NTNU
Kunnskap for en bedre verden

Prosjektoppgave – Bokregister for bibliotek

Oppgaven (Kravspek)

I prosjektet som skal gå ut semesteret, skal du lage en løsning som kan benyttes av biblioteket ved NTNU. Du skal med andre ord utvikle et **register av bøker. Prosjektet skal utvikles stegvis** i hver innlevering videre i semesteret. I siste innlevering skal vi legge på et tekstbasert brukergrensesnitt slik at vi får en fullstendig applikasjon.

Når prosjektet er ferdig (i siste innlevering før eksamen), skal det være mulig å gjøre følgende med applikasjonen din (kravspesifikasjon):

1. Det skal være mulig for brukeren å legge inn nye bøker
2. Bøker skal inneholde følgende informasjon:

    - Tittle
    - Forfatter
    - Forlag
    - Årstall utgitt
    - Antall sider
    - Strekkode (EAN-13)
    - Om boken er utlånt eller ikke

3. Brukeren skal kunne søke i registeret etter

    - Tittel
    - Forfatter
    - Strekkode

4. Bruker skal kunne slette en bestemt bok
5. Bruker skal kunne skrive ut på skjerm fullstendig liste av bøker i registeret

NB! Prosjektet skal løses **stegvis gjennom de neste innleveringene dette semesteret!** I denne innleveringen skal du utføre **Trinn 1** under.

Trinn 1 - Opprette en klasse for bok

Opprett et helt nytt prosjekt i BlueJ (eller den IDEen du benytter).

Opprett en klasse som skal representere en Bok. Gi klassen et fornuftig (engelsk) navn.

TIPS: Her kan du godt kopiere klassen du laget i oppgave 17 i Oblig 1 og bruke den som utgangspunkt.

Lag nødvendige *aksessor* og *mutatormetoder*. Ikke lag flere mutatormetoder enn det du mener det er behov for. M.a.o. reflekter over hvilken informasjon om en bok som kan endre seg.

Trinn 2 - Bokregister

Opprett en egen klasse som skal representere **selve bok-registeret** i applikasjonen din. Velg et navn du mener beskriver ansvaret og rollen til klassen godt (på engelsk).

Gjør deretter følgende:

1. Lag en metode som kan brukes for å legge til en bok til registeret.
2. Opprett en metode i registerklassen som fyller registeret med 4-5 bøker (nyttig funksjon å ha for senere å teste klassen slik at du slipper å legge inn bøker hver gang du skal teste). Velg et fornuftig navn på metoden som gjenspeiler hva metoden gjør (og ikke *hvordan*).
3. Lag en metode som skriver ut alle bøkene til konsollet. Tenk litt på at utskriften skal se oversiktlig ut.

Trinn 3 - Opprett/lag funksjoner for å søke i registeret

Følgende utvidelser skal implementeres:

1. I klassen som representerer biblioteket (som du laget i Trinn 2), lag en ny metode for å skrive ut samtlige bøker i biblioteket, men nå ved bruk av iterator og **while**-løkke. Kall metoden **listAllBooks2()** eller lignende.
2. I klassen som representerer biblioteket, lag en metode som søker etter første bok med en gitt tittel. Metoden skal returnere boken som blir funnet. Dersom ingen bok blir funnet som samstemmer med tittelen det søkes etter, skal **null** returneres.
3. I klassen som representerer biblioteket , lag en metode som søker etter alle bøker av en gitt forfatter. Metoden skal returnere **en samling av bøker** som alle er skrevet av forfatteren det søkes etter.
4. I klassen som representerer biblioteket , lag en metode som returnerer et Iterator<> objekt som andre objekter kan benytte for å iterere over bøkene i biblioteket (skal benyttes av brukergrensesnitt-klassen).

Trinn 4 - Brukergrensesnitt - første utgave

Opprett en helt ny klasse, som fremover skal utvikles til å bli **brukergrensesnittet** til applikasjonen. Kall klassen f.eks. **BookLibraryApp**.

Lag følgende metoder i denne klassen:

- **public void init()** - Denne metoden skal kalles ved oppstart av applikasjonen. Metoden skal opprette et objekt av bokregisteret ditt, og fylle registeret med et antall bøker.
- **public void listAllBooks()** - Metoden skal, ved hjelp av iterator-objektet som kan hentes fra bokregisteret, skrive ut samtlige bøker i registeret. (Her kan du kopiere/flytte koden du har skrevet i bokregisterklassen din over til denne metoden, slik at all brukerinteraksjon foregår i BookLibraryApp og ikke i bokregisteret eller i bok-klassen. Tenk også over hvordan du ønsker å presentere listen over bøker for brukeren. På tabell-form? Som en lang liste? Med eller uten innledende tekst? Skal du informere brukeren om hvor mange bøker det er i biblioteket osv?

Trinn 5 - Fullfør brukergrensesnittet

Følgende utvidelser skal implementeres:

1. Ferdigstill brukergrensesnittet i henhold til kravspesifikasjonen over (punkt 1 til 5).
2. Gå grundig igjennom endelig løsning og vurder om du:
    - Har laget en brukervennlig løsning.
    - Gir bruker tilstrekkelig tilbakemelding i alle stadier av applikasjonen
    - Gir brukeren gode tilbakemeldinger når feil oppstår, eller bruker taster feil verdier
    - Har laget en løsning som ikke kan "krasje"
3. Gå sammen 2 og 2. Bytt hverandres løsning og gjør samme vurdering som over av din medstudent sin løsning. Diskuter dere imellom hva som er god brukerinteraksjon og hva som er mindre god brukerinteraksjon (brukervennlighet).

Skriv et kort **refleksjonsnotat** under basert på gjennomgangen over.

Trinn 6 - Automatisk testing

Når du har lært om testing, og Unit-testing spesielt, legg til JUnit-tester på klassen som representerer en Bok, og for klassen som representerer selve bokregisteret.

For **Bok-klassen** skal du lage automatiske tester som:

- tester at et objekt (en instans) blir riktig opprettet (teste konstruktøren)
- tester at aksessormetodene returnerer riktige verdier (her kan du enten lage en test som tester alle aksessormetodene, eller en test pr aksessormetode)
- dersom klassen har mutator metode(r):
    - tester at mutatormetoden fungerer riktig
    - tester at det ikke er mulig å sette verdier som ikke er tillatt/ikke gir mening (negativt antall sider etc.)

For **Bokregister-klassen** skal du lage automatiske tester som

- tester at et objekt av klassen blir riktig opprettet
- tester at en bok blir lagt til riktig
- tester at det ikke er mulig å legge inn *null* i registeret (m.a.o. dersom noen skulle komme til å kalle **addBook(null)**)
- tester at metoden som returnerer antall bøker i registeret returnerer riktig verdi
- tester minst en av søkefunksjonene
- tester at det er mulig å slette en bok
- tester at registeret kan håndtere at noen forsøker å slette en bok som ikke finnes i registeret
- tester at søk etter bok som ikke finnes i registeret fungerer riktig

I svarfeltet under, skriv et kort **refleksjonsnotat** rundt det å skrive tester:

- Mener du at forslagene til hva som skal testes gitt over er tilstrekkelig for å garantere at løsningen din har høy kvalitet og ikke kan feile?
- Noen tester du mener mangler ? Beskriv i såfall hvilke og hvorfor du mener disse er viktige.
- Hva var vanskeligst med å skrive testene?

Prosjektet løses over 3 obligatoriske innleveringer i IDATA1001; Oblig #2, #3 og #4.