30/10-2020

DTU SOFTWARETEKNOLOGI


# CDIO_2

## Gruppe 18



Aya Fahim Yousif: S205466 Mikail Oguzhan Kocak: S195205 Shoba Parasi: S205476



Alec Ranjitkar : S205427 Yazan Mahmoud Bayazid: S181539 Jawahir Farah Shahal: s205414

## Timeregnskab

| Studerende | S205466 | s195205 | s205476 | s205427 | s205414 | s181539 |
|---|---|---|---|---|---|---|
| Timer | 8 | 9 | 7 | 7 | 6.5 | 6.5 |

## Abstract

This project is a game that consists of two players competing against each other. Every round each player throws 2 dice, where each die shows a value from 1-6. There are 11 fields from field number 2-12, at each field the player can either lose or gain points. The winner of the game is the first player to reach 3000 points. After each round both players return to their starting positions, hence there is no continuation from the earlier rounds regarding the placement of the players on the board, only regarding the total points of the players. Among other requirements, the game has been required to work on the Windows data bars at DTU campus. To succeed the game has been developed while considering the costumer's requirements. Lastly, we tested the game on the Windows data bars at DTU campus to ensure the requirement is met, and we have made several other successful tests to further insure that game works properly.

## Contents

# Indledning

Der skal implementeres et spil i databaserne på Danmarks Tekniske Universitet. Spillet skal kunne spilles mellem to personer, hvor der kan kastes med to terninger. Den samlede værdi af terningerne bruges (2-12), viser nummeret på det felt de lander på under runden. Ved hvert felt kan spillerne enten miste eller modtage point, få en ekstra tur eller modtage intet. Hvert felt repræsenterer et tema og hver gang en spiller lander på et felt, udskrives en oplysende tekst om feltet. Hver spiller starter med 1000 point, derefter spilles der indtil en af spillerne opnår 3000 og dermed vinder spillet.

Først er der i gruppen blevet diskuteret måder at skrive koden på, for at det er nemt at skifte terningerne ud. Dette er blevet gjort vha. Low Coupling. Dernæst er der lavet et use case diagram for at beskrive spillets trin, og for have et overblik over *the main flow*. Ydermere er der lavet en domain model, som er med til at visualiserer en grundlæggende struktur for spillet og spillets koncepter.

Endvidere er der dannet et overblik for det objektorienterede design, ved at lave et sekvensdiagram og et desigklassediagram. med et formål om at definere software objekter og deres samarbejde og ansvar (Larman, 2004).

Koden er udviklet i JavaScript i InteliJ. Derudover er der vha. Git skabt et remote repository på GitHub, til at holde styr på forskellige commits lavet af de forskellige gruppemedlemmer. Til sidst er spillet blevet testet ved forskellige JUnit tests, og på maskinerne i DTU's databarer.

# Kravliste

Kravliste dannet ud fra kundens vision for produktet.

1. Spillet skal indeholde 2 sekssidet terninger, som viser øjnene 1-6.

2. Spillet skal spilles af to personer, og det skal virke på maskinerne i DTU's databarer uden bemærkelsesværdige forsinkelser.
3. Hver runde starter spillerne på startfeltet. Der er felter fra 2-12.
4. Der skal udskrives en tekst omhandlende det aktuelle felt.
5. Hver spiller skal starte med 1000 point og en spiller skal vinde ved at få 3000.
6. Det skal være let at skifte til andre terninger.
7. Spillernes balance må ikke blive negativ.

## Analyse

Før udviklingen af spillet, har vi udført en analyse ved brug af et use case diagram, domænemodel og et sekvensdiagram. Ved hjælp af disse ting, får man et overblik af selve projektet, og dette har til formål at formindske eventuelle misforståelser og minimere risikoen for fejl.

### Use Case

For at give et overblik over hvordan spillet kommer til at foregå, vil der først blive beskrevet en række use cases i kort format, der hjælper til med at forstå spillets funktioner.

