CardGame, januar 2019
Udarbejdet af Päivi Eversbusch, DATC18

Kære Asger,

Jeg har kodet kortene. Jeg synes, der er ret meget ansvar at dele, så det blev dog flere end to klasser. Jeg har øvet mig både på koder og design i løbet af Interregnum (skønt ord!), men fordi jeg stadig har nogle ting, jeg ønsker at nå inden næste uge, har jeg valgt ikke at lave designdelen i denne opgave. Opgaven var til at overskue, så jeg følgte, jeg kunne klare mig med klassediagremmet. Selv om det ikke er det samme, selvfølgelig.
Jeg skriver her mine overvejelser og vedhæfter et klassediagram, men ellers er det koderne, som stod i centrum for mig i denne opgave. Det ville være rart at få lidt feedback på dem.

- Der er 52 kort og ingen jokere
- Kortene har værdier fra 1 til 13, dvs. es har værdien 1. Selv om knægt, dame, kong og es har talværdier, bliver de printet med betegnelserne knægt, dame, kong og es for at gøre det mere overskueligt for brugeren.
- Programmet er lavet således, at man får ét sæt kort, som man spiller med. Så det er det samme sæt, som man ændrer værdier på. I den virkelige verden ville man heller ikke tage et nyt sæt i hver omgang. Hvis man vil have et nyt ublandet sæt, kan man afslutte programmet og begynde forfra.
- Antallet af kort er fastsat til 52 og kortene har altid de samme værdier, så CRUD er ikke rigtig til at finde i systemet. Man kan ikke slette et kort fra et sæt, man kan ikke ændre værdier og man kan ikke lave nye kort ud over de 52.
- Der er lavet javadoc på koderne
  file:///C:/C%20DATAMATIKER%202018/JGRASP/2019/CardGame/CardGame_doc/KoererSystem.html
  (Det lykkedes mig ikke at vedhæfte dette linktil mappen i GitHub, beklager..)

Kuloer
- enum med de fire kulører
- final værdier, så skrives med stort og static, så de kan bruges af alle klasser

Kort
- holder på attributterne på kort: kulør og værdi
- toString er lavet sådan, at es, knægt, dame og kong bliver printet med disse betegnelser frem for tal

EtSaetKort
- her laver jeg et sæt kort bestående af 52 kort (et array af datatypen Kort). I konstruktøren giver jeg kortene værdier opdelt per kulør.
- I klassen har jeg både getKort() og getSaet(), fordi når jeg bruger sættet, har jeg nogle gange brug for hele sættet og andre gange et enkelt kort.

SaetGenerator

- Her laver jeg et sæt kort i konstruktøren. Jeg har lavet composition, fordi jeg ikke synes, det giver mening at have et sæt kort, hvis man ikke behandler det på nogen måde. Så kunne man lige så godt bare beholde de enkelte kort og give dem værdier. Så hvis jeg sletter SaetGenerator, kan jeg lige så godt slette EtSaetKort. Jeg ville dog stadig kunne lave enkelte kort.
- Denne klasse har ansvar for at blande kortene og her ville man kunne tilføje andre metoder, som man kunne bruge til at behandle et sæt kort med.
- Jeg synes, at blanding af kort er afgørende i kortspil for at man kan stole på en retfærdig fordeling af kortene. Dermed er dette det vigtigste punkt i systemet og her ville jeg lave fully dressed use case.
  Jeg har lavet forskellige metoder til at blande kortene (parameter er et sæt kort), fire i alt. En af metoderne er baseret på en random blanding, men ellers efter et bestemt mønster.

