History of Hexgame

Hex, 1942'de Danimarkalı matematikçi Piet Hein tarafından icat edilen bir strateji oyunudur. Hein'in amacı, basit ancak derin stratejik düşünme gerektiren bir oyun yaratmaktı. Oyun, altıgen hücrelerden oluşan bir tahta üzerinde oynanır ve iki oyuncunun kendi renkleriyle karşılıklı iki kenarı birleştirmesi amaçlanır.

Amerikalı matematikçi John Nash daha sonra oyunu keşfetti ve kendi adıyla anılmaya başlandı. Nash'in versiyonu "John" olarak adlandırıldı, ancak genellikle "Hex" olarak bilinir hale geldi.

Hex, 1952'de Parker Brothers tarafından ticari olarak piyasaya sürüldü. Oyunun kuralları basit olmasına rağmen, matematiksel derinliği ve stratejik karmaşıklığı sayesinde hem matematikçiler hem de oyun meraklıları tarafından ilgi gördü.

Bugün, Hex klasik masa oyunu formatında ve çeşitli dijital platformlarda popüler bir oyun olarak varlığını sürdürmektedir. Matematiksel ve stratejik oyunlar arasında önemli bir yere sahiptir ve özellikle oyun teorisi alanında çalışan bilim insanları arasında popülerdir.

Proje Raporu: JavaFX FXML Tabanlı Oyun Uygulaması

1. Giriş

Bu rapor, JavaFX FXML kullanarak geliştirilen bir oyun uygulamasını detaylı bir şekilde incelemektedir. Raporun ilerleyen bölümlerinde proje yapısını, kullanılan teknolojileri, sınıf diyagramlarını ve kodların detaylarını bulacaksınız.

2. Proje Tanımı ve Amacı

Bu proje, iki oyuncunun sırayla hücreleri kendi renkleriyle doldurarak kendi renkleriyle kenarları birleştirmeye çalıştığı bir Hex oyununun JavaFX ile geliştirilmiş versiyonudur. Oyunun amacı, bir oyuncunun kendi renk hücreleri arasında kesintisiz bir yol oluşturarak karşılıklı kenarları birleştirmesidir.

3. Kullanılan Teknolojiler

Projenin geliştirilmesinde aşağıdaki teknolojiler kullanılmıştır:

Kullanılan Teknolojiler: Java, JavaFX

Proje Yapısı: MVC (Model-View-Controller) deseni kullanılmış.

Önemli Algoritmalar: Rekürsif yol bulma algoritmaları (FindWay metodları).

JavaFX: Grafik arayüz bileşenlerini oluşturmak için kullanılmıştır.

FXML: JavaFX uygulamaları için arayüz tasarımını XML tabanlı olarak tanımlamak için kullanılmıştır.

Java: Projenin temel programlama dili olarak kullanılmıştır.

4. Sınıf Diyagramları

Yukarıdaki sınıf diyagramı, projenin temel sınıflarını ve aralarındaki ilişkileri göstermektedir.

1. MainScreenApplication: JavaFX uygulamasının ana sınıfıdır. Uygulamayı başlatır ve ana ekranı yükler.

2. MainScreenController: Ana ekranın kontrol sınıfıdır. FXML dosyası ile bağlantı kurar ve kullanıcı etkileşimlerini yönetir.

3. GameModel: Oyunun model sınıfıdır. Oyun mantığını ve temel işlevleri içerir.

4. MapBuilder: Oyun haritasını oluşturan sınıftır. GridPane üzerine altıgenler yerleştirir.

5. SelectAndColorize: Kullanıcının altıgenlere tıklamasını ve renk değişimini yöneten sınıftır.

6. EndControlRed: Kırmızı renkli altıgenlerin oyunun sonunu kontrol eden sınıftır.

7. EndControlBlue: Mavi renkli altıgenlerin oyunun sonunu kontrol eden sınıftır.

5. Sonuç ve Değerlendirme

Bu rapor, JavaFX FXML tabanlı bir oyun uygulamasının geliştirilmesi sürecini ve proje yapısını detaylı bir şekilde ele aldı. Projede kullanılan teknolojiler, sınıf diyagramları ve kod detayları incelenerek, proje hakkında kapsamlı bir bilgi sunuldu.

