

CDIO Del 1

Fag: 02312, 62531, 62532

Gruppe/hold nr. 8

Afleveringsfrist:

01-10-2021

Navn og studienr:

Ahmad Sandhu s215813

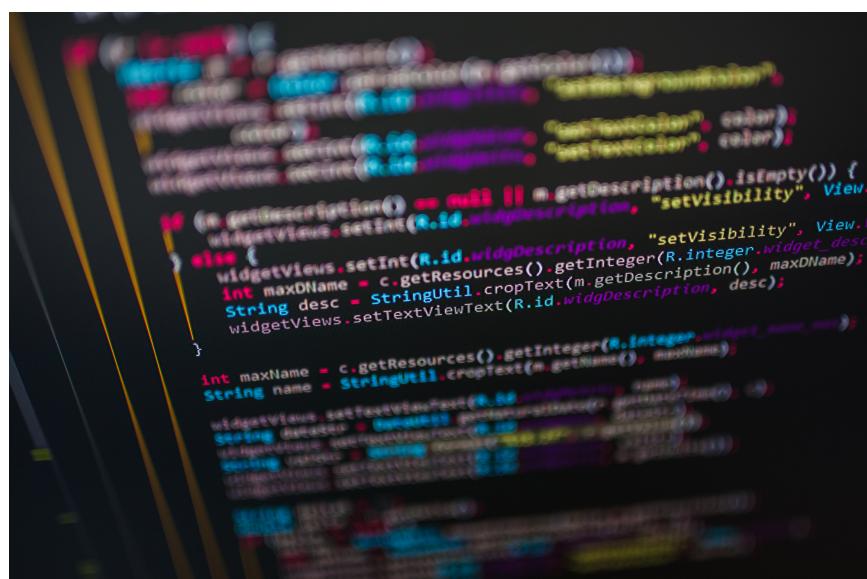
August Stax Nielsen s215785

Fares Al Jawabra s192080

Majid Al Shami s200787

Maiyar Abed s216251

Yazan Al-Hasan s193397



Indholdsfortegnelse:

Indholdsfortegnelse:	2
Indledning	3
Kravspecifikation	4
Dokumentation	5
Analyse	6
Design	7
Implementering	7
Test	7
Konklusion	7
Bilag	8
Noter	12

Indledning

Denne rapport indeholder en opgave om at udvikle et system, der bruges på maskinerne(Windows). Den pågældende opgave er et spil mellem 2 personer. Produktet skal kunne fungere mellem 2 spillere, hvor hver spiller skal kunne slå 2 terninger og være i stand til at se resultatet med det samme. Vi vil i denne opgave undersøge om det er muligt at implementere kundens krav og vision og dermed udvikle systemet/spillet. Vi udarbejder og koder opgaven i IntelliJ. Derudover overvejer vi alle de problemer der kan opstå under udviklingen af systemet og finder en løsning på forhånd for at sikre os det bedste resultat og tilfredse kunder. Rapporten indeholder en analyse, design, implementering og test af opgaven. En analyse for at forudse og reagere på forskellige risikoscenarier. Vi designer spillet på GUI for at give brugerne en visuel visning af spillet. implementeringsfasen dokumentere hvordan vi har udviklet spillet og de forskellige udfordringer undervejs. Sidst men ikke mindst, en test af spillets funktionalitet i henhold til kundens anmodning.

Hvad er formålet?

Formålet med opgaven er at udvikle et fungerende spil, som opfylder eller nogenlunde tæt på kundens krav for stadig at sikre en velfungerende og vellykket opgave.

