Java 19 Clean Code examination

Examinationen för kursen Clean Code är en arbetsuppgift som löses individuellt eller i par på egen vald tid och plats.

Redovisning sker skriftligt med inskickade uppgifter via mail <u>ulf@bilting.se</u> samt vid ett muntligt redovisningtillfälle. Båda redovisningarna krävs för betyget G.

Den skriftliga redovisningen sker med inskickade java-filer i ett zip-arkiv samt en ca en halv A4-sida som kortfattat beskriver de refaktoriseringar och andra förbättringar som utförts.

Vid den muntliga redovisningen skall programmet kunna köras och all kod och gjorda refaktoriseringar och tester skall kunna beskrivas, diskuteras och motiveras.

Programmet

Koden som skall refaktoriseras och testas är medvetet fullt av code smells men fungerar vid normal användning. Det implementerar spelet "Moo" som är ett Mastermind-liknande spel där programmet slumpar fram ett fyrsiffrigt tal som spelaren skall gissa, med så få gissningar som möjligt. Det slumpade talet har garanterat fyra unika siffror.

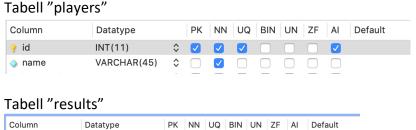
Spelaren gissar med fyra siffror. Som svar visar spelet ett antal B:n och C:n (bulls and cows). Ett B för varje siffra som har satts på rätt plats och ett C för varje siffra som finns med bland de fyra men är på fel plats. "B," betyder alltså att en av fyra siffror i gissningen är rätt och är på rätt plats (men man får inte reda på vilken som är rätt). ",CCC" betyder att tre gissade siffror finns med, men ingen av dem är på rätt plats. "BBBB," är sålunda målet och då är omgången slut.

Spela några gånger med en "fuskutskrift" av talet så är det lätt att lära sig reglerna. Spelaren får göra gissningar med icke-unika siffror om hen vill, men målet har alltid fyra unika. Efter varje spel skrivs alla spelares genomsnittliga antal gissningar ut i en topp-tio-lista.

Spelet för statistik över utförda spel i en databas "Moo" med en tabell "players" som innehåller två kolumner "id" och "name". Denna används för "inloggning" i början av spelet. I tabellen "results" sparas varje spel i tre kolumner "id", "result" och "player", den sista är foreign key som refererar till player-tabellens "id".

Denna databas behöver skapas innan programmet går att köra. Några spelare behöver läggas till i "player"-tabellen manuellt.

Tabellerna i databasen som skall skapas bör se ut så här (i mysql):



INT(11) INT(11)

result

Examinationsuppgiften

Uppgiften för examinationen är att refaktorisera programmet genom att identifiera code smells och minimera/eliminera dem samt att göra koddelarna testbara för unit-testing och skriva tester som testar programlogiken.

Refaktorisering

Programmet skall göras "vackrare" genom utförande av refaktorisering på lämpliga delar. Kursbokens ideer om

- Namngivning
- Funktioner/metoder
- Kommentarer
- Klasser
- Felhantering
- Clean tests

skall beaktas och genomföras. Om något designmöster är lämpligt att använda, gör så.

Programmets olika skikt som sköter

- Användargränssnitt
- Programlogik
- Databaskod

skall separeras och delarna skall inte känna till annat om de andra delarna än ett javainterface och **skall** använda Dependency Injection för sammansättning av delarna. Konkret objektskapande och sammansättning av delarna behöver inte utföras med något DIramverk, utan kan skötas med manuell kod i main-metoden/klassen.

Klassen SimpleWindow kan anses som en "färdig" biblioteksklass och behöver inte modifieras eller refaktoriseras alls eftersom vi inte gått igenom Swing mer än översiktligt. Göm den bakom ett java-interface.

Testning

Tester **skall** skrivas för programlogikkoden. Eftersom den anropar databaskod och användargränssnittkod, behöver dessa mycket troligtvis Mock-Object-implementeras för att kunna utföra testerna. Kanske behöver också metoden som genererar slumpsiffrorna mockas, så att tester på programlogiken kan utföras med kända "slump"-siffror.

Arbetets omfattning

En uppgift av detta slag har naturligtvis inget enda "rätt svar".

Koden **skall** efter utförd refaktorisering sakna tydliga code smells.

Koden **skall** vara uppdelad i helt skilda skikt för användargränssnitt, spel och DB/integration. Spelet bör troligtvis uppdelas i spel-controller och spellogik, så att logiken kan testas separat.

Testerna **skall** testa väsentliga delar av programlogiken, men behöver inte vara alltäckande, det viktigaste är att visa att tester går att genomföra och att några representativa tester finns på plats.

De viktiga steg som utförts **skall** dokumenteras på ca en halv A4-sida. Detaljer framgår av java-koden så beskrivningen behöver bara vara mer av en översikt.

Om du hittar svagheter, buggar, utelämnad felhantering eller säkerhet, etc. får du gärna påpeka eller rätta, men det ingår inte i examinationsuppgiften att göra det. (Programmet **är** säkert inte alls perfekt i dessa avseenden.) Samma sak gäller för JDBC-koden, det ingår inte i uppgiften att använda någon annan teknik, men själva koden kan och skall göras vackrare.

Frågor

Det är helt OK att ställa frågor om arbetets omfattning och krav, om något i denna beskrivning skulle vara oklart, eller min kod obegriplig. Det är också helt OK att diskutera och få handledning av arbetet under kurstid!

Uppgift för VG

För att få betyg VG skall allt ovanstående vara utfört plus en extrauppgift:

Strukturera/utöka programmet så att man kan välja att spela andra liknande spel.

T ex MasterMind, där de fyra siffrorna bara kan ha sex olika värden (sex färger i originalet) och dessutom kan förekomma fler än en gång i slumpgruppen. För varje spel krävs en ny resultattabell i databasen, men spelartabellen kan delas mellan spelen.

Implementera minst en extra spelvariant och beskriv vad som behöver göras för att lägga till ytterligare spel. Tester för det nya spelet behöver inte implementeras.

Eftersom det handlar om ett utbytbart varierande del-beteende luktar det Strategymönstret, men uppgiften kan kanske lösas snyggt även på annat sätt.

Resultatet är ett program där man med minimala ändringar kan lägga till kod för nya spel av "Moo-karaktär".

Redovisning av VG

Denna sker på samma sätt som för betyg G:

- Inskickad kod i zip-fil
- Muntlig redovisning med körning
- Halv A4-sida som beskriver hur fler-spels-dynamiken åstadkommit.