Dernæst vil der blive beskrevet en detaljeret use case, hvor der en række trin der bliver repræsenteret og samlet vil disse use cases afdække kundens krav til spillet.

**1.1. Start spil:**

Begge spillere starter med en pengebeholdning på 1000, og indtaster et navn der adskiller spiller 1 og spiller 2

**1.2 Kast terninger:**

Spillerne trykker på skift på en knap for at kunne kaste med to terninger, som viser to værdier mellem 1 til 6, de to værdiers sum mellem 2 til 12 repræsentere et felt man kan lande på i spillet.

**1.3 Felt Tower:**

Lander man på feltet Tower, har man slået med de to terninger og fået en samlet værdi på 2. Dette felt giver 250 point til pengebeholdningen og viser teksten "Du har fundet værdifulde genstande i et tårn og har solgt det for 250 point!"

**1.4 Felt Crater:**

Lander man på feltet Crater, har man slået med de to terninger og fået en samlet værdi på 3. Dette felt trækker 100 point fra pengebeholdningen og viser teksten ”Du faldt ned i et krater og mistede 100 point!

**1.5 Felt Palace Gates:**

Lander man på feltet Palace Gates, har man slået med de to terninger og fået en samlet værdi på 4. Dette felt giver 100 point til pengebeholdningen og viser teksten ”I din rejse besøgte du et palads, hvor kongen gav dig 100 point!”

**1.6 Felt Cold Desert:**

Lander man på feltet Cold Desert, har man slået med de to terninger og fået en samlet værdi på 5. Dette felt trækker 20 point fra pengebeholdningen og viser teksten ”Du er faret vild om natten i en kold ørken og tabte 20 point...”

**1.7 Felt Walled City:**

Lander man på feltet Walled City, har man slået med de to terninger og fået en samlet værdi på 6. Dette felt giver 180 point til pengebeholdningen og viser teksten ” Du fandt en indlukket by og som præmie for at finde den, modtager du 180 point!”

**1.8 Felt Monastery:**

Lander man på feltet Monastery, har man slået med de to terninger og fået en samlet værdi på 7. Dette felt giver ikke point til og trækker ikke point fra pengebeholdningen og viser teksten ”Du har fundet et kloster! Men der er dsv tomt, så du rejser videre...”

**1.9 Felt Black Cave:**

Lander man på feltet Black Cave, har man slået med de to terninger og fået en samlet værdi på 8. Dette felt trækker 70 point fra pengebeholdningen og viser teksten ”Du er oppe i bjergene, hvor du er inde i en mørk grotte. Fordi du intet kan se så taber du 70 point.”

### 1.10 Felt Huts in the Mountain:

Lander man på feltet Huts in the Mountain, har man slået med de to terninger og fået en samlet værdi på 9. Dette felt giver 60 point til pengebeholdningen og viser teksten ”Oppe ved bjergene finder du en masse hytter, som tilhører troldemænd! De giver dig 60 point til din rejse.”

### 1.11 Felt The Werewall:

Lander man på feltet Black Cave, har man slået med de to terninger og fået en samlet værdi på 10. Dette felt trækker 80 point fra pengebeholdningen, men giver spilleren en ekstra tur til at slå med terningerne og viser teksten ”Åh nej! Du finder en varulv! Den stjæler 80 point fra dig! Men du vinder en ekstra tur!"

### 1.12 Felt The Pit:

Lander man på feltet The Pit, har man slået med de to terninger og fået en samlet værdi på 11. Dette felt trækker 50 point fra pengebeholdningen og viser teksten ”Damn it! Du er faldet ned i et hul og har tabt 50 point...”