Kravspecifikation

1. Spillet skal kunne spilles på et Windows styresystem.
2. Spillet skal kunne spilles på maskinerne i databarerne.
3. Spillet medtager præcis 2 personer.
4. Spillet spilles med 2 terninger og resultatet skal vises lige efter et slag.
5. Summen af terninger ligges til den pågældende spillers point.
6. Spilleren mister alle sine point hvis spilleren slår to 1'ere.
7. Spilleren får en ekstra tur hvis spilleren slår to ens.
8. Spilleren kan vinde spillet ved at slå to 6'ere, hvis spilleren også i forrige kast slog to 6'ere uanset om det er på ekstrakast eller i forrige tur.
9. Spilleren skal slå to ens for at vinde spillet, efter at man har opnået 40 point.
10. Er spilleren på 40 point og slår to 1'er mister personen alle sine 40 point men får 2 point og en ekstra tur.
11. Alle almindelige mennesker skal kunne spille spillet uden en brugsanvisning.
12. Spillet skal fungere korrekt med en spiltest hen over 1000 kast.
13. Spillet skal indeholde en dokumentation.
14. Fagudtryk skal fremstå naturlige i dokumentationen.
15. Spillet vil indeholde den GUI som kunden har fået lavet til et andet projekt.
16. Programmet skal være på engelsk sådan at alle på DTU kan forstå spillet.
17. Programmet skal udvikles med IntelliJ.
18. Det skal kunne ses af historien på github at alle gruppens medlemmer har committet et eller andet.
19. Aflevering af programmet skal inkludere et link til vores github repo.
20. Kunden vil gerne kunne tjekke den seneste fungerende kode ud fra master-branchen, derfor inkluderes til sidst ændringer som ikke er kørt og testet i en development-branch.

