1 Teori

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

Ett objekt är en instans av en viss klass.

- Falskt
- Sant ✔

Ett sekvensdiagram visar i vilken ordning man skall bygga sitt system.

- Sant
- Falskt ✔

Ett Use Case Diagram ger en översikt över alla Use Cases, vilka aktörer som är inblandade, vilka delsystem respektive use case hör till, och hur use casen är relaterade till varandra.

- Falskt
- Sant ✔

Ett klassdiagram visar de metoder och attribut som objekt av varje klass har.

- Sant ✔
- Falskt

Man gör ett interaktionsdiagram för varje systemhändelse.

- Sant ✔
- Falskt

Ett interaktionsdiagram beskriver interaktionen mellan olika klasser.

- ○ Falskt ✔
- ○ Sant

Totalpoäng: 6

2 GRASP Patterns

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

Information Expert betyder att ansvaret för att arbeta med en viss information bör ligga i den klass som innehåller informationen.

- Sant ✔
- Falskt

Information Expert betyder att man skall lägga informationen i den klass som har metoderna för att hantera den.

- Sant ✔
- Falskt

Ansvarsdriven design handlar om att man alltid skall sätta ut vem som är ansvarig för ett visst designbeslut så man kan utkräva ansvar när något går fel.

- Sant
- Falskt ✔

High Cohesion går ut på att varje klass skall ha så få och så välavgränsade ansvarsområden som möjligt.

- Sant ✔
- Falskt

Low Coupling går ut på att man skall sträva efter att ha så få och så "lösa" associationer som möjligt mellan klasser i ett system.

- Sant ✔
- Falskt

Controllers kan anropa andra Controllers.

- Sant ✔
- Falskt

En Controller kan anropa Information Experts.

- Falskt
- Sant ✔

Controller kräver Polymorfism för att fungera.

- Falskt ✔
- Sant

Det får bara finnas en instans av en Information Expert i ett system.

- Falskt ✔
- Sant

Enligt High Cohesion skall varje klass göra så mycket som möjligt.

- Falskt ✔
- Sant

Totalpoäng: 10

3 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

Observer Pattern består av Observers som regelbundet letar efter förändringar i klasser av typen Observable.

- ○ Falskt ✔
- ○ Sant

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

- ○ Falskt ✔
- ○ Sant

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

- ○ Falskt
- ○ Sant ✔

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

- ○ Falskt ✔
- ○ Sant

Singleton betyder att man bara får anropa klassen en gång.

- ○ Falskt ✔
- ○ Sant

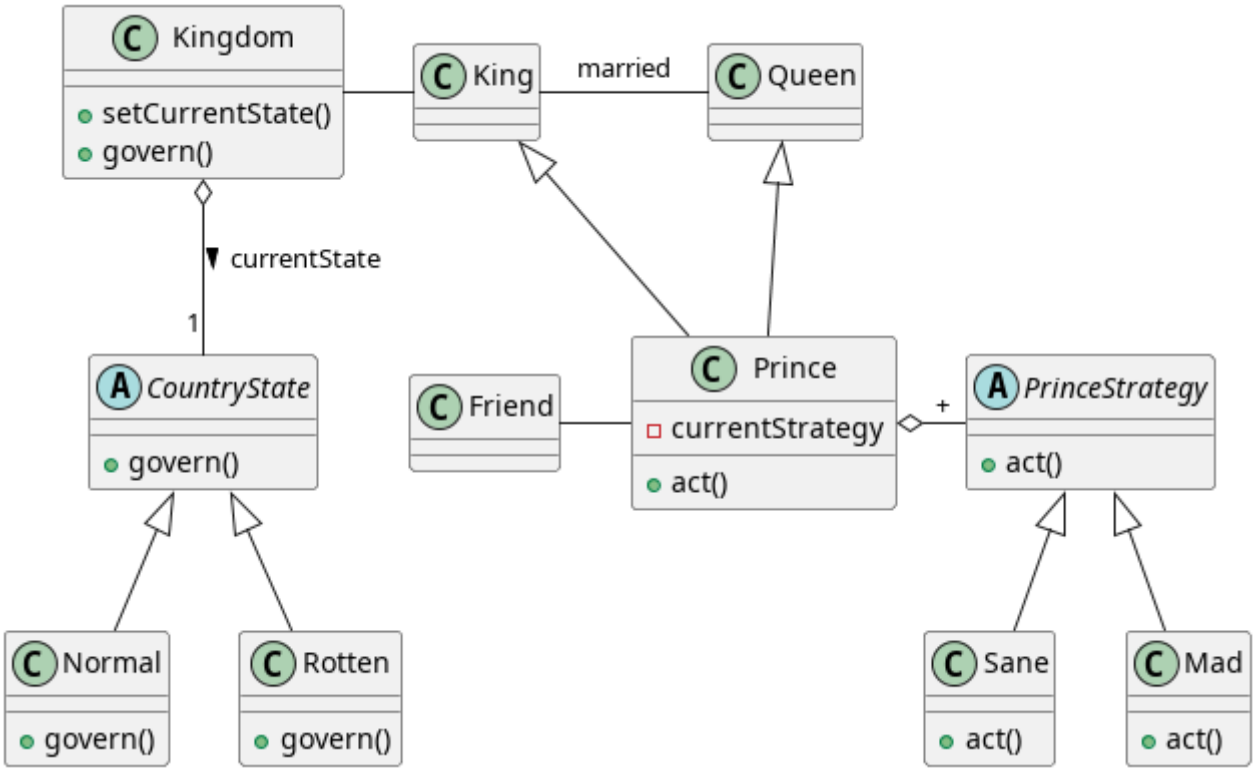
I State pattern är det klassen med rollen «Abstract State» som ansvarar för vilket tillstånd man skall byta till.

