2022-03-21.kl.08-12

Tentaupplägg

Allmänna Tips

Läs igenom ALLA uppgifterna. Välj den du känner är lättast först. Det kan gärna ta 10-20 minuter. Försök skriva saker som kan vara problem i uppgifterna. Är det något du absolut kommer att fastna på så kanske det är fel uppgift att ge sig på. Tiden du lägger på att noga läsa uppgifterna tjänar du in på att välja rätt uppgift. Kolla ibland till kommunikationsfönstret. Det kan ha kommit information till alla utan att ni skickat in en fråga. Om ni har problem med eclipse eller dylikt som INTE har med uppgifterna att göra, räck upp handen så kommer en assistent. Detsamma gäller om hur man kopierar givna filer. Frågor om själva uppgifterna tar vi i första hand via tentasystemet. Testa ditt program noga innan du skickar in. Om din lösning inte är korrekt kommer du få ett kompletteringsmeddelande och har möjlighet att rätta den och skicka in igen. Kompletteringar måste dock åtgärdas. Vi tolererar inte att ni spammar oss med samma lösning om och om igen.

Kommando	***Effekt***
`cp ~/Desktop/given_files/* .`	Kopierar alla givna filer till mappen du står
`xdg-open given_files/tenta.pdf`	Öppnar själva tentan. Blir läsbar då tentan startar.
`eclipse &`	Startar eclipse.

Generella krav (gäller för alla uppgifter om inget annat anges):

- Fullständiga uppräkningar och kodduplicering skall undvikas med klasshierarkier och polymorfi i de fall det inte går att generalisera på andra sätt (underprogram, loopar, etc.).
- Korrekt inkapsling krävs, d.v.s. rätt synlighet på instans/klassvariabler och metoder.
- Lämpliga konstruktorer skall finnas för alla klasser.
- Instansvariabler får inte användas som globala/lokala variabler (d.v.s. använd parametrar och lokala variabler i första hand).
- Korrekt användning av static krävs.
- Överanvändning av ***instanceof*** och typkonverteringar kan leda till komplettering.

Betygsgränser:	**Tid**	**TDDE10/11**	**725G90**
1 poäng	12:00	Betyg 3	Betyg G
3 poäng	11:00	Betyg 4	
3 poäng	10:40		Betyg VG
3 poäng	10:20	Betyg 5	
≥4 poäng	11:00	Betyg 5	
≥4 poäng	11:20		Betyg VG

Förlängd tid: +90 minuter på gränsen för 3/G, +45 minuter på övriga gränser. Bonustid från labbar tillkommer till dessa gränser (+5 minuter per B). Tiden för betyg 3/G överstiger dock aldrig 4 timmar.

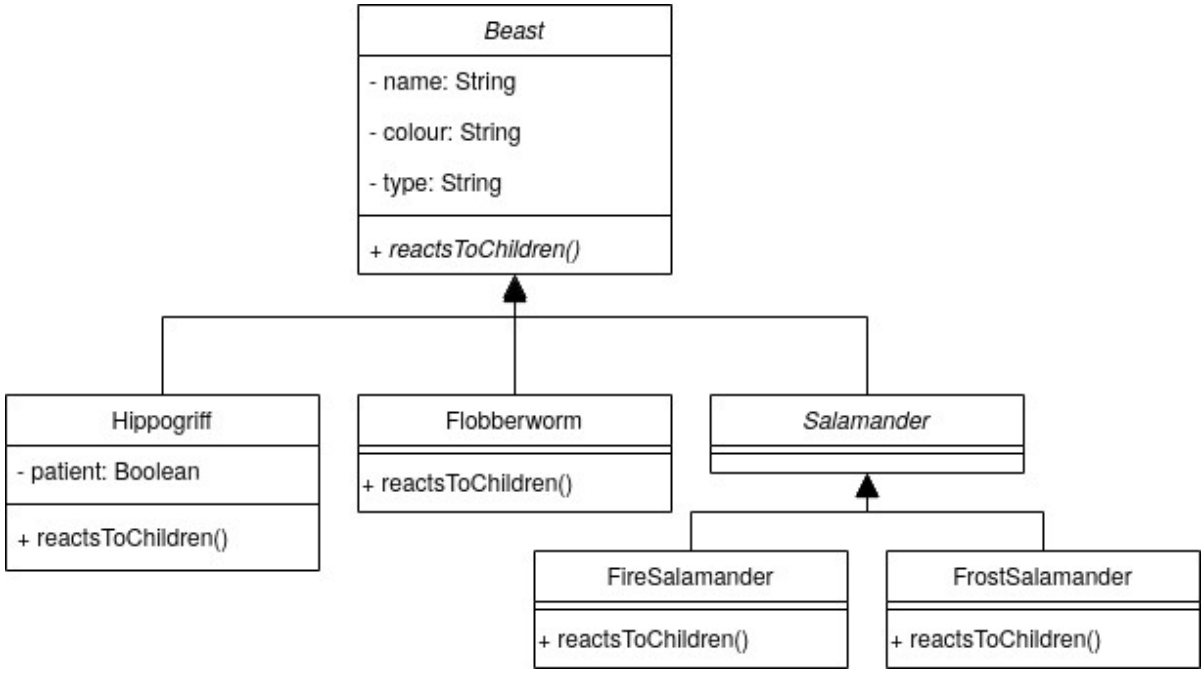
OBS: Delpoäng delas inte ut på uppgifterna. För att få poäng på en uppgift måste man alltså lösa uppgiften helt (och enligt specifikation).
M.v.h.
// Magnus och Emma