### 1.13 Felt Goldmine:

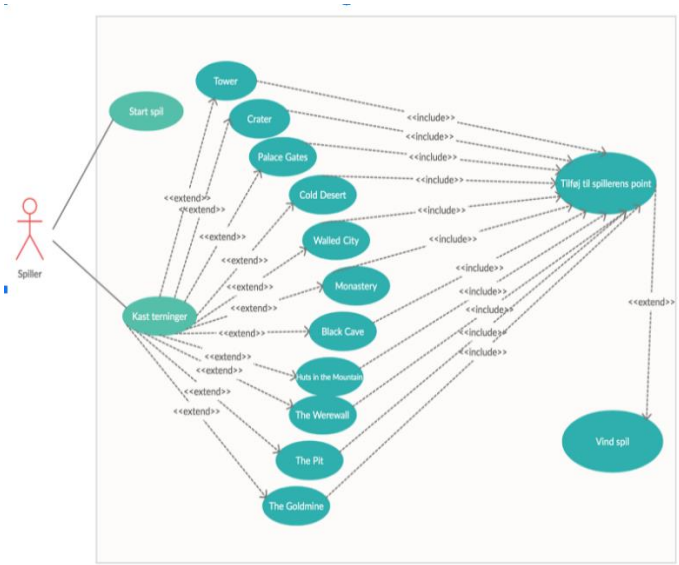
Lander man på feltet Goldmine, har man slået med de to terninger og fået en samlet værdi på 12. Dette felt giver 650 point til pengebeholdningen og viser teksten ”JUBI!!! Du har fundet en guldmine oppe på bjergene og har solgt guldet for 650 point!”

### 1.14 Spil igen:

Når spillet færdigt, bliver spillerne spurgt om man vil spille igen. Hvis de svarer ja, starter spillet igen. Hvis de svarer nej afsluttes spillet.

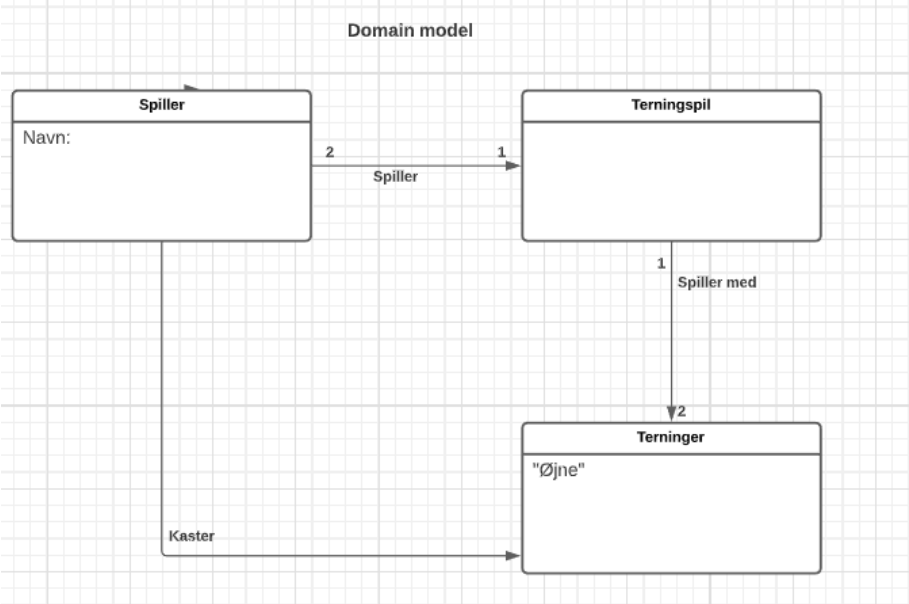
| UseCase: SpilSpillet |
|---|
| ID: 14 |
| Kort beskrivelse: To spillere starter med 1000 point, hvor de på skift kaster med to terninger, værdien af disse terninger, viser om spilleren skal få point, eller trække point fra, den spiller der først når 3000 point har vundet. |
| Primære aktører: Spiller1, spiller2 |
| Sekundære aktører: Ingen |
| Forudsætninger: Computeren skal have en compiler for at spillet kan køre. |
| Main flow:<br>1. Use casen begynder når spiller1 indtaster sit navn i den hvide boks og trykker på OK.<br>2. Spiller2 indtaster sit navn i den hvide boks og trykker på OK.<br>3. Spiller1 og spiller2's navne vises samt 1000 point nedenunder deres navne, der repræsentere spillerens pengebeholdning.<br>4. Spiller1 trykker på knappen OK, ved siden af knappen "Slå!"<br>5. Terningerne viser to random tal med en samlet værdi mellem 2-12<br>6. Den pågældende værdi repræsenterer et felt, som har enten en negativ eller positiv påvirkning på pengebeholdningen.<br>7. Alle felternes funktion er beskrevet i UseCase 3-13.<br>8. Spilleren vinder spillet ved at få 3000 point i sin pengebeholdning og så afsluttes spillet. |
| Postconditions: Der er en vinder af spillet. |

