DTU

02312 62531 62532

Introduktion til programmering, Udviklingsmetoder til IT-systemer, Versionsstyring og testmetoder
Gruppenummer: 15

CDIO 3

Jens Will Iversen - s205411

images/Jens.jpg

images/Jonathan.jpg

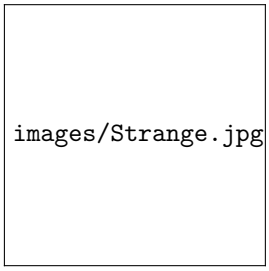
Jonathan Emil Zørn - s194134

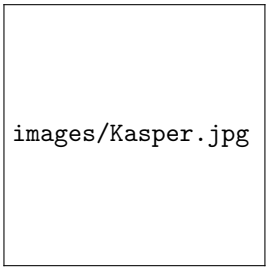
Niklas Jessen Børner - s205454

images/Thomas.jpg

Thomas Stender Bonde - s205440

Kasper Strange - s205467





Kasper Haugaard Hansen - s205434

27. november 2020

Timeregnskab

Dato	Jens	Jonathan	Niklas	Thomas	Kasper Haugaard	Kasper Strange
09/11/2020	2	4.5	1.5			
10/11/2020	3	3	2			
14/11/2020	1.5					
15/11/2020		1.5				
xx						
xx						
xx						
xx						
xx						
xx						
xx						
xx						

Tabel 1: Antal timer brugt på projektet

Abstract

GitHub - link

https://github.com/baldm/15_del3

Indhold

1 Indledning

2 Analyse

2.1 Krav

Functionality

1. Spillet skal have 2-4 spillere.
2. Spillet skal have 2 terninger.
3. Spillepladen skal have 24 plader.
4. Det skal være muligt at spille på forskellige sprog.
5. Spillet skal kunne tracke spillerens pengebeholdning.
6. Spillet skal kunne administrere spillerens pengebeholdning.
7. Spillet skal kunne administrere spillerens ejendomme, herunder husleje og ejerskab.
8. Spillet skal gøre brug af chancekort.

Usability

1. Spillerene skal kunne spille spillet på egen hånd efter en prøve omgang

Reliability

1. ?