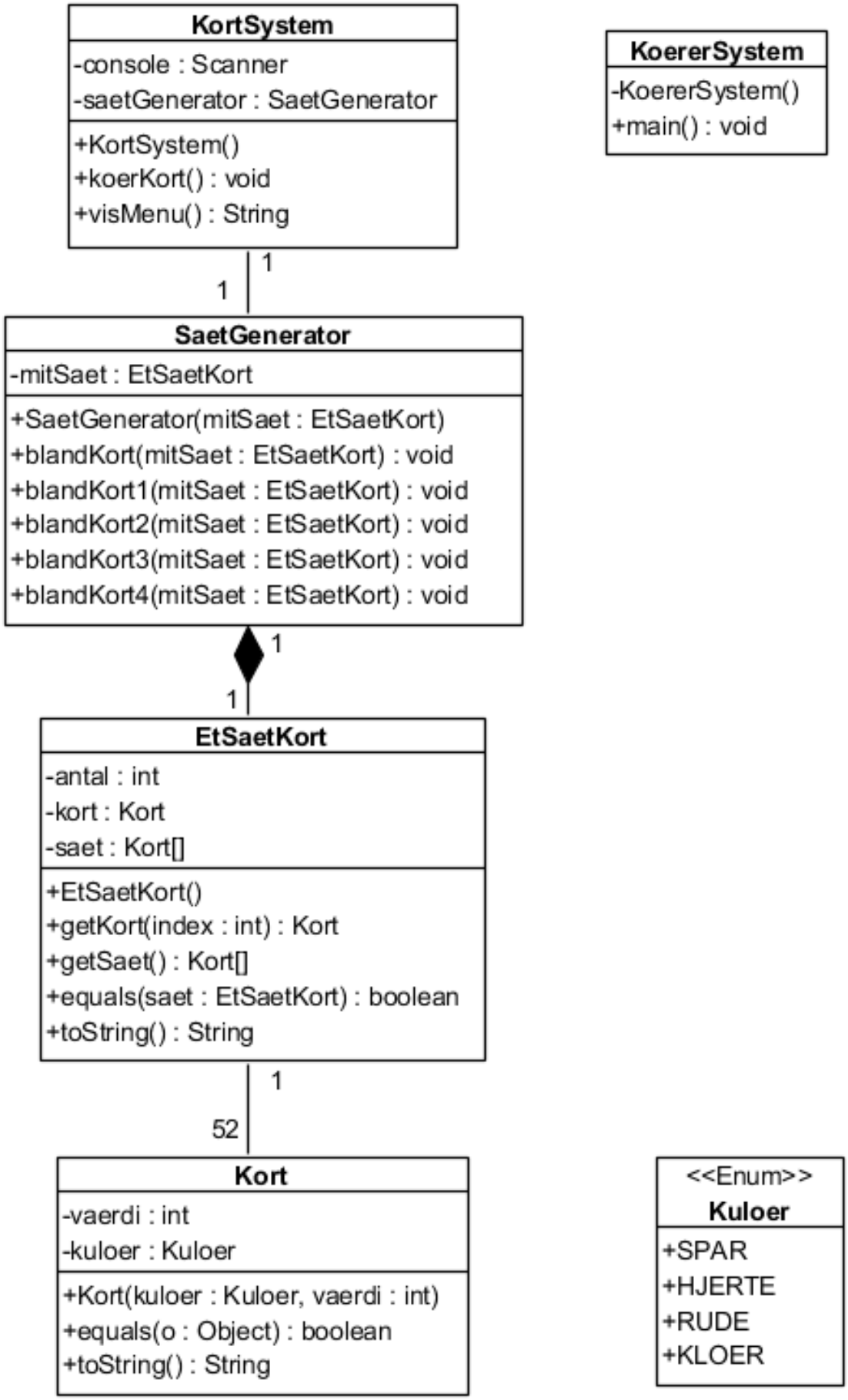
  Metoderne kan bruges hver for sig (det kunne jo være, at man havde brug for et bestemt mønster), men jeg har samlet dem til metoden blandKort, hvor man får kørt alle fire blandmetoder efter hinanden 30-100 gange (antallet af omgange er bestemt af et random tal). På den måde kan man være helt sikker på, at man hver gang får et forskelligt sæt.

KortSystem

- Her laver jeg en instans af SaetGenerator, som jeg bruger i menuen.
- Man har forskellige valgmuligheder og jeg har lavet koden således, at den gerne skulle tage højde for alle eventuelle forkerte indtasninger (negative tal og bogstaver, hvor man skal indtaste et tal)
- Jeg læser menupunkterne med charAt(0), fordi menuen er lille. Hvis jeg skulle udvide menuen til over 9 punkter (dvs. flere end et tegn), skulle jeg så justere denne måde at læse valg på.

KoererSystem

- Her har vi main-metoden, hvor jeg laver instans af KortSystem og kalder på menuen.
- Konstruktøren er privat, fordi jeg ikke vil have lavet nogen instanser af denne klasse. Så jeg minimerer det til højest én, hvis uheldet skulle være ude.


KortSystem
-console : Scanner
-saetGenerator : SaetGenerator
+KortSystem()
+koerKort() : void
+visMenu() : String
KoererSystem
-KoererSystem()
+main() : void
1
1
SaetGenerator
-mitSaet : EtSaetKort
+SaetGenerator(mitSaet : EtSaetKort)
+blandKort(mitSaet : EtSaetKort) : void
+blandKort1(mitSaet : EtSaetKort) : void
+blandKort2(mitSaet : EtSaetKort) : void
+blandKort3(mitSaet : EtSaetKort) : void
+blandKort4(mitSaet : EtSaetKort) : void
1
1
EtSaetKort
-antal : int
-kort : Kort
-saet : Kort[]
+EtSaetKort()
+getKort(index : int) : Kort
+getSaet() : Kort[]
+equals(saet : EtSaetKort) : boolean
+toString() : String
1
52
Kort
-vaerdi : int
-kuloer : Kuloer
+Kort(kuloer : Kuloer, vaerdi : int)
+equals(o : Object) : boolean
+toString() : String
<<Enum>>
Kuloer
+SPAR
+HJERTE
+RUDE
+KLOER