2022-03-21.kl.08-12

Uppgift 1 - Hagrids kursplan [1p]

Hagrid är lärare för kursen “skötsel och vård av magiska djur” på Hogwarts. Han håller på att planera kursplanen för tredje års-eleverna och vill ha med så många roliga djur som möjligt i sin kursplan.

Implementera följande klasser som visas i UML-diagrammet:



Krav: Konstruktorerna för subklasserna till Beast skall enbart ta *name* och *colour* (och i Hippogriffens fall *patient*). *Type* skall initieras i Beast-konstruktorn.

Krav: I de olika klassernas *reactsToChildren()* ska ett meddelande komma ut på skärmen om hur djuret reagerar på barn. Det skulle exempelvis kunna vara “just wants to be left alone.” för en Flobberworm. Hippogriffen ska reagera olika i *reactsToChildren()* beroende på om den har tålamod eller inte (värdet på *patient*).

I TestBeasts.java finns kod som testar din klassstruktur.

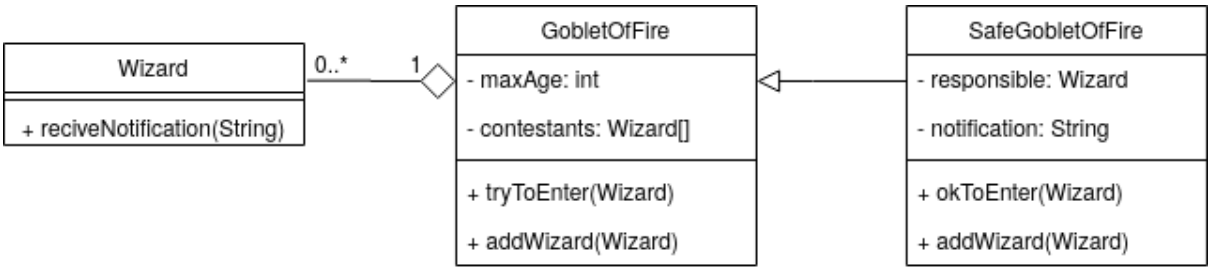
Körexempel:

```
Hagrids curriculum for third year students is:
Buckbeak the stormy grey hippogriff: if the children are rude, Buckbeak will be rude back.
Chestnut the sluggish green flobberworm: Chestnut just want to be left alone.
Smoke the ember fire salamander: Smoke might lightly burn a child or two.
Snow the frost blue frost salamander: Warning: Snow might cause frostbite!
```

Uppgift 2 - HARRY! DID YOU PUT YOUR NAME IN THE GOBLET OF FIRE?!?! [1p]

Under Harry Potters fjärde år på Hogwarts blev han utvald att delta i ett magisk mästerskap trots att han själv aldrig anmält sig. Anmälan skedde genom att folk över 17 år fick lägga sitt namn i en flammande bägare. Harry, som vid tillfället var under 17 år, kunde inte själv lägga sitt namn i bägaren. Någon annan lyckades dock förvirra bägaren med en trollformel och kunde därför lägga i Harrys namn.