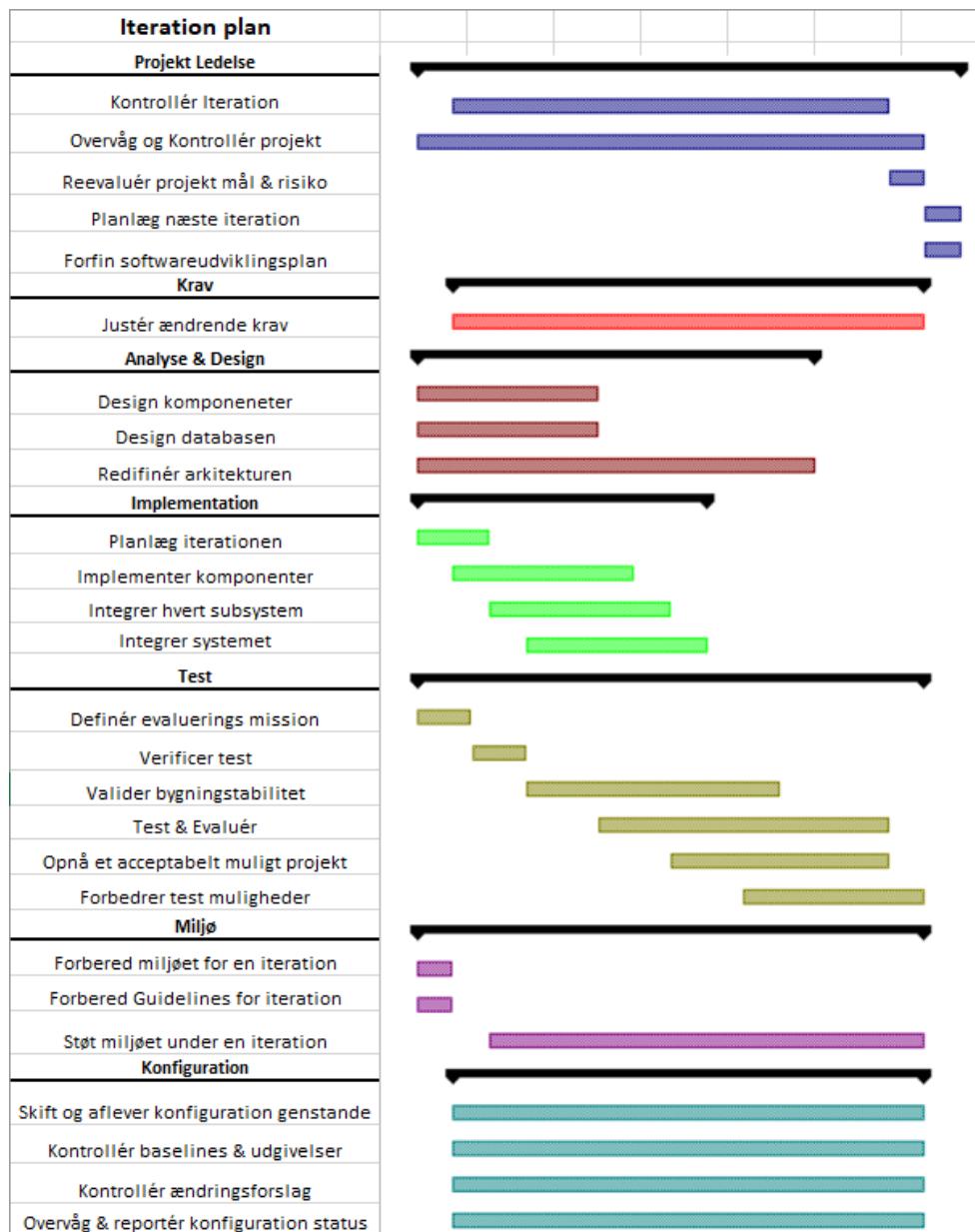
Dokumentation

Vi har valgt at beskrive vores udviklingsarbejde gennem to artifacts som vi synes er relevante i forhold til den konkrete opgave. De to artifacts ”Risk List & Risk Management plan” og ”Iteration Plan” viser en strategisk planlægning af arbejdsprocessen, samt hvordan vi kan forberede os på de problemer der kunne opstå under udviklingen og effektivt løse dem.

Artifact 1. Risk List & Risk Management plan

Sandsynlighed Impact	0-20%	20-40%	40-60%	60-80%	80-100%	
1	Strømsvigt	Person kommer til skade	Skrivefejl i kode	Person møder sent	Person bliver distraheret	
2	Kopiering af code					
3	Hardware fejl		Mangel på ekspertrie	Mangel på specifikation af krav	Mangel på finpudsning af software	
4	Person vil holde ferie		Fejl i kommunikation	Utilstrækkelig hardware	Utilstrækkelig tidsplan	
5	Person er syg		Tidsvindue lukker	Mangel på mandskab	Overbelastning af server	
INGEN REAKTION		5. Cut Off		Person er syg		
OVERVÅG		Risici vi er klar til at acceptere:		Person kommer til skade		
REAKTION				Person vil holde ferie		
HURTIG REAKTION		Skrivefejl i kode		Person møder sent	--> Vi forventer nogle af disse scenarer kommer til at ske men acceptere det på baggrund af at ingen mennesker er perfekte.	
STOP		Kopiering af code				
STOP		Mangel på finpudsning af software				
		Person bliver distraheret	--> Vi forventer at disse scenarer kommer til at ske og accepterer dem fuldt ud.		8. Svar kort på: Hvornår er det relevant at	
6. Hvilke risici bør/kan forebygges					a. Overføre en risiko	--> Det er relevant at overføre en risiko når man hyrer en tredje part at overtage risikoen.
Risici vi vil prøve at forebygge:		Fejl i kommunikation			b. Acceptere en risiko	--> Det er relevant at acceptere en risiko når der er en lille chance gængt med en lille impact.
Utilstrækkelig hardware		Hardware fejl	Vi forventer ikke at disse scenarer kommer til at ske men forventer også at kunne løse udfordringen		c. Afbøde en risiko	--> Det er relevant at afbøde en risiko når du vil minimere eller svække risikoen så den får mindre impact.
Mangel på mandskab					d. Eliminere en risiko	--> Det er relevant at eliminere en risiko når impacten er høj og derfor kan ruiniere projektet.
Utilstrækkelig tidsplan	7. Hvad gør i hvis de optræder		og accepterer dem på baggrund af dette.			
Mangel på specifikation af krav						
Mangel på ekspertrie	--> Disse scenarer optræder ved dårlig planlægning som vi derfor vil have forbygget forinden for at kunne komme i mål med projektet.					