- ○ Falskt ✔
- ○ Sant

Totalpoäng: 8

4 Klassdiagram

Betrakta följande klassdiagram:



Klassdiagrammet beskriver relationer ur en pjäs. För varje påstående nedan, markera om diagrammet stödjer påståendet (sant) eller inte stödjer påståendet (falskt).
(+1 för rätt svar, ingen förändring för fel svar)

hamlet:Prince har en association med ofelia:Friend

- Falskt
- Sant ✔

denmark:Kingdom kan antingen ha tillståndet currentState:Normal eller currentState:Rotten

- Falskt
- Sant ✔

england:Kingdom kan samtidigt ha en varsin association till currentState:Normal och currentState:Rotten

- Falskt ✔
- Sant

hamlet:Prince har en association med denmark:Kingdom

- Falskt
- Sant ✔

gertrude:Queen kan inte styra denmark:Kingdom

- Falskt
- Sant ✔

hamlet:Prince kan agera både enligt Sane::act() och Mad::act() samtidigt.

- Sant ✔
- Falskt

horatio:Friend vet alltid om hamlet:Prince agerar Sane eller Mad.

- Falskt ✔
- Sant

gertrude:Queen är gift med claudius:King

- Falskt
- Sant ✔

sweden:Kingdom vet inte längre exakt vilket CountryState landet befinner sig i, bara att det har en referens till något objekt av typen CountryState.

- Sant ✔
- Falskt

Det är polonius:CountryState som genom metoden CountryState::govern() bestämmer om landet skall styras som Normal::govern() eller Rotten::govern().

- Falskt ✔
- Sant

hamlet:Prince är en Queen □□

- Sant ✔
- Falskt

Totalpoäng: 11

5 Relationer mellan Klasser

Betrakta följande klassdiagram:

Klassdiagrammet beskriver hur kunder kan ha olika abonnemangsformer (t.ex. mobiltelefonabonnemang). För varje påstående nedan, markera om diagrammet stödjer påståendet (sant) eller inte stödjer påståendet (falskt).
(+1 för rätt svar, ingen förändring för fel svar)

anthony:Subscriber använder samma :Tariff för alla tider på dygnet

- ○ Sant ✔
- ○ Falskt

prepaid:ServicePlan och mini:ServicePlan använder nightPrice:Tariff

- ○ Falskt
- ○ Sant ✔

bob:Subsciber och charlie:Subscriber använder samma :ServicePlan

- ○ Falskt
- ○ Sant ✔

david:Subscriber har en workPhone:ServicePlan och en burnerPhone:ServicePlan

- ○ Falskt ✔
- ○ Sant

x:Tariff och y:Tariff har båda priset 2kr och gäller mellan 07:00 och 17:00

- Sant ✔
- Falskt

doppio:ServicePlan innehåller både x:Tariff och y:Tariff med priset 2kr och som gäller mellan 07:00 och 17:00

