DA339A Inlämningsuppgift 2

Syfte

Inlämningsuppgift 2 är en del av de moment i kursen som examinerar (delvis) följande lärandemål:

- Kunskap och förståelse
  - visa förståelse för ett algoritmiskt tankesätt
  - visa förståelse för strukturerad och objektorienterad programmeringsteknik
- Färdighet och förmåga
  - kunna strukturera och implementera objektorienterade program i ett programmeringsspråk
  - kunna använda strukturerad och objektorienterad programmeringsteknik vid programutveckling
  - kunna dokumentera programvara

Inlämningsuppgift 2 utgör provkod 2004 *Inlämningsuppgifter del 2*. Provkoden utgör 2 hp på kursen och har betygsskala UG vilket innebär att studenten kan erhålla betyg U eller G på uppgiften.

Den muntliga redovisningen är också en läraktivitet i kursen då studenten får återkoppling på inlämnad lösning. Vid redovisningen närvarar en annan student som också ska redovisa och varje student lyssnar på en annan students redovisning och återkoppling. Att se andras kod och andras lösningar är en viktig del i att lära sig programmera och detta är ett tillfälle för detta.

Redovisning

Inlämningsuppgift 2 redovisas genom skriftlig inlämning på kursplatsen på Canvas samt en muntlig redovisning.

Inlämning ska endast göras om du har en komplett lösning som du anser uppfyller kraven för uppgiften.

Du får inte lämna in en lösning som saknar delar och använda redovisningstiden för hjälp. Behöver du hjälp med inlämningsuppgiften så nyttjar du handledningstider och/eller labbtider till hjälp för detta. Den som försöker redovisa en lösning som saknar delar kommer att avbrytas och hänvisas till nästa hjälptillfälle för hjälp och nästa omtillfälle för inlämning/redovisning.

Skriftlig inlämning

I den skriftliga redovisningen ska följande lämnas in:

- En zip-fil med källkod som innehåller de filer och kataloger som ingår i projektet/paketen. De filer som lämnas in ska kunna kompileras och programmet exekveras. Struktur med kataloger för paket ska behållas i zip-filen men inga andra kataloger eller filer ska finnas i inlämningen.
- En pdf-fil med namn enligt mönstret *da339a_InUpp2_förnamn_efternamn.pdf* (förnamn och efternamn ersätts med ditt namn) som innehåller en bild med ett klassdiagram. Klassdiagrammet får inte vara ritat för hand utan ska vara ritat med något verktyg exempelvis Visual Paradigm (men måste inte nödvändigtvis vara ritat med VP).

Deadline för inlämning till det ordinarie tillfället är **söndag 14/11 23.55**. Inlämningar som lämnats in sent garanteras inte redovisning vid det ordinarie tillfället.

Om man inte lämnat in till det ordinarie tillfället eller blir underkänd vid den muntliga redovisningen för det ordinarie tillfället så ges det två omtillfällen med deadline för inlämning:

- Onsdag 1/12 23.55
- Augusti 2022, exakt datum ej bokat

Observera att detta är de enda examinationstillfällena vi garanterar för Inlämningsuppgift 2.

Muntlig redovisning

För att göra den muntliga redovisningen måste en skriftlig inlämning gjorts inom deadline. Endast kompletta lösningar får redovisas.

Har du gjort en inlämning av den skriftliga delen bokar du en tid för redovisning via kalendern för kursen i Canvas. Se separata instruktioner på kursplatsen för hur du gör detta.

Inlämningsuppgiften redovisas muntligt och genom uppvisande av den inlämnade käll-koden och kompilering och exekvering av motsvarande kod. Studenten ska kunna visa upp sin källkod på dator för lärare/assistent som examinerar och kunna visa hur hen kompilerar och exekverar koden på dator. Studenten ska också kunna svara tillfredställande på frågor om sin kod och de lösningar hen implementerat samt kunna demonstrera förmåga att göra mindre ändringar i koden direkt och kompilera och exekvera igen (exempelvis ändra indata eller något gränsvärde). Studenten ska även kunna visa upp andra lösningar som krävs i inlämningsuppgiften exempelvis diagram, sanningstabeller eller andra skriftliga delar.

