave 2.
Her skal du vise at du kan programmere klasser og interface. Det legges vekt på at alle instansvariabler er riktig deklarert og at alle untatt nestepekeren og den boolske variabelen opptatt er deklarert som konstanter (med final), at konstruktørene er riktig programmert (med super()), at interfacet er riktig programmert med en metodesignatur og at klassene som implementerer interface-et har en konstant som angir antall m2 og at metoden i interfacet returnerer denne verdien.
I tillegg skal den polymorfe metoden toString() defineres riktig med riktig kall på den samme metoden i superklassen (super.toString()).

Hotell med én etasje.

Du skal videre i oppgaven programmere deler av klassen `Hotell`, i første omgang med én etasje og siden med flere. Klassen skal inneholde en datastruktur som skal holde orden på alle rommene på hotellet. Hvert rom er beskrevet av et objekt av en subklasse til klassen `Rom`.

I et hotell med én etasje vil instansvariablen `etasje` være 1 i alle rom-objektene. Videre vil alle rommene i et slikt hotell være lenket sammen i en enkelkjedet liste der en referanse med navn `forsteRom` peker på det første rommet i denne listen.

Oppgave 3. 4 poeng
Anta at det er fire rom på hotellet. Tegn opp datastrukturen med disse fire rommene som alle skal være objekter av forskjellige klasser. Tegn opp alle instansvariabler med passende innhold men ikke tegn metoder.

Læringsmål oppgave 3. Her skal du vise at du skjønner hvordan en enkel datastruktur med en enkeltkjedet liste ser ut. Minst ett av objektene må ha egenskapen Kjøkken. Variabelen forsteRom må peke på det første objektet i listen . Objektene i denne listen henger sammen ved hjelp av neste-pekere i objektene selv og alle instansvariablene (og konstanter) er med.

Oppgave 4. 1 poeng
I klassen `Hotell` vil konstanten `MAX_ANT_SENGEPLASSER` (denne kan du sette til 8 i programmet ditt) angi det maksimale antallet sengeplasser det kan være i et rom. Deklarer denne konstanten og referansen `forsteRom` i klassen `Hotell`.

Læringsmål oppgave 4. Her skal du vise at du kan skrive meget enkle deklarasjoner. Referansen må være av typen Rom.

Gjester og reservasjoner

Anta at listen som `forsteRom` peker ut, er fylt med `Rom`-objekter som beskriver alle rommene i hotellet. Du skal nå programmere deler av et meget enkelt system med romreservasjoner og tildeling av rom til gjester.

Gjestene på hotellet er beskrevet av klassen `Gjest`. For enkelhets skyld skal den i ditt program bare inneholde et navn (en `String`) og, når en gjest har fått et rom, en referanse til dette rommet.

En romreservasjon er beskrevet av klassen `Reservasjon`. I denne oppgaven skal vi lage svært enkle reservasjoner uten noen datoer eller lengde på oppholdet. Objekter av klassen `Reservasjon` skal derfor bare inneholder en referanse til gjesten som har bestilt et rom, det antallet sengeplasser gjesten trenger på rommet sitt, en boolsk variabel som er true om gjesten ønsker et rom med kjøkken samt to pekere `forrigeR` (forrige reservasjon) og `nesteR` siden reservasjoner skal lagres i en dobbeltlinket liste. Den første reservasjonen pekes ut av instansvariablen `forsteR` og den siste reservasjonen pekes ut av variabelen `sisteR` i klassen Hotell.

Oppgave 5. 8 poeng
Deklarer de to variablene `forsteR` og `sisteR` i klassen `Hotell`. Programmer de to klassene `Gjest` og `Reservasjon`. For enkelhets skyld skal du ikke lage noen konstruktører eller metoder, og alle instansvariabler skal være åpent tilgjengelig i klassen `Hotell.` (Anta at hele programmet ditt er deklarert i samme mappe/pakke.)

