HÖGSKOLAN
I GÄVLE

AKADEMIN FÖR TEKNIK OCH MILJÖ
Avdelningen för industriell utveckling, IT och samhällsbyggnad

# Objektorienterad design och programmering
## Projektinstruktioner

Version 1 oktober 2015, perjee

# Innehåll

# 1 Projektförslag

## 1.1 Kasino

*”Kasino är ett kortspel för två till fyra personer. Jag har själv lärt mig det från min farmor och farfar, och i Åsa Linderborgs aktuella bok Mig äger ingen (Atlas, 2007) förekommer följande:*
När jag blev stor nog för kortspel tog jag ett parti kasino med farfar.
–Tabbe!
–Vad tog du med för kort? Knekt? Är sju plus sex elva?
–Va? Sir’u jänta, jag trodde det var en kung!

*Kasino spelas med en vanlig korlek (52 kort). Spelet börjar med att varje spelare får fyra kort och fyra kort vänds upp på bordet. Sedan får spelarna i tur och ordning antingen ”ta” kort från bordet eller ”lägga ut” kort. När man tar kort samlar man på sig poäng som i slutet av en spelrunda räknas ihop. Spelet går ut på att få så många poäng som möjligt.*

*Efter en giv spelar alla spelarna ut sina fyra kort i tur och ordning. Därefter får varje spelare fyra nya kort, men det läggs inga nya på bordet. Och så fortsätter man tills korten är slut. (Det blir tre givar ifall man är fyra spelare: 3 givar * 4 spelare/giv * 4 kort/spelare + 4 kort på bordet från början = 52 kort).*

(Tack till Fredrik Bökman som skrev texten hösten 2007!)

Uppgiften är att skapa ett program som ger en spelare möjligheten att spela Kasino mot datorn, eller flera spelare att spela mot varandra.

Programmet ska ha ett grafiskt användargränssnitt. Korten som ligger på bordet ska ritas ut på ett ’spelbord’ och det ska synas hur många kort varje spelare har på sin hand. En spelare ska kunna dra sina egna kort och korten på bordet med hjälp av muspekaren.

Poängberäkningen ska skötas av programmet.

I grundversionen kan datorspelaren spela ut sina kort per slump.

Utvidgningar:

(1) Utveckla olika strategier som kan användas av datorspelaren (enkel, lagom smart, intelligent, ...?). Användaren ska kunna välja vilken strategi datorspelaren ska spela efter.

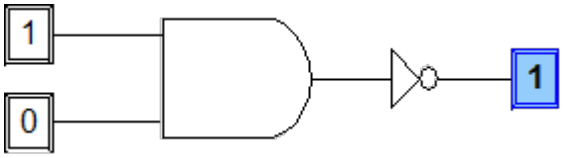
Följ strategy-mönstret!

(2) Flera spelare ska kunna spela tillsammans, men från var sin dator, alltså via nätverk.

Detaljerade spelregler finns att läsa i bilagan, avsnitt A.2.

## 1.2 Logiska kretsar

Ett program för att simulera enkla logiska kretsar. Definiera några enkla byggstenar (AND, OR, NOT) och skapa grafiska representationer för dessa. Det skall gå att hämta de enkla komponenterna från en meny (eller liknande), lägga in dem på en rityta, dra anslutningar mellan komponenter samt definiera input och output. Programmet skall kunna bestämma output för olika kombinationer av binära input.



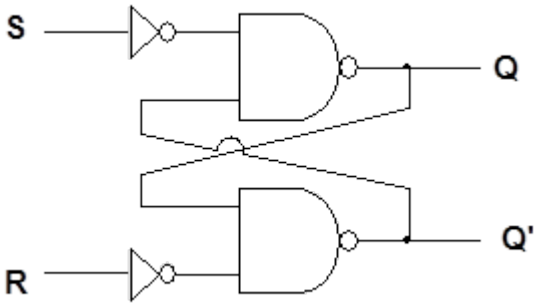

Figur 1: Exempel för logik-simulering.