Yapılan Testler

1. Birim Testler

Birim testler, her bir sınıfın ve metodun doğru çalıştığını kontrol etmek için yapılır. İşte bazı önemli birim testler:

EndControlBlue ve EndControlRed

endControl Metodu:

Harita boşken doğru sonuç döndürüyor mu?

Mavi/kırmızı oyuncu için kazanma durumu var mı?

Haritada geçerli bir yol bulabiliyor mu?

FindWay Metodu:

Yol bulma algoritması doğru çalışıyor mu?

Kenar durumlarını doğru kontrol ediyor mu?

Yanlış renk hücrelerde doğru sonuç döndürüyor mu?

GameModel

winnerWriter Metodu:

Kazananı doğru bir şekilde bildiriyor mu?

Label'ı doğru şekilde güncelliyor mu?

MainScreenController

Event Handlers:

Kullanıcı eylemlerini doğru şekilde işliyor mu?

Oyun akışını doğru şekilde yönetiyor mu?

MapBuilder

buildMap Metodu:

Haritayı doğru şekilde oluşturuyor mu?

Tüm hücreler doğru şekilde yerleştirilmiş mi?

SelectAndColorize

selectAndColorize Metodu:

Seçilen hücreyi doğru şekilde renklendiriyor mu?

Geçerli/Geçersiz seçimleri doğru işliyor mu?

2. Entegrasyon Testleri

Entegrasyon testleri, birimlerin birlikte doğru çalıştığını kontrol eder.

Oyun Başlangıcı:

Oyun başlangıçta tüm bileşenler doğru şekilde çalışıyor mu?

Kullanıcı arayüzü ve arka plan mantığı doğru şekilde entegre olmuş mu?

Oyun Akışı:

Oyuncu hareketleri doğru şekilde işleniyor mu?

Her iki oyuncu için de kazanma durumu doğru şekilde kontrol ediliyor mu?

Oyun sonunda kazanan doğru şekilde bildiriliyor mu?

3. Sistem Testleri

Sistem testleri, tüm sistemin birlikte doğru çalıştığını kontrol eder.

Kullanıcı Arayüzü:

Tüm kullanıcı arayüzü bileşenleri doğru şekilde çalışıyor mu?

Butonlar, etiketler ve diğer kontroller doğru şekilde işliyor mu?

Performans:

Oyun büyük haritalarda performansını koruyor mu?

Ani kullanıcı eylemlerinde sistem yanıt süresi makul düzeyde mi?

4. Kullanıcı Kabul Testleri

Kullanıcı kabul testleri, son kullanıcıların uygulamayı beklenen şekilde kullanabildiğini kontrol eder.

Kullanılabilirlik:

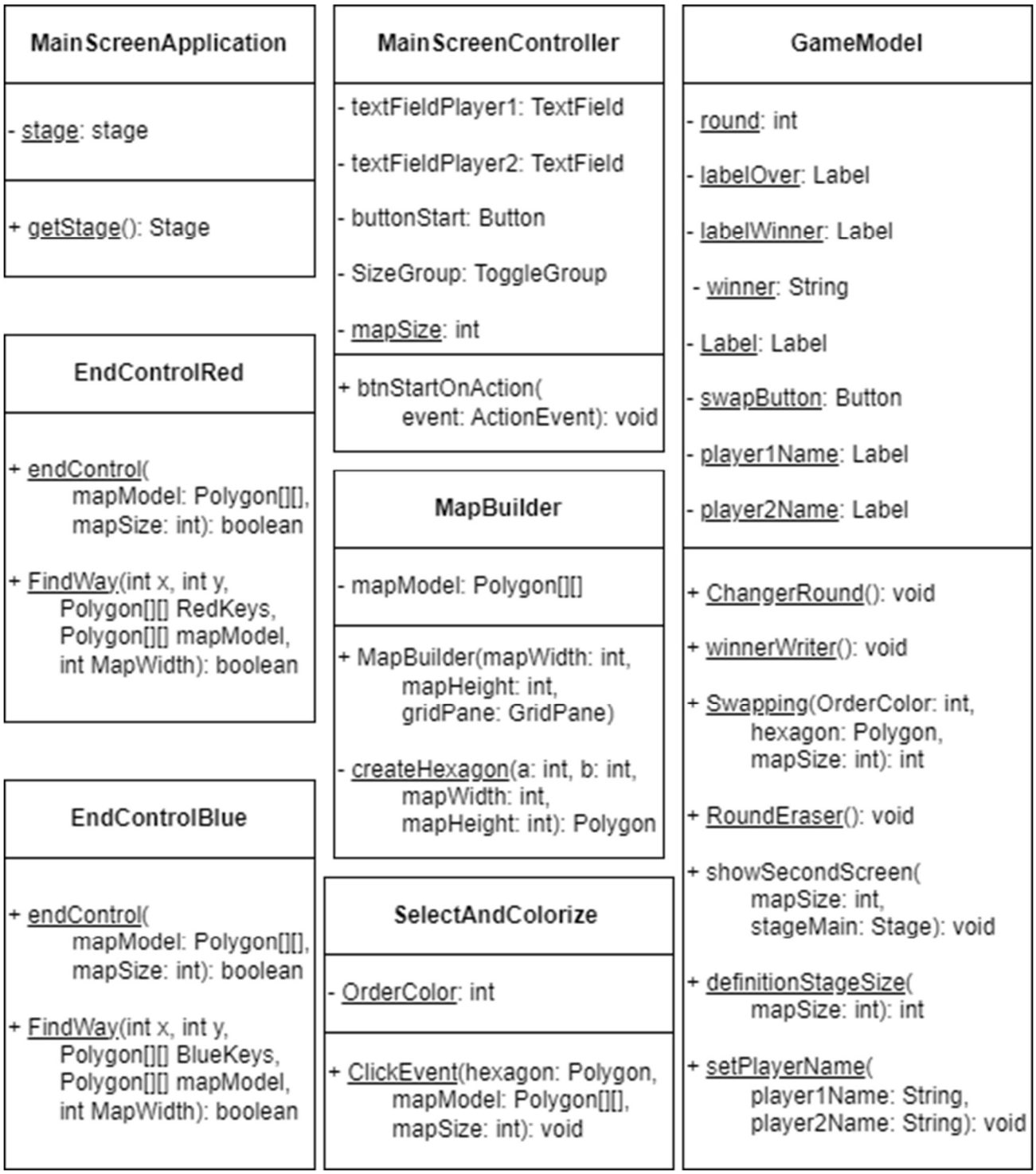
Kullanıcı arayüzü kullanıcı dostu mu?

