ALTIN TOPLAMA OYUNU

Kocaeli Üniversitesi Bilgisayar Mühendisliği Yazılım Labaratuvarı 1 – 1. Proje

Emre Yelbey
Kocaeli Üniversitesi
Bilgisayar Mühendisliği
180202043
emre_yelbey@hotmail.com

Ege Özeren
Kocaeli Üniversitesi
Bilgisayar Mühendisliği
180202047
ege_ozeren@gmail.com

Hazırlanan program kendine has davranış stillerine sahip 4 oyuncunun 2 boyutlu bir matris üzerinde bulunan altınları toplaması mantığı üzerine kurulmuş bir oyundur.

I. GİRİŞ

Program çalıştırıldığı anda kullanıcıyı oyun ayarlarının konfigüre edildiği menü karşılar. Kullanıcı bu menü aracılığıyla oyun içerisinde kullanılacak değerleri istediği şekilde değiştirebilir. Eğer herhangi müdahale yapılmak istenmiyorsa doğrudan "Başla" butonuna tıklanarak varsayılan değerlerle oyuna devam edilebilir.

Bu menüde bulunan ayar seçenekleri şunlardır:

- Oyun Alanı X: Oluşturulacak oyun alanı matrisinin x eksenindeki boyutunu alır.
- Oyun Alanı Y: Oluşturulacak oyun alanı matrisinin y eksenindeki boyutunu alır.
- Altın Oranı: Oyun alanı matrisinde % kaç oranında altın bulunacağı bilgisini alır.
- Gizli Altın Oranı: Oluşturulan altınların % kaçının gizli altın olacağı bilgisini alır.
- Başlangıç Altın Miktarı: Oyuna başlarken her oyuncuya dağıtılacak altın sayısı bilgisini alır.
- Adım Sayısı: Oyuncuların her hamlede kaç birim ilerleyeceği bilgisini alır.
- Hamle Maliyeti: Her oyuncu için ayrı ayrı hamle başına ne kadar altın harcayacağı bilgisini alır.
- Hedef Belirleme Maliyeti: Her oyuncu için ayrı ayrı hedef belirleme başına ne kadar altın harcayacağı bilgisini alır.
- Açılacak Gizli Altın Sayısı: C oyuncusunun her hedef belirleme işlemi öncesi kaç gizli altını açığa çıkaracağı bilgisini alır.

II. TEMEL BİLGİLER

Program **C#** programlama dilinde geliştirilmiş olup geliştirme ortamı olarak **Visual Studio 2019 16.7.6** kullanılmıştır.

III. TASARIM

Altın toplama oyunu programının geliştirme aşamaları belirtilen başlıklar altında açıklanmıştır.

A. Algoritma

Program temel olarak iki parçadan oluşur. İlk kısım kullanıcının oyun ayarlarını değiştirebildiği ayar menüsünü barındıran kısım, ikincisi ise oyunun bu ayarlara göre kullancıdan bağımsız olarak programı yürüttüğü kısımdır.

İlk kısımda tahmin edileceği üzere sadece winform komponentleri ile oyuna başlamayı veya oyundan çıkmayı tetikleyen *"eventhandler"*lar bulunur. Aynı zamanda bu aşamada kullanıcının girdiği değerlerin format kontrolü de yapılır.

B. Yöntem

Programın geliştirilme aşamasında ilk olarak kullanıcıyı karşılayacak ayarlar panelinin tasarlanması hedeflenmiştir. Bu menüdeki komponentler tasarlanıp programlandıktan sonra oyunun yürütüleceği adımın programlanmasına geçilmiştir.

Bu aşamadan sonra ise programın geri kalanı tamamen kullancıdan bağımsız olarak, her oyuncunun kendine has yöntemler ve algoritmalar kullanarak oynayacak şekilde programlanmıştır. Bu aşamada program şu şekilde ilerler:

Öncelikle oyunun oynanabilmesi, hatta oynanabilirlikten önce oynanan oyunun takip edilip izlenebilmesi için ilk olarak görsel programlama aşaması yapılmıştır. Bu aşamada ilk olarak üzerinde oyuncuların ve altınların bulunacağı oyun alanı matrisi oluşturulup çizdirilmiştir. Bu matris tüm oyunu üzerinde bulundurur ve çizimler bu matris referans alınarak yapılır.

Bu matrisi oluştururken yapılan ilk deneme matristeki her elemanı birer winform komponentinden oluşturup (Label veya pictureBox) bu komponentlerin hazır metodlarından faydalanmaktı. Bu düşünce başta kolaylık açısından mantıklı gibi gelse de performans açısından programın oldukça yavaş çalışmasına sebep oluyordu. 20x20 boyutlu bir matris oluşturulurken bile satırlar ardarda ve yavaş şekilde görüntüleniyordu. Biraz düşünüp araştırma yaptıktan sonra bu mantığı terk edip Rectangle sınıfını kullanmaya karar kıldık. Adından anlaşılacağı üzere bu sınıf 4 parametre alıp (bunlar: Xpozisyonu, Ypozisyonu, genişlik, yukseklik) bu parametrelere sahip bir kare objesi oluşturuyordu. Bu karelerden türetip bunları çizdirme, boyama veya yazdırma amacıyla kullanmak için Kare adında bir sınıf oluşturduk ve grafik işlemlerini bu sınıf üzerinden yönettik.