## Use Case Diagram



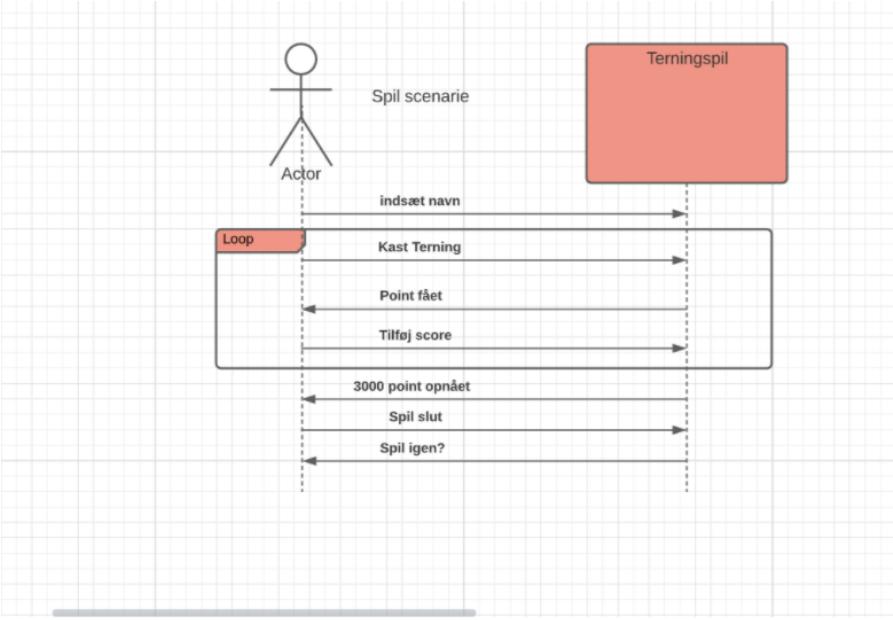
## Domænemodel

Denne model giver et visuelt overblik over handlingen i spillet, ved at vise hvor koden er blevet implementeret. I modellen fremstår objekterne og deres interaktion. Spillerne spiller spillet, der bliver spillet med terninger, og spillerne kaster med terningerne.




## Systemsekvensdiagram

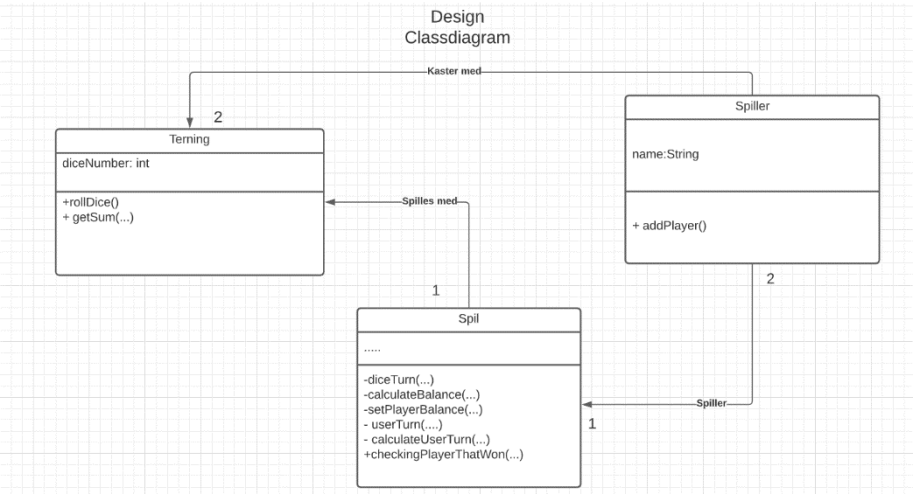
Systemsekvensdiagram er visualisering, der viser et bestemt scenarie af en usecase. I det nedenstående diagram ser man en visualisering af Use Case: Spil Spillet.