Sammansatta komponenter bör gå att spara under egna namn (t.ex. NAND), eventuellt ges egna symboler, och sedan kunna användas som byggkomponenter. Se till att sammansatta komponenter kan sparas mellan programkörningar.


Figur 2: Möjliga symboler för NAND.

Tips: Om man har ett objekt kan man få en kopia med metoden clone().

Det räcker att du hanterar tidsoberoende logiska kretsar. Om du vill göra programmet intressantare hanterar du sekventiella kretsar där ut-tillståndet beror på en serie av in-tillstånd. Den enklaste sekventiella logiska kretsen är en SR-flip-flop:




Figur 3: SR-flip-flop.

Ännu intressantare blir det ifall du inför en klockfunktion. Men fokusera på att få det enkla fallet att fungera först!

Utvidgning:

Inför någon form av automatisk tillsnyggning av ritningen, så att komponenter ordnas snyggt horisontellt och vertikalt samt att antalet korsande linjer minimeras. (Nöj dig med en enkel lösning, att göra det här bra är mycket komplicerat.)

(Författare: Jonas Boustedt, Fredrik Bökman)

## 1.3 Kartapplikation

Ett program som ska hjälpa att planera speditionsföretag, hemtjänst, post o. dyl. att planera färdrutter.

Applikationen ska visa en karta, och med kartan som utgångspunkt ska användaren kunna skapa en vägmodell:

Modellen ska innehålla positioner i kartan som är kopplade till varandra med vägar, dvs. man kan åka från en position $A$ till en position $B$ som är kopplad till $A$. Varje koppling har ett attribut 'avstånd'.

Från position $B$ kan man möjligtvis åka till position $C$ och därifrån till en annan position. Dessutom är det möjligt att man kan nå ett flertal positioner från ett ställe.

Vägmodellen är alltså en abstrakt beskrivning av en del av gatunätet som visas i kartan.

Funktioner som ska finnas i programmet är

- Visa kartan, dvs. rita ut en digital bild som innehåller kartbilden.
- Visa vägmodellen, dvs. rita ut positioner som punkter och kopplingar mellan positioner som linjer till en given karta.
- Visa kortaste vägen från en position $A$ till en annan position $B$ i en given karta/modell.
- Definiera en modell: Användaren skapar en ny position och knyter ihop den med en existerande eller markerar en existerande position och drar en förbindelselinje till en annan existerande position.
- Spara modellen i en fil.
- Hämta kartbild resp. vägmodell från filsystemet.

Använd funktionerna som Java-biblioteket *JGraphT* erbjuder (finns i ZIP-filen `jgrapht-0.9.0.zip` på Blackboard, se även `http://jgrapht.org/`). Bl.a. finns i biblioteket klasser som kan användas för att modellera grafer och utföra operationer på graf-objekt, t.ex. att hitta kortaste vägen med hjälp av Dijkstras algoritm.

Utvidgning:

Använd en databas för att lagra vägmodeller. Databasen kan kopplas till programmet via JDBC. Alternativt kan ni använda 'Data Access Objects' (DAO).

# 2 Allmänt

Projektuppgifterna ska lösas i grupper (max. 2 studenter).

Projektet ska dokumenteras i en rapport (se bilagan, avsnitt A.1). Rapporten ska innehålla en beskrivning av programmets struktur samt UML-diagram. Ni ska även motivera era beslut när det gäller programmets design. Ifall ni lagrar data i filer resp. databaser, så måste filernas uppbyggnad eller databasmodellen dokumenteras. Dessutom behövs en beskrivning hur programmet kan användas.

Koden ska vara formaterad på ett enhetligt sätt. Till koden ska ni leverera en dokumentation i HTML-format (använd javadoc). Använd Junit-tests för att testa era klasser (testen är en del av koden som ni ska lämna in).