Læringsmål oppgave 5. Referansene forsteR og sisteR må være av typen Reservasjon.
Du skal også vise at du kan programmere to enkle klasser. Klassen Reservasjon må ha riktige instansvariabler slik som angitt i oppgaveteksten. Instansvariablene børikke være private, og siden det står de skal være tilgjengelig i hele mappe/pakken bør de være "friendly", dvs. uten aksessmodifikator (men dette vurderes mildt siden de har fått beskjed om at aksessmodifikatorer ikke er viktige på eksamen)

I det følgende kan du regne med at det i programmet allerede er opprettet objekter for gjester og reservasjoner og at alle reservasjoner er kjedet sammen i den dobbeltlenkede listen.

Du skal skrive en metode i klassen `Hotell`: **`void`** `tildelRom(String navn)`.
Den skal brukes når en ny gjest som allerede har en reservasjon, kommer til hotellet. Metoden skal først gå gjennom listen av reservasjoner og finne den reservasjonen som er gjort av gjesten med dette navnet. Deretter skal reservasjonen tas ut av listen av reservasjoner. La metoden kaste et egendefinert runtime-unntak med gjestens navn om gjesten ikke har noen reservasjon.

Så skal metoden bruke hjelpemetoden `Rom finnRom(int antSeng, boolean kjokken)` til å finne et egnet rom. `finnRom()` skal gå gjennom alle rommene i hotellet for å finne et rom som er ledig og som tilfredsstiller kriteriene i reservasjonen. Antall sengeplasser i rommet skal være det samme som i reservasjonen. Eventuelt ønske om kjøkken må også tilfredsstilles, men en reservasjon som ikke har markert for ønske om kjøkken, kan gjerne få det allikevel. Hvis det ikke finnes et egnet rom, skal `finnRom()` returnere **`null`**.

Hvis `finnRom()` ikke finner et ledig rom med riktig antall sengeplasser, skal metoden `tildelRom()` gå i løkke og øke antall sengeplasser med én og kalle `finnRom()` på nytt helt til et rom er funnet.

Når et rom er funnet, settes det til opptatt, informasjon om rommet skrives ut i terminalvinduet (bruk `toString()`-metoden til dette) og en referanse til rommet settes inn i gjestens objekt.

Hvis `tildelRom()` har prøvd antall sengeplasser opp til `MAX_ANTALL_SENGEPLASSER` og det ikke er funnet noe ledig rom, skal metoden kaste et egendefinert runtime-unntak med navnet på gjesten og antall sengeplasser som ønskes.

Opppgave 6. 10 poeng
Skriv hjelpemetoden med signaturen `Rom finnRom(int antSeng, boolean kjokken)` som går gjennom listen av rom og returnerer referansen til et ledig rom med antall sengeplasser som angitt i parameteren og med kjøkken hvis andre parameter er **`true`**. Om det ikke finnes noe slikt rom, skal metoden returnere **`null`**.

Læringsmål oppgave 6. Her skal du vise at du kan skrive en metode som leter gjennom en lenket liste. Operatoren instanceof kan brukes for å teste om objektet har egenskapen Kjøkken (polymorfi er også en mulighet). Løkken må startes på riktig måte, løkken skal terminere når et Rom er funnet eller på slutten av listen. Riktig verdi må returneres.

Oppgave 7. 10 poeng
Skriv hjelpemetoden `void taUtRes(Reservasjon r)` som tar ut reservasjonen `r` fra den dobbeltkjedede listen av reservasjoner. Vi kan anta at `r` er i listen.

Læringsmål oppgave 7. Her skal du vise at du kan programmere en algoritme som fjerner et element fra en dobbeltkjedet liste. Det er viktig at programmet behandler tilfellet at det bare er én reservasjon i listen og at objektet først og sist i listen tas ut riktig.

Oppgave 8. 8 poeng
Skriv metoden `void tildelRom(String navn)` i klassen `Hotell`. Metoden skal kalle hjelpemetodene `finnRom()` og `taUtRes()`. Deklarer også de to egendefinerte unntakene.

