

1 UML/RUP

Markera om följande påståenden är sanna eller falska:

(+1 för rätt svar, ingen förändring för fel svar)

Ett sekvensdiagram beskriver alla klasser som är inblandade i en sekvens med händelser

☐ Sant

☐ Falskt



Ett klassdiagram visar alla objekt som skapas av varje klass

☐ Sant

☐ Falskt



Ett use case diagram visar hur man använder en viss klass

☐ Sant

☐ Falskt



Att klassen "Barn" ärver från klassen "Förälder" betyder att klassen Förälder måste ha ett privat attribut av typen "Barn".

☐ Sant

☐ Falskt



Ett sekvensdiagram visar bara användarna, systemet, och vilka händelser som användarna genererar mot systemet

☐ Sant

☐ Falskt



En domänmodell är ett slags karta över alla use cases i domänen, och hur de hänger samman

☐ Sant

☐ Falskt



Det är först när man har ett klassdiagram som man kan börja fundera på hur systemet skall testas

☐ Sant

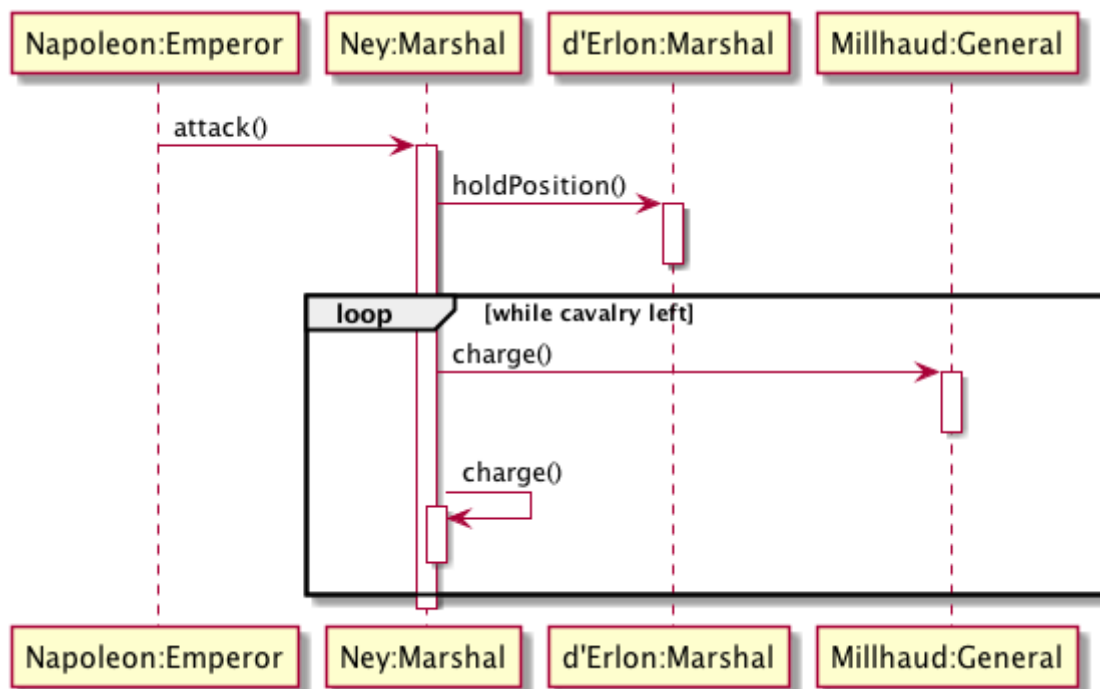
☐ Falskt



Totalpoäng: 7

2 Sekvensdiagram

Givet följande sekvensdiagram:



Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

metoden "attack()" skall ligga i klassen "Marshal".

☐ Sant



☐ Falskt

Både klassen "Marshal" och klassen "General" skall ha en metod "charge()"

☐ Sant



☐ Falskt

Objektet "d'Erlon" har inte metoden "charge()"

☐ Sant

☐ Falskt



Objektet "Napoleon" får aldrig veta (enligt det här sekvensdiagrammet) att objektet "Ney" håller på att ta slut på hans kavalleri.

☐ Sant



☐ Falskt

i metoden "attack()" finns det programkod för att hitta objektet "Millhaud:General" och anropa metoden "charge()" på detta objekt.

☐ Sant



☐ Falskt

metoden "attack()" har inte programkod för att anropa "this->charge()", utan anropet måste komma utifrån.

☐ Sant

☐ Falskt



metoden "attack()" måste först vänta på att objektet "d'Erlon" blir klar med att köra metoden "holdPosition()" innan den kan fortsätta.