Performance

1. Fra man kaster med terningerne til man ser resultatet må der højst gå 333 ms.

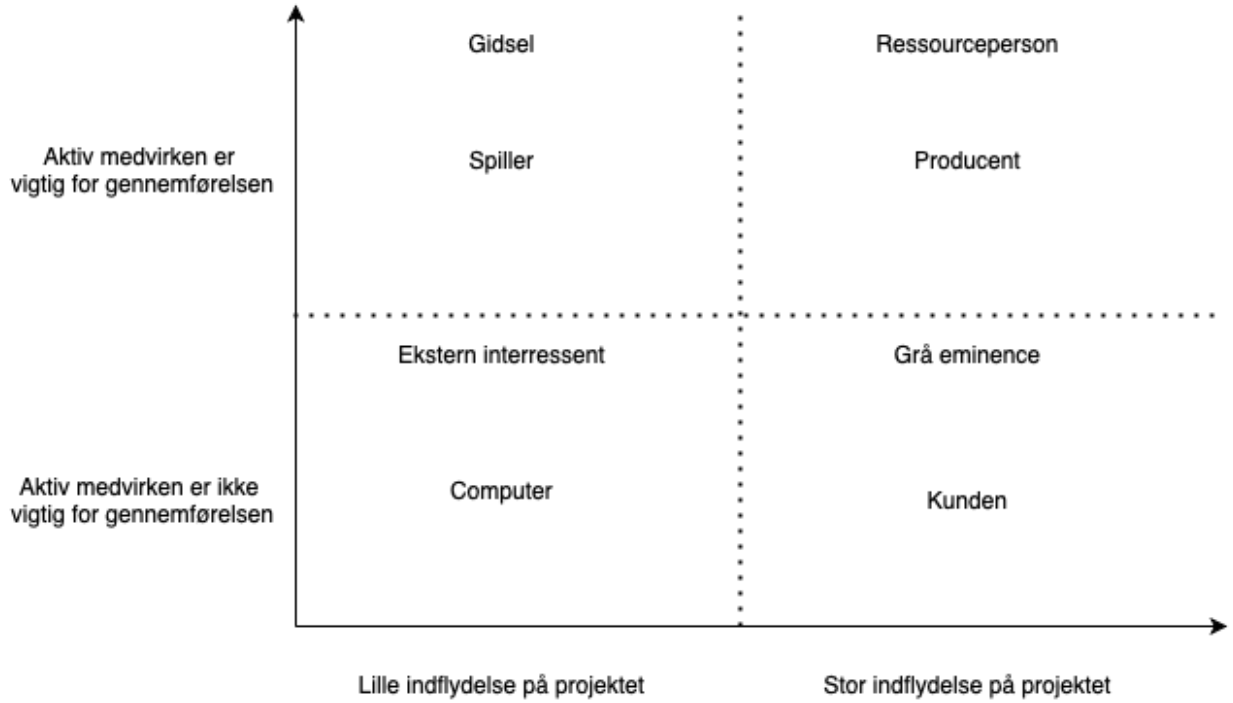
Supportability

1. Spillet skal kunne spilles på alle computere med styresystem Windows, MacOS eller Linux, hvis Java 15 eller nyerer er installeret.
2. Spillet skal kunne spilles på computerne i DTU's databarer.

2.2 Interessanter[1]

1. Kunden
2. Spiller
3. Producent
4. Computer (Til at spille på)

Interessantanalyse diagram[1]



Figur 1: Interessantanalyse diagram

2.3 Aktører

1. Spiller - primær
   Spillets målgruppe og hovedsageligt den person der har allermest med spillet at gøre.
2. Producent - primær
   Skal vedligeholde og teste spillet regelmæssigt, og vil derfor have meget indflydelse.

2.4 Use case

1. Spil spil
2. Test terninger
3. Test spilelplade
4. Test chancekort

Use case diagram

Beskrivelse

2.5 Domænemodel

2.6 Systemsekvensdiagram

3 Design

3.1 Designklassediagram

3.2 Sekvensdiagram

4 Implementering

4.1 Krav til computer[1]

1. Java JDK 15
2. Windows, Linux eller Mac styresystem

4.2 Importering af Git repository[1]

1. Oprettet en mappe, hvortil spilles skal ligges i
2. Kopier mappen som stinavn
3. Åben computerens terminal/command prompt
4. Skriv 'cd ' og indsæt derefter mappens stinavn og klik 'enter'
5. Kopier dette link: 'https://github.com/baldm/15_del3'
6. I terminalen/command prompten: Skriv 'git clone ' og indsæt derefter det kopieret link og tryk 'enter'
7. Koden er herefter downloadet ned på computeren og placeret i den valgte mappe

4.3 Compiling og afvikling af koden

Terminal

IntelliJ

Jar fil

5 Dokumentation

5.1 Arv

5.2 Abstract

5.3 LandOnField

5.4 Test

5.5 GRASP

5.6 Overvejelser omkring spillets regler

6 Test

6.1 Brugertest

6.2 Terning (JUnit)[1]

NB: denne test er lavet i CDIO2, dog bruges de samme terninger og testen er derfor stadig relevant i CDIO3.

Vi har testet, hvorvidt terningernes totale antal viste øjne, laver en binominal-fordeling omkring 7, og har fået følgende plot:

Vi har udregnet variansen for vores udregning ved hjælp af formlen:

$$\sigma^2 = \sum_{i=1}^{12} \sqrt{i - \mu} \cdot p_i$$

Hvoraf p_i er sandsynligheden for at et givent antal øjne er blevet slået og μ er middelværdien.
μ udregnes således:

$$\mu = \sum_{i=1}^{12} i \cdot p_i$$

og p_i udregnes ved hjælp af formlen:

$$p_i = \frac{x_i}{n}$$

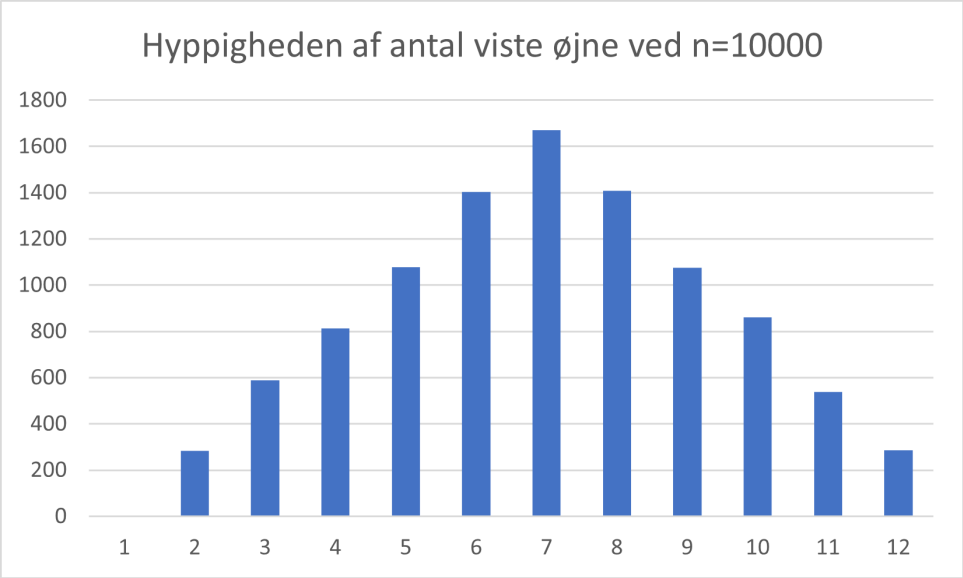
Hvoraf x_i er antal gange det givne antal øjne er slået, og n er antal gange der er i alt er blevet slået med terningerne. Når σ^2 er udregnet, udregnes $\sigma = \sqrt{\sigma^2}$ både for den teoretiske σ_t samt for vores terninger σ, hvorefter afvigelsen af hhv. σ og μ udregnes, således:

$$\sigma_{afvigelse} = \frac{|\sigma_t - \sigma|}{\sigma_t}$$

$$\mu_{afvigelse} = \frac{|\mu_t - \mu|}{\mu_t}$$

Teoretisk set vil $x_i = [0, 1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]$ for $i = [1, 2, ..., 12]$, $\mu = 6.\overline{9}$ og $\sigma \approx 2.4152$.

Her er et søjlediagram over hyppigheden af antal øjne slået, ved antal kast $n = 10000$:



Figur 2: Binominalfordeling over 10000 terningekast

Hvoraf henholdsvis σ og μ er udregnet til $\sigma \approx 2.4228$ og $\mu \approx 6.9948$, hvilket giver en afvigelse $\mu_{afvigelse} = 7.42 \cdot 10^{-3}$ samt en $\sigma_{afvigelse} \approx 0.0031$
Dermed kan vi konkludere, at vores virtuelle terning har en så lille afvigelse fra en teoretisk terning, at den er god at bruge.
Alle informationer omkring udregning af sigma er fundet på [2]

7 Versionsstyring

7.1 Navngivning[1]

Der skal være en skabelon for navngivning af branches inden man starter. Det kunne f.eks. være:

- Ekstraopgave-1
- Ekstraopgave-2
- bugfix-rules
- bugfix-launcher

Med denne regel vil man ved et hurtigt kig, kunne danne sig et overblik over de branches, der er aktive.

8 Konfigurationsstyring

9 Konklusion

Litteratur

[1] Gruppe 15. *CDIO 2*. URL: https://www.overleaf.com/read/frssjscpgchz. (accessed: 09.11.2020).

[2] Ismor Fischer. *Classical Probability Distributions*. URL: http://pages.stat.wisc.edu/~ifischer/Intro_Stat/Lecture_Notes/4_-_Classical_Probability_Distributions/4.1_-_Discrete_Models.pdf. (accessed: 02.10.2020).