Læringsmål oppgave 8. Her skal du vise at du kan programmere en litt mer kompleks algoritme og bruke metoder som allerede er skrevet. Uten at det står eksplisitt i oppgaven

bør tilfellet at det ikke finnes noe passende rom også håndteres riktig. Metoden må først lete etter reservasjon med riktig navn, så kalle taUtRes() og så gå i løkke og kalle finnRom() med økende antall sengeplasser til et rom er funnet eller maksimum antall sengeplasser er nådd. Instansvariablen i Gjest-objektet må settes til å peke på dette rommet. Det må lages to Unntaks-klasser og unntak må deklareres og kastes riktig.

Hotell med flere etasjer

Vi antar nå at hotellet har flere etasjer. For å holde oversikt over hvilke rom som er i hvilke etasjer, skal du innføre en ny datastruktur i hotellet. Hotellet har et antall etasjer som skal være en konstant (`ANTALL_ETASJER`) i klassen `Hotell`. Verdien til ANTALL_ETASJER skal settes i konstruktøren.

Alle rommene i samme etasje skal lenkes sammen i en enkelkjedet liste.
Referansen til det første rommet i hver etasje er inneholdt i en array kalt `forsteRomEtasje`. Array-elementet med indeks k peker således på listen av rom i etasje k. Vi skal også ta med etasje 0 (kjelleren), slik at arrayen skal være `ANTALL_ETASJER+1` lang.
I hele resten av dette oppgavesettet skal du regne med at klassen `Hotell` er slik at alle rommene er organisert i etasjer slik som beskrevet her.

Oppgave 9. 5 poeng
Tegn opp datastrukturen i et hotell med fire etasjer (`ANTALL_ETASJER` = 4), der det er to eller tre rom i hver etasje, men ingen rom i andre etasje eller i kjelleren. Tegn opp alle referanser, men av andre instansvariabler skal du bare tegne opp etasjenummer og antall kvadratmeter.

Læringsmål oppgave 9. Her skal du vise at du skjønner hvordan en datastruktur med en array der elementene refererer starten av lenkede lister av objekter ser ut.
På tre arrayplasser må arrayaelementet peke på første objekt i en liste av to eller tre rom.
To etasjer er tomme dvs. det er null-verdier i arrayen.
Det er ok om tegningen ikke viser hvilke egenskaper som evt. arves fra superklasser (som i lf)

Oppgave 10. 1 poeng
Deklarer klassen `Hotell` med konstruktør og arrayen `forsteRomEtasje` , men ennå ingen metoder.

Læringsmål oppgave 10. Her skal du vise at du kan programmere en klasse med en array og en konstruktør. Arrayen må opprettes av konstruktøren med riktig lenge som gis av parameter til konstruktøren.

Du skal nå skrive en iterator over alle rommene i hotellet. Iteratoren må kunne håndtere det tilfellet at det finnes etasjer der det ikke er noen rom.

Oppgave 11. 14 poeng
Skriv en iterator over alle rommene i klassen `Hotell`.

Læringsmål oppgave 11. Her skal du vise at du kan skrive en iterator. Klassen Hotel må implementere Iterabale<Rom> og det må finnes en metode som heter iterator() som returnerer en referanse til et objekt av en klasse som implementerer Iterator. Denne klassen inneholder next() og hasNext() og må programmeres på riktig måte. Antagelig er det best å ha en konstruktør til å initialisere variablene i denne klassen.

Du skal nå skrive en metode med signaturen `int[] ledigeRom()` i klassen `Hotell` som finner antall ledige rom fordelt på antall sengeplasser i hvert rom. Det betyr at antall ledige rom med 1 sengeplass skal legges i resultat-arrayen på indeksplass 0, antall ledige rom med 2 sengeplasser skal legges i resultatarrayen på indeksplass 1, osv. opp til at antall ledige rom med `MAX_ANT_SENGEPLASSER` sengeplasser skal legges inn i resultat-arrayen på indeks `MAX_ANT_SENGEPLASSER-1`.

Oppgave 12. 8 poeng
Bruk iteratoren du skrev i oppgave 11 til å skrive metoden `ledigeRom()`.