Den flammande bägaren representeras av klassen GobletOfFire. Som ledtexten ovan beskriver finns det problem med bägaren. När en trollkarls namn vars ålder är under gränsen läggs i bägaren av en trollkarl som är över åldern kraschar klassen GobletOfFire. I denna uppgift skall du skapa en ny klass, SafeGobletOfFire, som förbättrar den givna klassen.



Klasserna GobletOfFire och Wizard finns i given_files och skall **INTE** ändras.

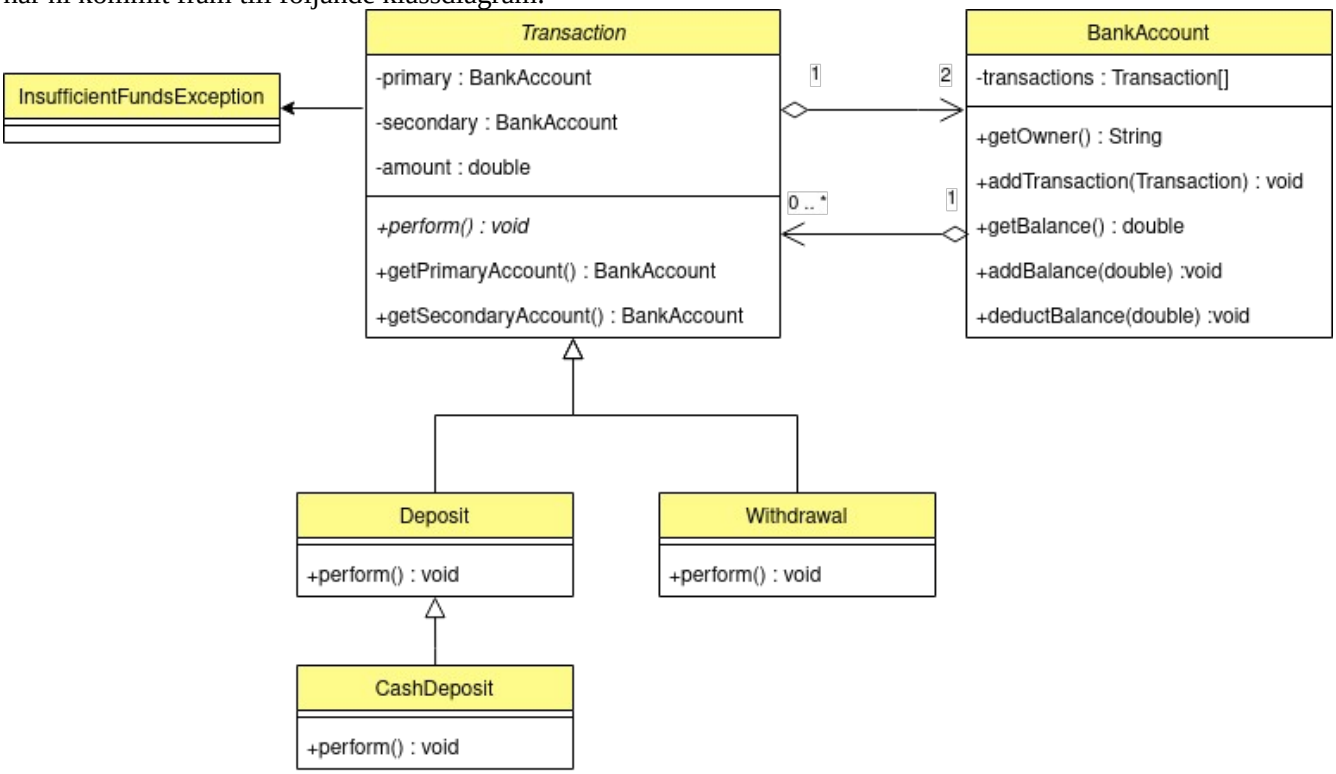
När man instansierar en GobletOfFire så måste man ange ett minvärde som anger vid vilken ålder bägaren ska kasta ett exception. Klassen har två metoder. Metoden *tryToEnter(Wizard)* kollar om en trollkarl får komma i närheten av bägaren för att kunna lägga in ett namn i den. Metoden *addWizard(Wizard)* anmäler en trollkarl till uttagningen. Om trollkarlen som skickas in till någon av metoderna är under minvärdet kastas YoungWizardException. Även detta Exception måste göras.