Artifact 2. Iteration plan



Analyse

Vores arbejdsproces er blevet effektiviseret og gjort mere stabilt ved hjælp af forskellige artefakter. I dokumentationen ser vi Risk List & Risk Management Plan som har hjulpet med risikoidentifikation, samt at vurdere hvordan man skal reagere på forskellige risikoscenarier for at optimere arbejdsprocessen. Man kan foroven også se vores Iteration plan, som går mere i detaljer med kodningen og viser den mængde tid og ressourcer som vi forventer at bruge på hver iteration som vi har opstillet.

Design

Vi har benyttet GUI (Graphical User Interface) som giver os en design interface af et matador spil. Vi benytter GUI interfacet for at printe terningerne på matador boardet, så spillet ser lidt livligt ud og ikke kun skal ses på et terminalvindue. Man skal dog stadig se og taste på terminal vinduet for at spille spillet og se hvis tur det er og hvem der vinder.

Implementering

Vores program er kodet i Java i IntelliJ programmet. Da vi har benyttet Java version 16, kan DTU's computere ikke køre programmet, da de kører en ældre version. Løsning til dette problem er at vi er nødt til at compile vores jar.fil til ældre version.

Vores program består af en Class, if/else statement og while-løkke. Vi har benyttet en Class til at programmere vores terninger (se bilag 3.). Så har vi benyttet if/else statement til at programmere reglerne for spillet (se bilag 4.), og til sidst har vi benyttet en while-løkke til at lade spillet køre indtil den finder en vinder.

Test

Vi fik besked fra kunden at vi skulle teste om terningerne stadig virkede når der blev kastet 1000 gange og at vi skulle tælle hvor mange kast, hvor terningerne er ens.

Måden vi har gjort det er at benytte en while-løkke og en If- statement. While-løkkens benytter vi til at kaste terningerne 1000 gange og if-statement benytter vi inde i while løkken til at tjekke om terningerne er ens. (Se bilag 5. og bilag 6.)