- Falskt
- Sant ✔

flex:ServicePlan har en separat :Tariff för varje timme på dygnet

- Falskt ✔
- Sant

whistle:ServicePlan saknar helt :Tariff

- Falskt ✔
- Sant

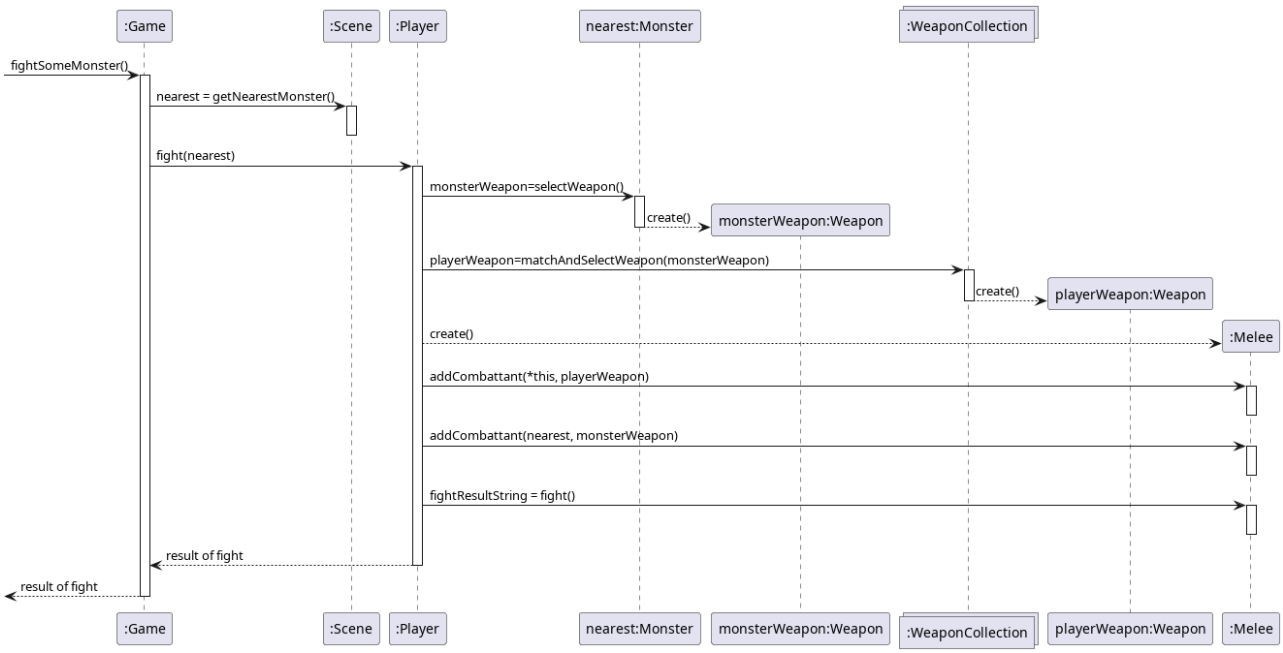
När eric:Subscriber vill byta upp sig till maxi:ServicePlan så måste han först säga upp sin mini:ServicePlan

- Sant ✔
- Falskt

Totalpoäng: 9

6 Interaktionsdiagram

Betrakta nedanstående sekvensdiagram:



Interaktionsdiagrammet visar systemhändelsen fightSomeMonster() i ett textbaserat äventyrsspel. För varje påstående nedan, markera om diagrammet stödjer påståendet (sant) eller inte stödjer påståendet (falskt).
(+1 för rätt svar, ingen förändring för fel svar)

Monster har inga vapen (Weapon) färdiga utan skapar dem vid behov.

- ○ Falskt
- ○ Sant ✔

:Scene är information expert på vilka Monster som finns i närheten.

- ○ Falskt
- ○ Sant ✔

:Player är information expert på vilka vapen de har.

- ○ Falskt ✔
- ○ Sant

Eftersom man måste ta hänsyn till omgivningen när man slåss, så är det :Scene som är information expert på hur en fight() skall gå till.

- Falskt ✔
- Sant

Klasserna Game, Player, och Melee har alla en metod fight()

- Sant
- Falskt ✔

Objektet :Scene tar emot resultatet från Melee::fight() och formatterar om det så att det går att visa i användargränssnittet.

- Falskt ✔
- Sant

:Player är en controller.

- Sant ✔
- Falskt

Klassen Melee måste ha två metoder som båda heter addCombattant().

- Sant
- Falskt ✔

Totalpoäng: 8

Betygsgränser

Betygsgränserna för denna tenta är:

Betyg	Procent	Poäng
MAX	100%	44
A	90%	40
B	80%	35
C	70%	31
D	65%	29
E	60%	26

Lycka till!