Din uppgift är att skapa en ny klass SafeGobletOfFire som förbättrar GobletOfFire. Från SafeGobletOfFire skall det inte längre vara möjligt att YoungWizardException kastas. Om trollkarlen som lägger i lappen eller trollkarlen som står på lappen är för unga skall en notifikation till den ansvariga för tävlingen skickas. Vilken trollkarl som har ansvaret anges när SafeGobletOfFire instansieras. Notifieringen är en sträng t.ex. "Something fishy is going on with the Goblet of Fire!" som också anges vid instansieringen av SafeGobletOfFire. Metoden *okToEnter* ska använda sig av *tryToEnter* från superklassen.

Det finns ett givet testprogram TestGobletOfFire.java som ska gå att köra när du är klar med din klass. Du skall inte behöva ändra på detta testprogram.

Uppgift 3 – Gringotts trollkarlsbank [2p]

Svartalferna på Gringotts trollkarlsbank har tröttnat på all handskriven bokföring och har beslutat sig för att införskaffa datoriserade system. Då Hogwarts ännu inte har några programmeringskurser har du tack vare dina javakunskaper fått inresetillstånd, trots att du är mugglare, så att du kan hjälpa dem bygga systemet. Tillsammans har ni kommit fram till följande klassdiagram:



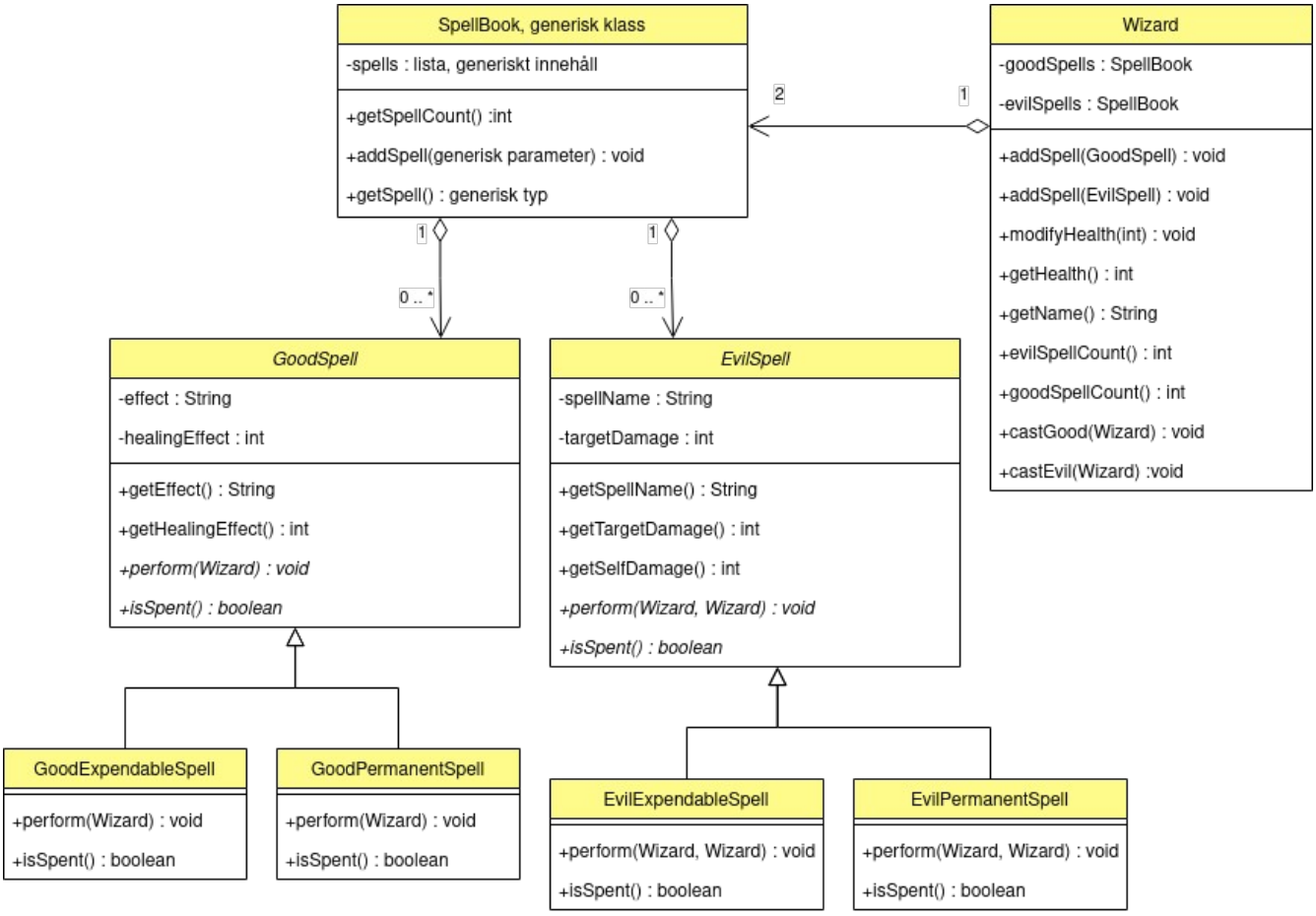
Implementera klasserna i klassdiagrammet. Klassen GringottsTest.java är given och får ej modifieras. Den hjälper dig se hur klasserna ska instansieras och användas. Röda utskrifter när testerna körs beskriver något som inte står rätt till. Om utskrifterna ser ut som nedan är du redo att lämna in (inte nödvändigtvis godkänd):

