1 Blandad Teori

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

En domänmodell visar hur olika objekt interagerar.

- ○ Sant
- ○ Falskt ✔

Ett objekt är en instans av en viss klass.

- ○ Sant ✔
- ○ Falskt

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

- ○ Sant
- ○ Falskt ✔

Ett sekvensdiagram visar de objekt som samarbetar för att lösa en viss systemhändelse.

- ○ Sant ✔
- ○ Falskt

Man skriver testfall till ett system först när man har skrivit programkoden. Först då vet man vilket system man faktiskt byggde och därmed skall testa.

- ○ 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.

- ○ Sant ✔
- ○ Falskt

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

- Sant ✔
- Falskt

Ett systemsekvensdiagram är ett slags sekvensdiagram där man ser vilka systemhändelser aktörer anropar mot systemet i ett visst use case.

- Sant ✔
- Falskt

I ett sekvensdiagram kan man se vilka klasser som objekten är instanser av.

- Sant ✔
- Falskt

En metod som är deklarerad som public får inte använda sig av attribut (i samma klass) som är private.

- Sant
- Falskt ✔

I sekvensdiagram listar man alla attribut och deras värden längst ner under varje objekts livlina.

- Sant
- Falskt ✔

Man gör ett interaktionsdiagram för varje systemhändelse.

- Sant ✔
- Falskt

Ett interaktionsdiagram beskriver interaktionen mellan olika klasser.

- Sant
- Falskt ✔

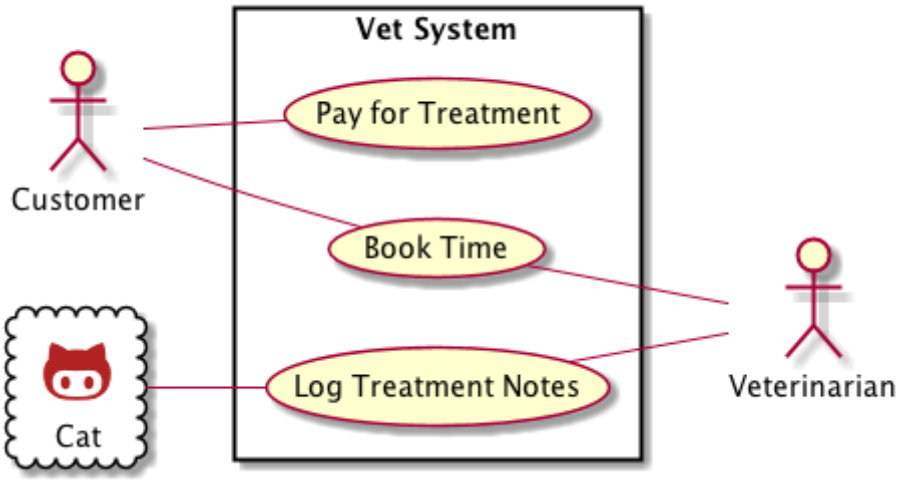
Samarbetsdiagram och Sekvensdiagram visar samma sak.

- ○ Sant ✔
- ○ Falskt

Totalpoäng: 14

2 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)

Kunden och Katten behöver inte träffas.

- ○ Sant ✔
- ○ Falskt

Veterinären tar emot betalning för en behandling.

- ○ Sant
- ○ Falskt ✔

Kunden måste betala för en behandling först, sedan bokar kunden en tid, och sist behandlas katten (och anteckningar från behandlingen loggas).