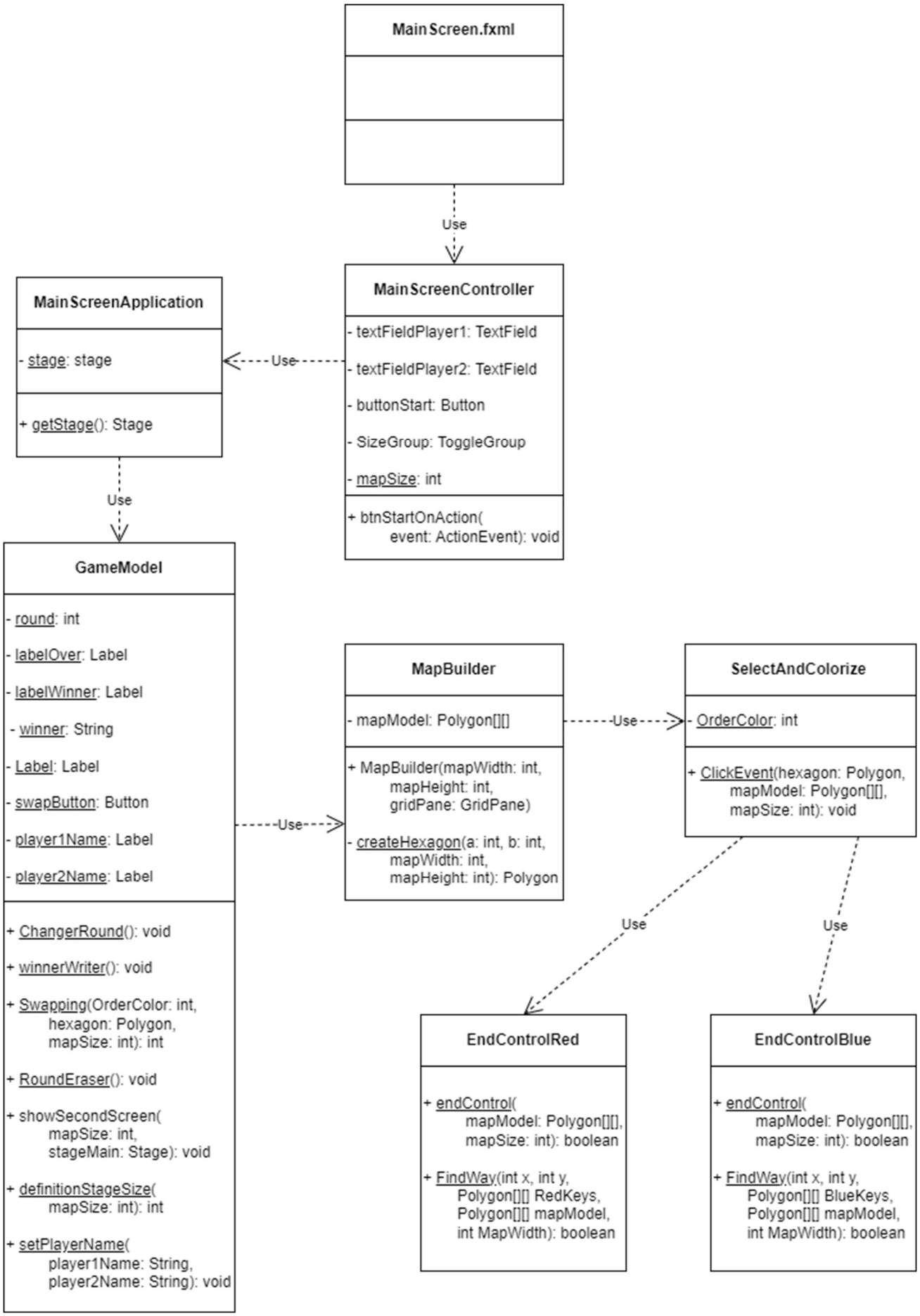
Kullanıcılar oyunu kolayca öğrenip oynayabiliyor mu?

Fonksiyonellik:

Oyunun tüm fonksiyonları son kullanıcı tarafından doğru şekilde kullanılabiliyor mu?

Kullanıcılar kazananın doğru şekilde belirlendiğini teyit edebiliyor mu?


MainScreenApplication
- stage: stage
+ getStage(): Stage
MainScreenController
- textFieldPlayer1: TextField
- textFieldPlayer2: TextField
- buttonStart: Button
- SizeGroup: ToggleGroup
- mapSize: int
+ btnStartOnAction(
event: ActionEvent): void
GameModel
- round: int
- labelOver: Label
- labelWinner: Label
- winner: String
- Label: Label
- swapButton: Button
- player1Name: Label
- player2Name: Label
+ ChangerRound(): void
+ winnerWriter(): void
+ Swapping(OrderColor: int,
hexagon: Polygon,
mapSize: int): int
+ RoundEraser(): void
+ showSecondScreen(
mapSize: int,
stageMain: Stage): void
+ definitionStageSize(
mapSize: int): int
+ setPlayerName(
player1Name: String,
player2Name: String): void
EndControlRed
+ endControl(
mapModel: Polygon[][],
mapSize: int): boolean
+ FindWay(int x, int y,
Polygon[][] RedKeys,
Polygon[][] mapModel,
int MapWidth): boolean
MapBuilder
- mapModel: Polygon[][]
+ MapBuilder(mapWidth: int,
mapHeight: int,
gridPane: GridPane)
- createHexagon(a: int, b: int,
mapWidth: int,
mapHeight: int): Polygon
EndControlBlue
+ endControl(
mapModel: Polygon[][],
mapSize: int): boolean
+ FindWay(int x, int y,
Polygon[][] BlueKeys,
Polygon[][] mapModel,
int MapWidth): boolean
SelectAndColorize
- OrderColor: int
+ ClickEvent(hexagon: Polygon,
mapModel: Polygon[][],
mapSize: int): void