Körexempel:

```
Got an expected exception when Ron Weasley attempted to withdraw more money than they have
Got an expected exception when Ron Weasley attempted to transfer more money than they own to
Harry Potter
Harry Potter was allowed to transfer all their money to Ron Weasley. Harry Potter now has:
0.0, Ron Weasley has: 1500.0
Harry Potter successfully deposited 2000.0 in cash, and now has: 2000.0
```

Uppgift 4 – Roligt med trollformelsböcker [3p]

Dina kompisar på Hogwarts har hittat en massa roliga trollformelsböcker som ni givetvis måste undersöka närmare genom att testa dem. Då du har lite högre säkerhetstänk än dina kamrater vill du först simulera de tänkbara konsekvenserna. Du har kommit fram till följande klassdiagram:



Givet finns ett testprogram, SpellbookTest.java, samt Wizard.java. Wizard-klassen är inte riktigt klar, utan saknar Spellbook-variablerna samt cast-metoderna. Din uppgift är att bygga färdigt denna klass samt de övriga i diagrammet. Du kan behöva lägga till variabler, eller göra extra metoder för att exempelvis undvika kodduplicering, men testprogrammet ska gå att köra. Om du får röda utskrifter från testprogrammet fungerar det inte som tänkt ännu. Det finns tre stora tester, se kommentarer i koden för hur de bör se ut. EvilSpell ska göra 1/3 av skadan på den som kastar trollformeln, full skada på dem som är måltavla.

Körexempel:

```
The power of infinite motivational memes inspires Emmnus, healing 0 points, before it
vanishes.
Magemm has taken 1 points of damage from Emmnus's itchy clothing spell, before it vanishes.
The feeling of great confidence boosts Emmnus, healing 1 points, before it vanishes.
Magemm has taken 2 points of damage from Emmnus's painfully tight shoes spell, before it
vanishes.
The power of infinite caffeine fuels Emmnus, healing 5 points, before it vanishes.
Magemm has taken 7 points of damage from Emmnus's fire rain spell, before it vanishes.
Magemm has taken 15 points of damage from Emmnus's gaggle of venomous snakes spell, before
it vanishes.
```