Redovisningen är en examination och är fokuserad på att göra en bedömning av den lösning och dokumentation av denna som studenten producerat. Det finns inte tid vid redovisningen att ge utförligare kommentarer om hela implementationen av uppgiften. Det finns vid denna redovisning inte heller tid att studenten beskriver hela sin tankeprocess eller förklarar hela programmet. Den lärare eller assistent som studenten redovisar för kommer att ställa frågor och be studenten demonstrera vissa saker. Diskussioner utöver detta finns det inte utrymme för vid redovisningen.

Önskar studenten återkoppling i större utsträckning på en lösning ges detta som vanlig hjälp vid laborationer.

Förberedelser för redovisning

Studenten förväntas vara förberedd när redovisningen startar och ha följande förberett innan redovisningen startar:

- Ha kommandotolk eller IDE startad och redo att kompilera och exekvera källkoden för uppgiften som kan redovisas (det vill säga ha navigerat till den katalog du har dina filer i innan din redovisningstid startar).
- Ha källkodsfiler öppnade och tillgängliga i editor/IDE för att kunna visa upp koden och kunna göra ändringar i denna vid förfrågan.
- Ha övriga filer med efterfrågad dokumentation tillgänglig och redo att visas upp.

Generella krav vid redovisning

För godkänd redovisning krävs:

- Väl formaterad källkod
- Källkod ska gå att kompilera och exekvera med ett resultat som efterfrågas för uppgiften
- Att uppgiften är löst på rimligt sätt oavsett om det är i kod eller annan form av dokumentation av lösning som efterfrågas
- Studenten ska muntligen kunna förklara sin lösning och beskriva varför lösningen ger det efterfrågade resultatet
- Studenten ska kunna visa förmåga att göra mindre ändringar i koden och kompilera och exekvera den ändrade koden med efterfrågat resultat via kommandotolk
- De lösningar som visas upp överensstämmer med de lösningar som lämnats in skriftligt.

Studenten ska alltid vara beredd på att kunna visa legitimation vid redovisning.

Redovisning i Zoom

Vid redovisning via Zoom så behöver du

- kunna kommunicera via mikrofon/högtalare
- ha kameran påslagen så att den som tar redovisningen kan se dig
- kunna uppvisa legitimation via kamera
- skärmdela den dator du sitter vid så att du kan visa dina lösningar, din kod och kompilering och exekvering

Har du inte möjlighet till detta vid en privat dator kontakta IT-supporten för studenter för att få hjälp med en lämplig plats för din redovisning.

Förberedelser

Inlämningsuppgift 2 omfattar kursinnehåll som gåtts igenom till och med föreläsning F15 och laboration L16. I Inlämningsuppgift 2 så används flera olika byggstenar för problemlösning som tidigare används i laborationer och exempel. Du bör ha lämnat in Inlämningsuppgift 1 då Inlämningsuppgift 2 bygger i stora drag bygger på den. Du bör även genomfört alla laborationerna L1-L15 innan du tar dig an Inlämningsuppgift 2 då dessa innehåller lösningsmetoder som du har nytta av till den här uppgiften.