Lämna in kod, dvs. Eclipse-projektet i en ZIP-fil, samt dokumentation och rapport via Blackboard senast på söndag, 2015-11-01 innan kl. 23:59 och presentera projektet inför kursen under ett projekt-seminarium (fredag, 2015-11-06, 13:15 - 15:00). Närmare information om seminariet kommer att publiceras i Blackboard.

Ifall ni inte kan lämna in i tid så kom överens med läraren om ett annat redovisningstillfälle.

Om ni inte lämnar in era resultat utan rimlig anledning eller inte redovisar era resultat vid tillfället som ni kom överens med läraren om, så blir projektet underkänt. Ni hänvisas i så fall till nästa examinations-tillfälle, dvs. nästa gång när kursen ges igen.

**Observera att det inte är tillåtet att kopiera andras lösningar!**

**Ifall det används material (såsom kod, texter eller bilder) i program eller rapport som inte skapades av studenterna i gruppen, så är det ett krav att källorna redovisas!**

# 3 Betygsättning

Möjliga betyg: A, B, C, D, E, F

Betyg sätts individuellt, dvs. medlemmar i en och samma grupp kan få olika betyg på projektet.

För betygen B och A krävs att projektet redovisas på utsatt tid.

Följande kriterier används för betygsättningen och betygsätts med 0, 1 eller 2 poäng var:

1. Koden kan kompileras.
2. Programmet kan exekveras.
3. Koden är formaterad på ett enhetligt sätt.
4. Rapporten är utformad enligt instruktionerna.
5. Koden är skriven enligt vedertagna konventioner (namn etc.)
6. Koden struktureras med hjälp av paket.
7. Kraven som ställs gentemot programmet beskrivs tydligt i rapporten.
8. Programmet uppfyller kraven som anges.
9. Designen dokumenteras med hjälp av UML-diagram.
10. Beslut angående programmets design motiveras i rapporten.
11. Koden dokumenteras med hjälp av `doc`-kommentarer i koden.
12. Junit-tests används för att testa programmets komponenter.
13. Undantag resp. fel som kan uppstå under exekveringstiden hanteras - programmet svarar under alla omständigheter. Det tas hänsyn till eventuella varningar som kompilatorn genererar.
14. Paketstrukturen återspeglar programmets arkitektur.
15. Programmets design visar en tydlig gränsdragning mellan modell/logik/användargränssnitt.
16. Det används `javadoc` för att generera en API-dokumentation.
17. Resultaten utvärderas i diskussions-delen till rapporten.
18. Programmets design följer MVC-mönstret.
19. Programmet erbjuder funktionalitet som beskrevs under 'utvidgning' i respektive projektbeskrivning.

Sammanlagt är det möjligt att få upp till 38 poäng. Följande skala används för att sätta betyg:

- A: 35 - 38 poäng
- B: 31 - 34 poäng
- C: 26 - 30 poäng och kraven för D uppfylls.
- D: 21 - 25 poäng och kraven för E uppfylls.
- E: 16 - 20 poäng och minst en poäng till varje av kriterierna 1 till 12.
- F: 0 - 15 poäng

Bonuspoängen från laborationerna kan användas i betygsättningen om projektet själv betygsätts med E eller bättre och om laborationerna och projektet lämnades in vid ett och samma kurstillfälle.

# A Appendix

## A.1 Mall för rapporter

Rapporten ska vara strukturerad på följande sätt:

1. Titel, författarna.
2. “Inledning”. (Problembeskrivning - vad handlar projektet om.)
3. “Förutsättningar och krav”. (På programmet som ska utvecklas.)
4. “Resultat”. (Beskrivning av lösningen, dvs. implementering och test. Resultat av testkörningar.)
5. “Diskussion”. (Finns det alternativ till lösningen som ni valde? Vilka fördelar har de om ni jämför? Vilka nackdelar?)
6. “Sammanfattning”.