- ○ Sant
- ○ Falskt ✔

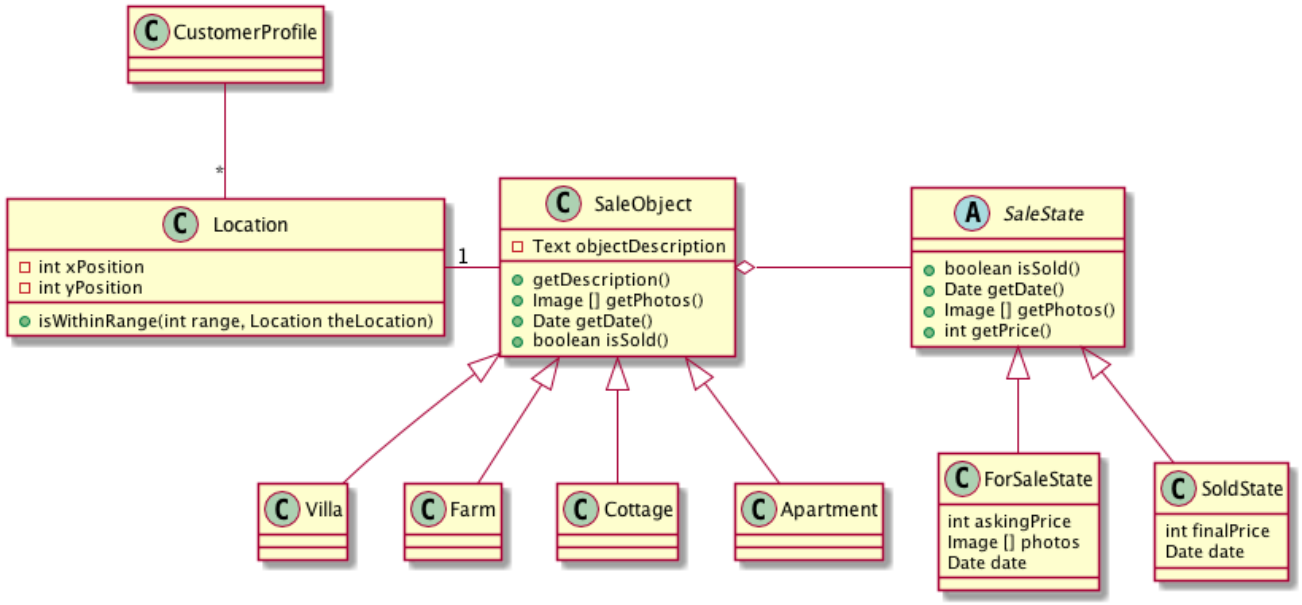
Katten och Veterinären arbetar båda två mot systemet för att logga behandlingsanteckningar.

- ○ Sant ✔
- ○ Falskt

Totalpoäng: 4

3 Klassdiagram

Betrakta följande klassdiagram:



Klassdiagrammet beskriver en marknadsplats för hus (tänk Hemnet) där man kan se vilka hus som finns till salu på en karta. I sin användarprofil anger man en (eller flera) Location, och sedan kan man söka efter hus i en viss radie runt den punkten (t.ex. Ge mig alla hus inom en radie på 50 km från Karlskrona Centrum). Man kan också se vilka hus som har sålts i samma region, och vad de kostade.

För varje påstående nedan, markera om klassdiagrammet stödjer påståendet (sant) eller inte stödjer påståendet (falskt) (+1 för rätt svar, ingen förändring för fel svar).

När man skapar ett object av typen Location så kommer det automatiskt skapas ett objekt av typen SaleObject.

- ○ Sant
- ○ Falskt ✔

Det finns inga metoder för att ändra på värdena i ett Location-objekt. Har man väl skapat en :Location så kan man med andra ord inte byta koordinaterna i den.

- ○ Sant ✔
- ○ Falskt

SaleObject, SaleState, ForSaleState, och SoldState ingår tillsammans i designmönstret State pattern.

- Sant ✔
- Falskt

klasserna Villa, Farm, Cottage, och Apartment använder sig av Polymorfism för att hantera de skilnader som finns mellan de olika typerna av hus.

- Sant ✔
- Falskt

f:Farm består av flera åkrar som var och en har en referens till ett objekt av typen Location.

- Sant
- Falskt ✔

a:Farm, b:Cottage, och c:Apartment har alla tre en referens till samma :Location -objekt.

- Sant ✔
- Falskt

I metoden SaleObject::isSold() finns kodraden myState = new SaleState(); .