Uppgift

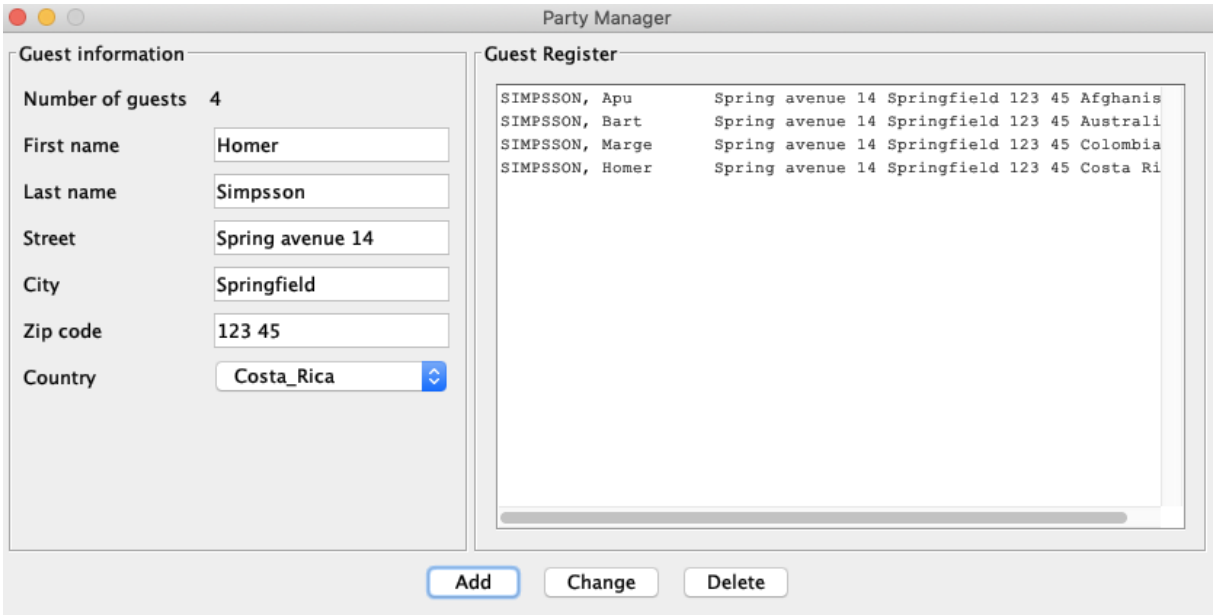
Övergripande beskrivning

I uppgiften så ska du implementera ett program i Java Swing. Programmet ska hantera en gästlista för ett event där man bjuder på mat (exempelvis ett bröllop eller någon annan fest).

I programmet ska användaren kunna välja saker som att lägga till, ta bort eller redigera gäster. Listan med namn på gäster och hur många gäster som finns skall synas i presentationsfönstret Party Manager. Exakt funktionalitet presenteras i lista med krav på programmets funktionalitet nedan.

Gästlistan i ditt program skall hantera ett bestämt antal gäster som sätts vid uppstart. Programmet ska däremot vara skrivet så att man enkelt skulle kunna ändra på gästlistans längd. Det vill säga: när du behöver veta antalet element i gästlistan så använd *.length* och inte en variabel som anger antalet element i listan.

Programmet skall vid starten fråga hur stor gästlistan skall vara (detta kan ske via kommandotolk eller dialogfönster). När programmet körs skall det visas ett fönster enligt bild nedan (exakt utseende varierar med operativsystem). Texten under Guest Register till höger behöver inte vara exakt utformad som nedan.



Krav på funktionalitet

Det program du skapar ska uppfylla följande krav på funktionalitet i programmet:

Id för krav	Kravbeskrivning
IU2F1	Användaren ska kunna se en lista med alla gäster som visar förnamn, efternamn och adress.
IU2F2	Användaren ska kunna se det totala antalet gäster.
IU2F3	När en gäst tas bort eller läggs till ska det totala antalet gäster som visas uppdateras.
IU2F4	Användaren ska kunna lägga till en ny gäst med namn och adress till gästlistan.
IU2F5	Användaren ska kunna ändra uppgifter om en gäst och gästens adress via GUI:t genom att välja en gäst i listan.
IU2F6	Användaren ska kunna ta bort en gäst ur gästlistan genom att välja gästen i listan i GUI:t.
IU2F7	Om ingen gäst är vald ska det visas ett felmeddelande om man försöker ändra eller ta bort en gäst.
IU2F8	När en gäst ändras, tas bort eller läggs till ska listan med alla gäster uppdateras.