Konklusion

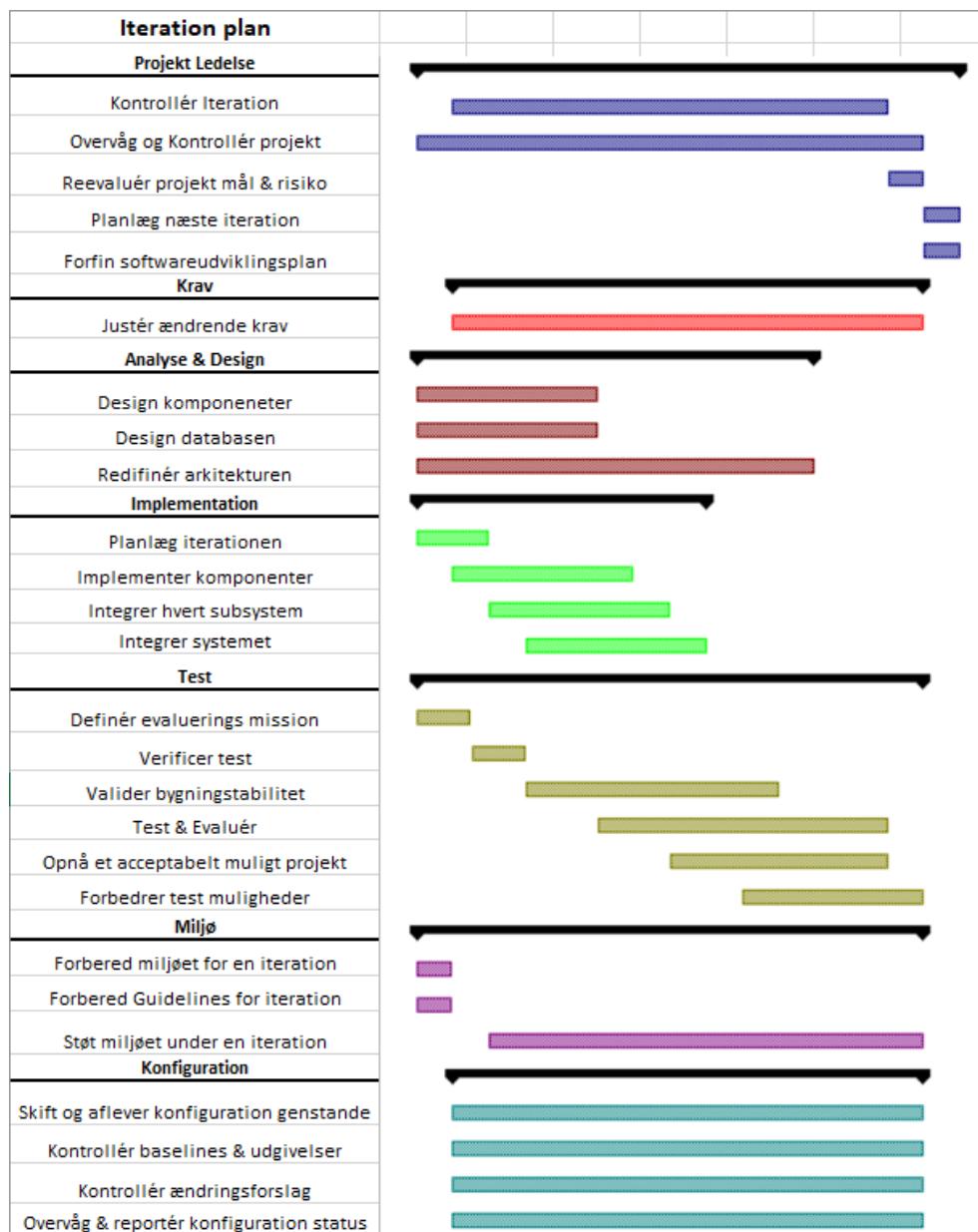
Vi kan konkludere at vores program virker, og at vores test viser og fortæller at vores terninger virker stadig helt perfekt efter 1000 kast. Dog kunne der godt være nogle forbedringsforslag fx. at man kunne forkorte programmet, så det fylder mindre end det gør nu.

Bilag

Bilag 1. Dokumentation – Risk List & Risk Management plan.