- Sant
- Falskt ✔

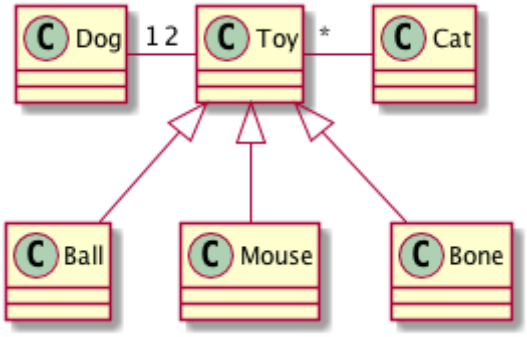
CustomerProfile har referenser till två olika :Location-objekt som har samma värden på xPosition och yPosition.

- Sant ✔
- Falskt

Totalpoäng: 8

4 Relationer mellan Klasser

Betrakta följande relationer mellan klasser:



Notera att detta diagram fokuserar på relationerna mellan klasserna, och går inte in på detaljer i vilka metoder eller attribut som finns.
För varje påstående nedan, markera om relationerna mellan klasserna stödjer påståendet (sant) eller inte stödjer påståendet (falskt) (+1 för rätt svar, ingen förändring för fel svar).

tofflan:Cat äger b1:Ball

- ○ Sant ✔
- ○ Falskt

lasse:Dog äger b2:Ball och bone1:Bone

- ○ Sant ✔
- ○ Falskt

lasse:Dog och ludde:Dog tuggar på varsin ända av bone1:Bone

- ○ Sant
- ○ Falskt ✔

lasse:Dog får en ny leksak b3:Ball i julklapp. Då måste han glömma bort antingen b2:Ball eller bone1:Bone

- ○ Sant ✔
- ○ Falskt

odd:Cat har en m1:Mouse och en bird:Toy

- Sant ✔
- Falskt

Syskonen lau:Cat och leo:Cat leker med samma m2:Mouse

- Sant ✔
- Falskt

tofflan:Cat har en std::vector<Toy*> (Eller en java.util.Vector<Toy>), där de första 42 objekten är av typen Ball

- Sant ✔
- Falskt

lasse:Dog och tofflan:Cat leker med b4:Ball tillsammans.

- Sant ✔
- Falskt

Totalpoäng: 8

5 Design Patterns och Arkitekturstilar

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

I Strategy Pattern har en klass rollen kontext, vilket innebär att den är ansvarig för att hålla koll på vilken strategi som är aktuell för stunden och skicka vidare anrop från resten av systemet till den aktuella strategin.

- ○ Sant ✔
- ○ Falskt

Arkitekturstilen Layered går ut på att man delar in systemet så att varje systemhändelse utförs i en separat del av systemet med -- nästintill -- vattentäta skot mellan olika systemhändelser.

- ○ Sant
- ○ Falskt ✔

i Observer Pattern har klasser med rollen Observable en lista med objekt av typen Observer. När någonting relevant händer går Observable-klassen igenom listan och anropar var och en av sina Observers för att berätta att något hänt.

- ○ Sant ✔
- ○ Falskt

I designmönstret State Pattern sparar man värdet på alla attribut (ett objekts tillstånd) i en separat klass.

- ○ Sant
- ○ Falskt ✔

Model View Controller är ett arkitekturmönster för applikationer med flera olika sätt att visa en underliggande datamodell.

- ○ Sant ✔
- ○ Falskt

Det får bara finnas en instans (ett objekt) av en Singletonklass i ett system.

- ○ Sant ✔
- ○ Falskt

Totalpoäng: 6

6 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

Strategy pattern använder sig av polymorfism

- ○ 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 syste,.

- ○ Sant ✔
- ○ Falskt

Polymorfism hade inte fungerat i Java och C++ utan arv mellan klasser.

- ○ Sant ✔
- ○ Falskt

Totalpoäng: 7

Betygsgränser

Betygsgränserna för denna tenta är:

Betyg	Procent	Poäng
MAX	100%	47
A	90%	42
B	80%	38
C	70%	33
D	65%	31
E	60%	28

Lycka till!