Læringsmål oppgave 12. Her skal du vise at du kan bruke iteratoren du selv har skrevet. Det gjøres mest elegant med en for-each-løkke, men eksplisitte kall på hasNext() og next() kan også godtas. Metoden må opprette en array av riktig lenge og øke verdien på riktig plass i arrayen med én hver gang metoden finner et ledig rom.

Programmering med tråder

Bedriften du har søkt jobb i, har mange hoteller spredt rundt i hele verden. Noen ganger trenger administrasjonen av hotellkjeden å få rapporter i løpet av svært kort tid, og det er derfor viktig at den som tilsettes, kan programmere med tråder. I denne oppgaven skal du lage en rapport om hvor mange ledige rom av hver størrelse (dvs antall sengeplasser) hele hotellkjeden har på tvers av alle hotellene i kjeden.

Anta at vi har en klasse `Hotellkjede` med konstantene `ANTALL_HOTELLER` og `KJEDE_MAKS_ANTALL_SENGEPLASSER` i tillegg til en array kalt `alleHoteller` med alle hotellene. Du skal ikke skrive denne klassen. I resten av oppgaven kan du anta at arrayen `alleHoteller` er fylt opp med hoteller.

Til oppgaven trenger vi en trådklasse. Hver tråd jobber på hvert sitt hotell og trenger derfor en referanse til hotellet det jobber på. Når en tråd kjøres, skal den hente ut hvor mange ledige rom det er i dette hotellet fordelt på antall sengeplasser per rom. Det er jo nettopp det metoden du programmerte i oppgave 12 gjør; derfor skal tråden din kalle denne metoden.

For å samle og summere opp verdiene fra de ulike trådene skal du lage en monitor. Trådene skal sende inn sine resultater til metoden `void rapporterLedigeRom(int[] ledige)`

i monitoren. (Derfor trenger trådene en referanse til en monitor.) Monitoren skal også ha en metode `int[] hentLedigeRom()` som henter ut antallet ledige rom fordelt på antall sengeplasser i hvert rom etter at alle trådene er ferdig.

Oppgave 13. 6 poeng
Skriv monitoren slik den er beskrevet over.

Læringsmål oppgave 13. Her skal du vise at du kan skrive en meget enkel monitor. Det må deklareres en lås og innholdet i metoden `rapporterLedigeRom()` *må beskyttes av denne tråden. Monitoren må også inneholde en array av lengde* `KJEDE_MAKS_ANTALL_SENGEPLASSER` *og verdiene i arrayen som er parameter må kopieres inn i denne arrayen. Å bruke original Java med synchronized er også mulig.*

Oppgave 14. 6 poeng
Skriv trådklassen.

Læringsmål oppgave 14. Her skal du vise at du kan skrive en meget enkel klasse som implementerer interface-et Runnable med metoden run(). Klassen må ha en konstruktør som angir hvilket hotell tråden skal jobbe på, og run-metoden skal bare kalle ledigeRom-metoden fra oppgave 12 og så kalle monitoren med resultatet. Hvis metoden `skrivUtLedigeRomMedTrader()` *venter vha. en barriere, må den være med i konstruktøren og telles ned på slutten av run()-metoden. Barrieren kan f.eks. vær en CountDownLatch. Subklassing av klassen Thread er også mulig.*

Oppgave 15. 6 poeng
Skriv en metode med signaturen `void skrivUtLedigeRomMedTrader()` i klassen `Hotellkjede`. Metoden skal opprette monitoren og opprette og starte alle trådene. Når alle trådene er ferdige, skal metoden hente ut resultatet fra monitoren og skrive ut dette resultatet i kommandovinduet.

Læringsmål oppgave 15. Her skal du vise at du kan opprette et monitor-objekt, opprette og starte mange tråder, vent på at trådene er terminert og skrive ut resultatet. Aktuelle parametre til trådenes konstruktører må være riktig. Venting kan f.eks. gjøres med en egen barriere eller vha. join().