# Design

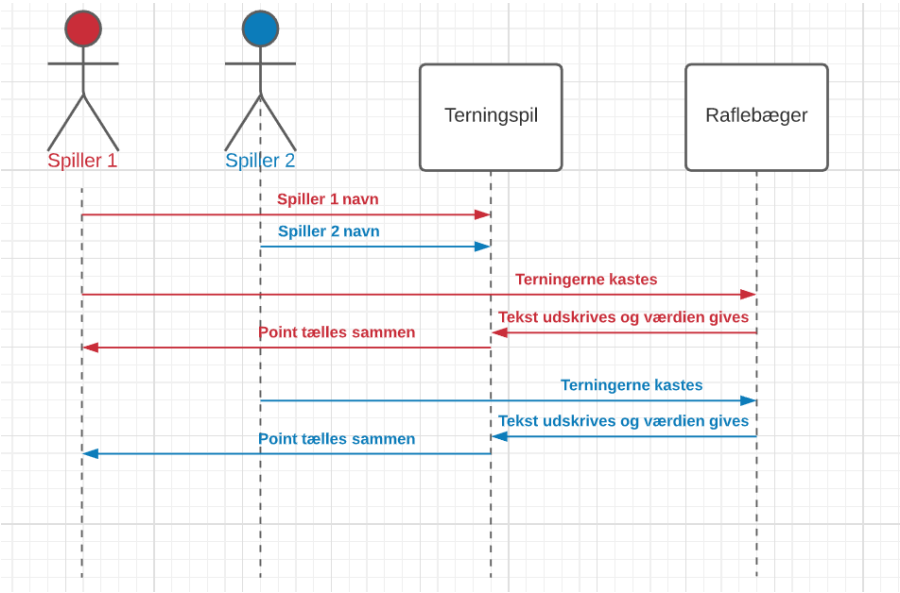
## Designklassediagram

Design klassediagram viser hvornår en software klasse spiller brugt, og hvordan klasserne interagerer med hinanden.




## Sekvensdiagram

Der er blevet lavet et sekvensdiagram. Den viser objekterne der indgår i systemet, og hvordan de interagerer med hinanden. Der bliver vist i rækkefølge hvordan meddelelserne spillet fungerer. I vores sekvensdiagram kan man se spilleren skal indsætte sit navn. Derefter kaster hver spiller med de 2 terninger, hvor der angives værdien, som efterfølgende bliver tilføjet til deres samlet point.



## Implementering

For at terningerne nemt kan skiftes ud, har vi vurderet at det bedste ville være at lave en die class, med de forskellige attributter og metoder en terning har. Vi valgte bl.a. at lave en metode rollDie(), som kun kaster med en terning og en metode diceTurn() som kaster to terninger. Vi har derudover oprettet en Gui klasse som kun holder styr på felternes navne & farver. Player klassen har til formål at holde styr på spillernes navne, derudover har den en metode som har til formål at tilføje spillerne til brættet. Vi vurderede at det ville være bedst at have spil logikken i en Game klasse og vores Main metode for sig selv i en Main klasse, hvor der er blevet oprettet en instance af Game klassen. Alt dette er blevet gjort med low coupling in mente.