☐ Sant

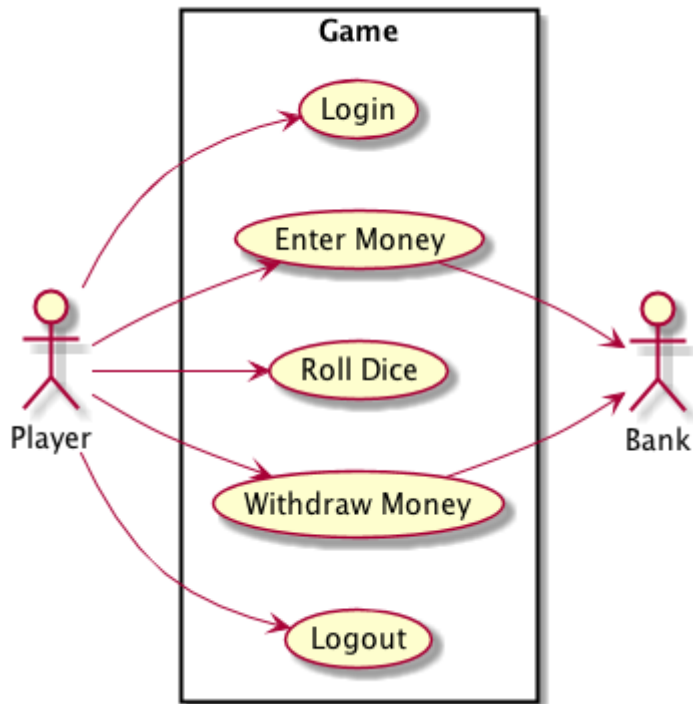


☐ Falskt

Totalpoäng: 7

3 Use Case Diagram

Givet följande Use Case Diagram:



Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

En Player måste utföra Use Casen uppifrån och ner i den ordning de står (Login, Enter Money, Roll Dice, Withdraw Money, Logout)

- ☐ Sant
- ☐ Falskt



Om man inte får kontakt med aktören Bank så går det inte att genomföra use caset "Enter Money"

- ☐ Sant
- ☐ Falskt



Use case diagrammet säger ingenting om att man måste vara inloggad ("Login") för att spela ("Roll Dice")

- ☐ Sant
- ☐ Falskt



Att man måste vara inloggad (dvs. framgångsrikt avslutat use caset "Login") innan man kan "Roll Dice" beskriver man med hjälp av ett precondition i use caset "Roll Dice"

☐ Sant

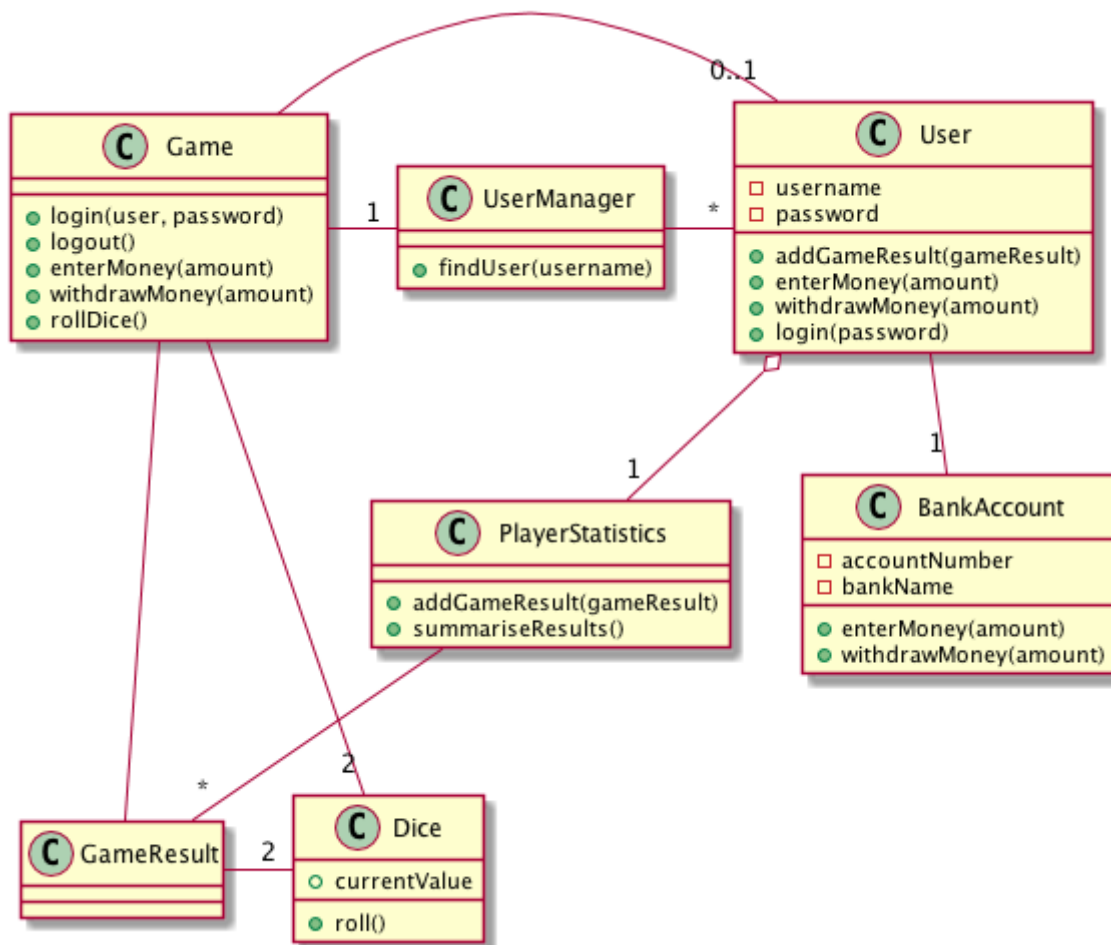


☐ Falskt

Totalpoäng: 4

4 Klassdiagram

Givet följande klassdiagram:



Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

Varje gång man spelar behöver man två nya tärningar

- ☐ Sant
- ☐ Falskt



Enligt designmönstret "High Cohesion" finns klassen "PlayerStatistics" för att inte klassen "User" skall behöva ha två olika ansvarsområden.

- ☐ Sant
- ☐ Falskt



Enligt designmönstret "Low Coupling" hade det varit bättre om "Game" inte hade haft någon association till "Dice" -- det hade räckt om GameResult hade det.

☐ Sant



☐ Falskt

Objektet "mainGame:Game" får bara ha högst en association till ett objekt av typen User i taget.

☐ Sant



☐ Falskt

Associationen mellan "User" och "BankAccount" implementeras bäst med hjälp av en array.

☐ Sant

☐ Falskt



Klassen "User" är en controller för allt som har med enskilda användare att göra.

☐ Sant

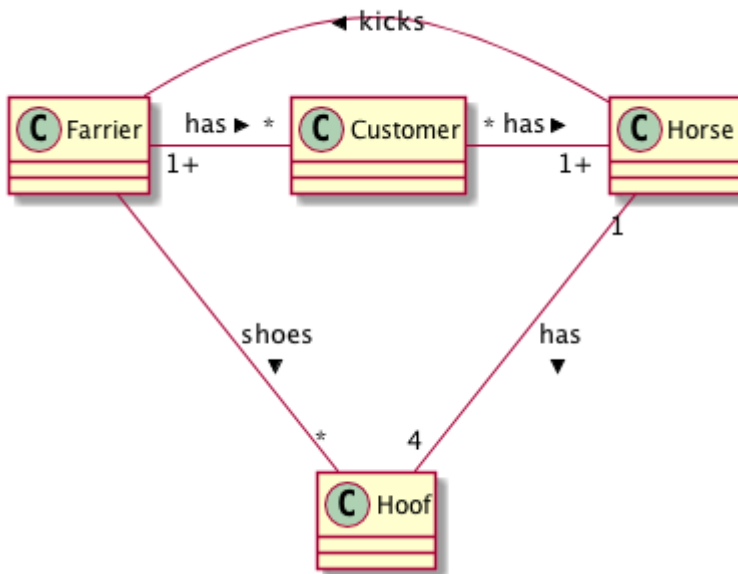


☐ Falskt

Totalpoäng: 6

5 Klasser och Relationer

Givet följande klassdiagram:



Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

En hovslagare (Farrier) behöver inte ha några kunder

☐ Sant



☐ Falskt

Det är odefinierat hur många hästar som sparkar sin hovslagare

☐ Sant



☐ Falskt

Börje:Customer har Brunte:Horse och Rosa:Horse

☐ Sant



☐ Falskt

En hovslagare (Farrier) skor hästar (Horse)

☐ Sant

☐ Falskt



Hästen Lukas:Horse har, efter en olycka, bara tre hovar.

☐ Sant

☐ Falskt



Hovslagaren John:Farrier kan sko hovarna på hur många hästar som helst.

☐ Sant



☐ Falskt

Wellington:Customer har bara Copenhagen:Horse

☐ Sant



☐ Falskt

Totalpoäng: 7

6 Design Patterns

Markera om följande påståenden är sanna eller falska:
(+1 för rätt svar, ingen förändring för fel svar)

En Observable är en klass med data som andra klasser kan vara intresserade av

☐ Sant



☐ Falskt

Objektet main:GUIController, som är en Controller, ansvarar för att kontrollera att användaren använder gränssnittet rätt.

☐ Sant

☐ Falskt



Objektet main:GUIController, som är en Controller, ansvarar för att skicka vidare händelser som användaren genererar mot gränssnittet till andra delar av applikationen som utför själva jobbet.

☐ Sant



☐ Falskt

Ett Strategy pattern består av minst tre klasser med rollerna Context, AbstractStrategy, och ConcreteStrategy

☐ Sant



☐ Falskt

Designmönstret Factory handlar om att all data (Facts) skall samlas i så få klasser som möjligt.

☐ Sant

☐ Falskt



Totalpoäng: 5

i Grade limits

Betygsgränserna för denna tenta är:

Betyg	Procent	Poäng
MAX	100%	36
A	90%	32
B	80%	29
C	70%	25
D	65%	23
E	60%	21

Lycka till!