MainScreen.fxml
Use
MainScreenController
- textFieldPlayer1: TextField
- textFieldPlayer2: TextField
- buttonStart: Button
- SizeGroup: ToggleGroup
- mapSize: int
+ btnStartOnAction(
event: ActionEvent): void
MainScreenApplication
- stage: stage
+ getStage(): Stage
Use
Use
GameModel
- round: int
- labelOver: Label
- labelWinner: Label
- winner: String
- Label: Label
- swapButton: Button
- player1Name: Label
- player2Name: Label
+ ChangerRound(): void
+ winnerWriter(): void
+ Swapping(OrderColor: int,
hexagon: Polygon,
mapSize: int): int
+ RoundEraser(): void
+ showSecondScreen(
mapSize: int,
stageMain: Stage): void
+ definitionStageSize(
mapSize: int): int
+ setPlayerName(
player1Name: String,
player2Name: String): void
Use
MapBuilder
- mapModel: Polygon[][]
+ MapBuilder(mapWidth: int,
mapHeight: int,
gridPane: GridPane)
- createHexagon(a: int, b: int,
mapWidth: int,
mapHeight: int): Polygon
Use
SelectAndColorize
- OrderColor: int
+ ClickEvent(hexagon: Polygon,
mapModel: Polygon[][],
mapSize: int): void
Use
Use
EndControlRed
+ endControl(
mapModel: Polygon[][],
mapSize: int): boolean
+ FindWay(int x, int y,
Polygon[][] RedKeys,
Polygon[][] mapModel,
int MapWidth): boolean
EndControlBlue
+ endControl(
mapModel: Polygon[][],
mapSize: int): boolean
+ FindWay(int x, int y,
Polygon[][] BlueKeys,
Polygon[][] mapModel,
int MapWidth): boolean

Üstlenilen Sorumluluklar

Muhammed Berke Said Darığa - 1030520980

1. **Harita Oluşturulması:**
    - Oyunun dinamik ve etkileyici bir ortamda oynanabilmesi için detaylı haritaların tasarlanması ve geliştirilmesi.
    - Harita elementlerinin, oyuncuların deneyimlerini artıracak şekilde yerleştirilmesi.
2. **FXML ile Arayüzler Oluşturulması:**
    - Kullanıcı dostu ve estetik açıdan çekici arayüzlerin FXML kullanarak oluşturulması.
    - Arayüzlerin oyunun genel temasına uygun şekilde dizayn edilmesi.
3. **Oyun Bitiş Algoritması Çözümü ve Planlaması (%30):**
    - Oyunun mantıklı ve tatmin edici bir şekilde sonlanmasını sağlayacak algoritmaların geliştirilmesi.
    - Bitiş senaryolarının planlanarak oyun akışının kesintisiz sağlanması.
4. **Oyunun Test Edilip Hataların Düzeltilmesi:**
    - Oyunun farklı senaryolarda test edilerek olası hataların tespit edilmesi.
    - Tespit edilen hataların düzeltilerek oyunun sorunsuz çalışmasının sağlanması.
5. **Animasyon Oluşturulması:**
    - Oyundaki nesneler için akıcı ve gerçekçi animasyonların tasarlanması.

Ali Çağrı Kaya - 1030521190

1. **Oyun Bitiş Algoritması Çözümü ve Planlaması (%70):**
    - Oyun sonunun nasıl gerçekleşeceğine dair algoritmaların detaylı bir şekilde planlanması ve uygulanması.
    - Oyun sonunun oyunculara tatmin edici ve anlamlı gelmesini sağlamak için farklı senaryoların geliştirilmesi.
2. **2 Kişilik Oyun Tasarımı:**
    - İki oyuncunun aynı anda oyunu oynayabileceği interaktif ve dengeli bir oyun tasarımı oluşturulması.
    - Çoklu oyuncu deneyimini zenginleştirecek özelliklerin eklenmesi ve optimize edilmesi.
3. **Oyun Yan İşlevleri (Swapping vs.):**
    - Oyunun ana mekaniklerinin yanı sıra oyunculara ekstra eğlence sağlayacak yan işlevlerin geliştirilmesi.
    - Yan işlevlerin oyun içi dengeleri bozmadan akıcı bir şekilde çalışmasının sağlanması.
4. **Oyunun Test Edilip Hataların Düzeltilmesi:**
    - Oyunun kapsamlı bir şekilde test edilerek kullanıcı deneyimini olumsuz etkileyebilecek tüm hataların tespit edilmesi.
    - Tespit edilen hataların hızlı ve etkili bir şekilde giderilmesi.

Ali Doğan - 1030520984