Krav på implementation

Följande krav ställs på din implementation av uppgiften:

Id för krav	Kravbeskrivning
IU2I1	Källkoden ska vara väl formaterad med lämplig indentering som presenterats på kursen.
IU2I2	Högst upp i alla .java-filer ska det finnas en kommentar som innehåller: • Förnamn och efternamn • Datorid • Vilket program du läser
IU2I3	Källkoden ska gå att kompilera och exekvera.
IU2I4	Uppgiften ska vara löst på rimligt sätt utifrån de verktyg för problemlösning som presenterats på kursen till och med F15.
IU2I5	Lösningen ska följa den struktur som givits i filerna för uppgiften. Dessa filer innehåller också ledning till implementation.
IU2I6	Gästlistan ska representeras av en array i koden i klassen GuestManager och programmet ska skrivas så att man kan använda en array med olika antal element utan att behöva ändra något annat än initieringen av arrayen.
IU2I7	Implementationen får inte använda klasser med färdiga collections exempelvis Array, ArrayList eller liknande.
IU2I8	Element i arrayen skall implanteras som objekt enligt den struktur som givits i filen Guest.java
IU2I9	Element i arrayen som inte innehåller information om någon gäst skall sättas till NULL.
IU2I10	Objekt placeras i arrayen så att denna fylls från position 0 och uppåt. När en gäst tas bort ska den interna ordningen i arrayen behållas. Element på ett index större än det som tas bort flyttas till vänster (till ett index med lägre värde).
IU2I11	Om arrayen är full och en gäst ska läggas till ska arrayen utökas så det finns mer plats i denna.
IU2I12	Efter uppstart och inmatning av storlek på array ska enbart debug-utskrift visas i kommandotolk.
IU2I13	Utskrift och inmatning skall ske via fönster (undantaget storlek på array vid start).

Klassdiagram

Din lösning ska dokumenteras med ett klassdiagram som innehåller klasser med attribut, detaljer för associationer, operationer med parametrar och returvärden. Säkerställ att diagrammet du lämnar in stämmer överens med den kod du lämnar in.

Följande krav ställs på Klassdiagrammet:

Id för krav	**Kravbeskrivning**
IU2K1	Klassdiagrammet ska följa den UML-notation som presenterats på kursen.
IU2K2	Klassdiagrammet ska endast visa klassen MainFrame för Gui-delen. Inga andra klasser i paketet partyView ska visas i klassdiagrammet.
IU2K3	Alla klasser i paketen partyModel och partyController ska visas i klassdiagrammet.
IU2K4	Klassdiagrammet behöver inte ta hänsyn till att klasserna ligger i olika paket. Alla klasser visas i ett och samma klassdiagram utan paket i diagrammet.
IU2K5	De klasser som ska visas i klassdiagrammet ska visas med alla attribut och operationer (metoder i Java) och parametrar och returvärden till dessa.
IU2K6	Associationer ska visas med multiplicitet i bägge ändarna och lämpliga typer av associationer ska användas (exempelvis navigeringsriktning och aggregation eller komposition).
IU2K7	Klassdiagrammet ska vara ritat med något verktyg som Visual Paradigm eller liknande och får inte vara en handritad skiss på papper som sedan skannats in.

Tips

Nedan följer några tips för att ta sig an uppgiften.

Det första tipset är att verkligen läsa och titta på den kod som finns i det givna kodskelettet. Du hittar exempel på hur du använder JOptionPane i denna och hur du kommer åt värden från GUIt och hur du uppdaterar detta.

Kom ihåg att jobba iterativt. Gör små ändringar i taget och testa. Försök bara göra en sak i taget så långt det går.

Lägg första till kod i klasserna Address och Guest så att du kan kompilera dessa. Ta sedan en sak i taget och uppdatera GuestManager och Controller för en funktion i taget. Det enklaste är att börja med funktionen för att lägga till en gäst. Där efter att ta bort en gäst och sist att ändra en gäst.

Även när du gör detta så bör du bryta ner det i mindre delar. Använd debug-funktioner i din IDE eller extra utskrifter i prompten för att kontrollera att du får rätt värden från GUI, att det blir rätt index i arrayen och liknande.

Spara ofta! Testa ofta!

För specifika saker du behöver kolla upp se vilken föreläsning och labb som hade vilket innehåll via filen med innehåll och läsanvisningar i Moduler->Kursinformation på kursens Canvasplats.

Gå verkligen till labb/hjälp-tillfällen för att få hjälp om du sitter fast med något.