## Test

Vi har udfør en række tests vha. JUnit, derudover har vi testet programmet af på DTU's databarer.

Inde i en GameTest klasse har vi udført følgende 3 tests:

1. **testCalculateNegativeBalance()**
   Tester at spillernes balance ikke kan blive negativ.
2. **testDiceMax()**

   Tester at summen af de to terninger ikke kan blive over 12.
3. **tesDiceMin()**

   Tester at summen af de to terninger ikke kan blive under 2.

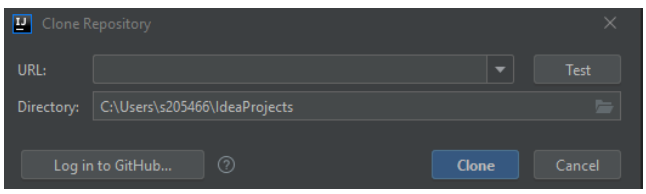
Prøverne fungerede og programmet bestod dem alle. Ydermere fungerede programmet på DTU's Databarer, efter det blev importeret ned fra GitHub.

## Konfiguration

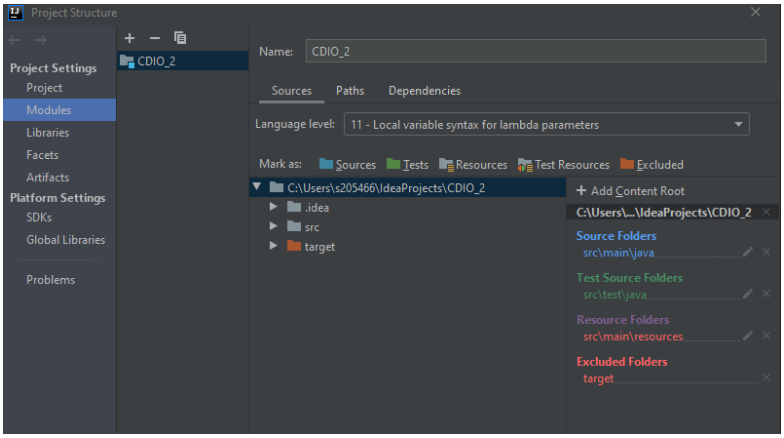
For at kunne køre spillet og dets tests, skal InteliJ (version 2019.2.1 og op) skal være downloadet. Koden hentes fra vores Git repository, linket til Git repository ligger for enden af afsnittet.

### Vejledning

Åben InteliJ og tryk på "File" → "New" → "Project from version control" → "Git", derefter insæt linket til Git repository der hvor der står URL.



Når dette er gjort, skal man sørge for at man kører de rigtige indstillinger, dette gør man ved at trykke på ”Settings” under ”File”, og folde ”Build, Execution, Deployment” → folde ”compiler” ud også trykke ”java compiler”. Under ”target bytecode version” skal der ikke stå noget, det er derfor vigtigt at tjekke om der står noget. Dernæst skal man åbne ”open module settings”, den finder man ved at højre klikke på projektetmappen. Man skal så sørge for at ”language level” er sat til at være 11. Nu kan man køre spillet.




Der er mulighed for at Maven ikke er blevet importeret korrekt når man først importerer projektet, hvis dette sker og man ikke kan køre programmet, så skal man højreklikke på filen ”pom.xml” → klikke ”maven” → ”Reimport”. Når alt er reimporteret, så er problemet løst. Der er dog mulighed for at dette trin ikke er nødvendigt.

### GitHub Repository

https://github.com/Ayafahim/CDIO_2.git

## Konklusion

Vi kan hermed konkludere at kundens krav blev opfyldt og at programmet kører godt. Der har derudover været et godt samarbejde i gruppen under projektet, hvilket gjorde det meget nemt at udarbejde et kørende program. Alle kravene for de tests der skulle køres, er blevet mødt med succes.

## Litteraturliste

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development* (3rd ed.). Pearson.