Sandsynlighed Impact	0-20%	20-40%	40-60%	60-80%	80-100%		
1	Strømsvigt	Person kommer til skade	Skrivefejl i kode	Person møder sent	Person bliver distraheret		
2	Kopiering af code						
3	Hardware fejl		Mangel på ekspertise	Mangel på specifikation af krav	Mangel på finpudsning af software		
4	Person vil holde ferie		Fejl i kommunikation	Utilstrækkelig hardware	Utilstrækkelig tidsplan		
5	Person er syg		Tidsvindue lukker	Mangel på mandskab	Overbelastning af server		
INGEN REAKTION OVERVÅG REAKTION	5. Cut Off	Risici vi er klar til at acceptere:	Person er syg Person kommer til skade Person vil holde ferie Person møder sent	-->	Vi forventer nogle af disse scenarier kommer til at ske men acceptere det på baggrund af at ingen mennesker er perfekte.		
HURTIG REAKTION STOP STOP	Skrivefejl i kode Kopiering af code Mangel på finpudsning af software	Person bliver distraheret	-Vi forventer at disse scenarier kommer til at ske og accepterer dem fuldt ud.				
6. Hvilke risici bør/kan forebygges	Risici vi vil prøve at forebygge:	Fejl i kommunikation Hardware fejl	Vi forventer ikke at disse scenarier kommer til at ske men forventer også at kunne løse udfordringen og accepterer dem på baggrund af dette.				
Utilstrækkelig hardware Mangel på mandskab Utilstrækkelig tidsplan	7. Hvad gør i hvis de optræder	Mangel på specifikation af krav Mangel på ekspertise -->	Disse scenarier optræder ved dårlig planlægning som vi derfor vil have forbygget forinden for at kunne komme i mål med projektet.	a. Overføre en risiko -->	Det er relevant at overføre en risiko når man hyrer en tredje part at overtage risikoen.		
				b. Acceptere en risiko -->	Det er relevant at acceptere en risiko når der er en lille chance givet med en lille impact.		
				c. Afbøde en risiko -->	Det er relevant at afbøde en risiko når du vil minimere eller svække risikoen så den får mindre impact.		
				d. Eliminere en risiko -->	Det er relevant at eliminere en risiko når impacten er høj og derfor kan ruinere projektet.		

Bilag 2. Dokumentation – Iteration Plan.



Bilag 3. Kode - Dices Class.

```
public class Dices {  
    private final int MAX = 6;  
    private int faceValue;  
  
    public Dices() { faceValue = 1; }  
  
    public int roll(){  
        faceValue = (int) (Math.random() * MAX)+1;  
        return faceValue;  
    }  
  
    public void setFaceValue(int Value) { faceValue = Value; }  
  
    public int getFaceValue() { return faceValue; }  
  
    public String toString(){  
        String result = Integer.toString(faceValue);  
        return result;  
    }  
}
```

Bilag 4. Kode - reglerne (if-statements)

```
if (die1Facevalue == 6 && die2Facevalue == 6){  
    System.out.println(player1 + " WON!");  
    break;  
}  
  
else if (die1Facevalue == 1 && die2Facevalue == 1){  
    totalsum1 = 0;  
}  
  
  
  
else if (die1Facevalue == die2Facevalue) { // player 1 extra turn  
    if (totalsum1 >= 40){ // Player1 after 40 points  
        System.out.println(player1 + " WON!");  
        break;  
    }  
    else{
```

Bilag 5. Kode - Test koden.

```
import java.util.Scanner;
public class test {
    public static void main(String[] args) {
        Dices die1, die2;
        die1 = new Dices();
        die2 = new Dices();
        int x = 0;
        int t = 0;
        while (x <= 1000) {
            x = x+1;
            int die1Facevalue = die1.roll();
            int die2Facevalue = die2.roll();
            System.out.println("Die One: " + die1 + ", Die Two: " + die2);

            if ( die1Facevalue == die2Facevalue){
                t = t+1;
            }
        }
        System.out.println(" der er så mange kast med samme værdi: " +t);
    }
}
```

Bilag 6.. Kode - Test resultat.

```
Die One: 1, Die Two: 5
Die One: 4, Die Two: 3
Die One: 4, Die Two: 2
Die One: 5, Die Two: 3
Die One: 1, Die Two: 2
Die One: 2, Die Two: 5
Die One: 1, Die Two: 1
Die One: 3, Die Two: 4
Die One: 5, Die Two: 3
Die One: 6, Die Two: 1
Die One: 3, Die Two: 3
Die One: 6, Die Two: 1
Die One: 3, Die Two: 3
Die One: 5, Die Two: 2
Die One: 6, Die Two: 5
Die One: 3, Die Two: 6
Die One: 6, Die Two: 2
Die One: 2, Die Two: 1
Die One: 5, Die Two: 6
der er så mange kast med samme værdi: 175

Process finished with exit code 0
```

Gruppe 08

DTU

Kursusnr: 02312, 62531,

62532