## A.2 Spelregler för Kasino

*Man ”tar” kort på följande sätt:*

- Ifall du har en sjua på handen och det ligger en sjua på bordet kan du ta den. Båda korten hamnar då i din hög med ”tagna kort” (du tar inte upp dem på handen).
- Ifall du har en sjua och det ligger en trea och en fyra på bordet kan du ta båda två med sjuan.
- Ifall du har en sjua på hand och det ligger 2, 3, 4, 5 ute på bordet kan du ta alla fyra korten (2+5 samt 3+4).
- Ifall du tar alla kort som finns på borde tar du en ”tabbe” , det ger extra poäng.
- Ifall du inte kan ta något måste du lägga ut ett kort på bordet. Då blir de andra spelarna glada eftersom de har chansen att ta det kortet.
- Du får inte använda mer än ett kort åt gången från din egen hand.

  Exempel: På bordet ligger 5, 8, 9, 10 och du sitter med två sjuor på handen. Du måste lägga ut ena sjuan. (Och du kan hoppas att den ligger kvar tills det är din tur nästa gång.) Du kan inte addera dina sjuor till 14 och med den summan ta 5+9, additionen sker bara på bordets kort.
- Du behöver inte ta ifall du inte vill, du kan (om du är modig) välja att lägga ut istället.
- Ess är värt ett eller fjorton, vad som passar bäst för den spelaren som är på tur.

*Poängberäkning:*

- 1 p för varje ess
- 1 p för varje tabbe
- 1 p för ”lillan”, spader två
- 2 p för ”storan”, ruter tio
- 1 p till den spelare som har tagit flest kort (1 p var ifall flera spelare)
- 1 p till den spelare som har tagit flest spader (1 p var ifall flera spelare)
- 1 p extra till till den spelare som både har flest kort och flest spader.

*Spelets vinnare utses på något av följande sätt:*
*(a) Man spelar en kortlek och räknar samman poängen, flest vinner.*
*(b) Man spelar ett i förväg angivet antal gånger (t.ex. tre kortlekar), flest poäng vinner.*
*(c) Man spelar upp till ett i förväg angivet antal poäng, t.ex. 15.*

*Ett exempel:*

*Jag är spelare 1 och har följande kort på hand: ruter 3, spader 5, hjärter 5, klöver knekt.*

*På bordet ligger spader dam, hjärter dam, klöver 5, spader 8.*

*Jag har alltså följande tre alternativ:*
*(1) Ta spader 5 med min klöver 5.*
*(2) Ta hjärter 5 med min klöver 5.*
*(3) Ta inget, lägg ut något av mina fyra kort på bordet.*

*Det säkra valet är att ta spader fem. En chansning vore att lägga ut hjärter fem och hoppas att bägge femmorna ligger kvar näst gång det blir min tur, så att jag kan ta båda. Det gör jag inte, jag tar det säkra före det osäkra.*

*Efter att jag har tagit femman finns alltså spader dam, hjärter dam och spader 8 på bordet.*

*Spelare 2 har klöver dam, tar båda damerna som finns på bordet.*
*Spelare 3 lägger ut hjärter 2.*
*Spelare 4 lägger ut hjärter ess.*

*Nu är det min tur igen. På bordet finns spader 8, hjärter 2 och hjärter ess. Jag har på handen ruter 3, hjärter 5 och klöver knekt.*

*Mina alternativ är nu*
*(1) Ta tvåan och esset med trean.*
*(2) Ta 8 + 2 + 1 med knekten.*
*(3) Inte ta något alls utan lägga ut ett av mina kort.*

*Alternativ (2) är klart bäst. Jag tar alla tre korten med min klöver knekt, och ler ett försmädligt leende mot spelare 2 som nu inte kan göra något annat än att lägga ut ett kort på det tomma bordet.”*