Oyunun grafik çizdirme temeli atıldıktan sonra oyunun arka planında yatan asıl algoritmalar tasarlanıp yazılmaya başlandı. Bunların başında oyun alanı üzerinde oyuncular tarafından toplanmak üzere yer alan altınların ve gizli altınların oluşturulup rastgele ama belirli kurallar çerçevesinde dağıtılmasıydı. Bu kurallardan başlıcası: Altınlar dağıtılırken altının herhangi bir oyuncu konumuna isabet etme ihtimaline karşı bu durumun engellenmesi. Eğer altın herhangi bir oyuncu konumuna isabet etmişse tekrar rastgele bir pozisyona atanır. Bir diğer kural ise atanan altının daha önceden var olan bir altın konumuna isabet etmesi. Bu durum da aynı mantıkla engellenerek tüm altınların olması gerektiği şekilde dağıtılması sağlanmıştır. Gizli altınlar ise bu daha önce oluşturulan normal altınlardan belli sayıda altının gizli altına dönüştürülmesiyle oluşturuldu. Son olarak oluşturulan her altın ve gizli altın için istendiği şekilde 5 ile 20'nin katlarından rastgele bir sayı seçilip altınlara miktar değeri olarak atandı. Bu işlemler yapıldıktan sonra sıra oyunu oynayacak oyuncuların programlanmasına geldi.

Oyuncuların programlanma aşamasında temel mantık diğer tüm oyuncuların kalıtım alacağı bir oyuncu sınıfı oluşturup her oyuncuya ait değişkenlerin ve kullanacağı metodların bu sınıfta tanımlanması ve oyuncuların kendi sınıflarında da bunların özelleştirilip her oyuncunun davranış stili için uygun hale getirilmesi hedeflendi.

Oyun tur bazlı çalışır. Her oyuncu her turda kendi davranış stili ile hamle yapar. Oyuncuların kendi davranış stillerine geçmeden önce her oyuncunun her tur temelde yaptığı işlemler aynıdır. Bunlar kısaca:

1.a Oyun başı ise hedef belirle

* Oyna
* Hedefe ulaşıp ulaşmadığını kontrol et
* Ulaştıysan altını al ve yeni hedef belirle
* Ulaşamadıysan diğer turu bekle

1.b Oyun başı değilse zaten belirlenmiş olan hedefin var olup olmadığını kontrol et

2.a Hedef mevcutsa oyna

* Hedefe ulaşıp ulaşmadığını kontrol et
* Ulaştıysan altını al ve yeni hedef belirle
* Ulaşamadıysan diğer turu bekle

2.b Hedef mevcut değilse yeni hedef belirle

* Oyna
* Hedefe ulaşıp ulaşmadığını kontrol et
* Ulaştıysan altını al ve yeni hedef belirle
* Ulaşamadıysan diğer turu bekle

Oyuncular her turda temelde bu algoritmaya dayalı olarak hamle yaparlar. Her hedef belirlemenin ve hamle yapmanın belirli bir maliyeti vardır. Oyuncuları birbirinden ayıran kısım oyuncuların hedef belirlerken kullandıkları algoritmalardır. Bunlar şöyledir:

A oyuncusu hedef belirlerken altın miktarlarına bakmaksızın kendisine ek yakın olan altını hedef olarak belirler.

B oyuncusu hedef belirlerken yakınlığın yanında altın miktarlarını da hesaba katarak kendisine en fazla altın kazandıracak olan en karlı altını hedef olarak seçer.

C oyuncusu da aynı B oyuncusunda olduğu gibi en karlı altını hedef olarak seçer ama C oyuncusunun diğer oyunculardan farklı olarak her tur belirli sayıda kendine en yakın gizli altını açığa çıkarma yeteneği vardır.

D oyuncusu da B ve C oyuncuları gibi en karlı altını hedef olarak seçer ama D oyuncusunun diğer oyunculardan farklı olarak her tur kontrol ettiği bir koşul vardır. Bu, eğer D oyuncusu diğer oyuncuların hedeflerinde onlardan önce gidip o altına sahip olabilme durumudur. Bu koşul sağlanıyorda öncelikle o altını hedef olarak seçer. Ulaşamıyorsa bu altınlar haricinde en karlı altını hedef alır.

C. *Kullanılan Bazı Metodlar ve Classlar*

- ***private void altinlariDagit():***

Bu metod oyunda kullanılacak olan altınların oyun alanı matrisine dağıtılmasını sağlar.

- ***public void altinlariCizdir(PaintEventArgs g, Kare[,] kareler):***

Bu metod dağıtılan altınlara göre hangi konumlara altın çizdirilmesi gerektiğini söyler.

- ***public class Kare:***

Bu sınıf oyun ekranında kullanıcının gördüğü her bloğun çizdirilme işlemini gerçekleştiren sınıftır.

- ***public class Oyuncu:***

Bu sınıf tüm oyuncuların kalıtım aldığı temel değişken ve metodların bulunduğu sınıftır.

D. *Karşılaşılan Bazı Sorunlar*

- Yazının başında bahsettiğim üzere oyun alanını tasarlarken winform objelerini kullanmayı denediğimizde programın aşırı yavaş şekilde çalışması hatta bazı durumlarda çökmeye sebep olmasıydı. Bu sorunu Rectangle sınıfını kullanarak çözdük.

- İkinci bir sorun ise program sonunda gösterilen tabloda yazdırılan değerlerde bir mantıksızlık olmasıydı. Programı adım adım takip edip gerekli düzeltmeleri yaptıktan sonra bu sorundan da kolayca kurtukduk.

E. *Kazanımlar*

- Rectangle sınıfını kullanmayı öğrendik

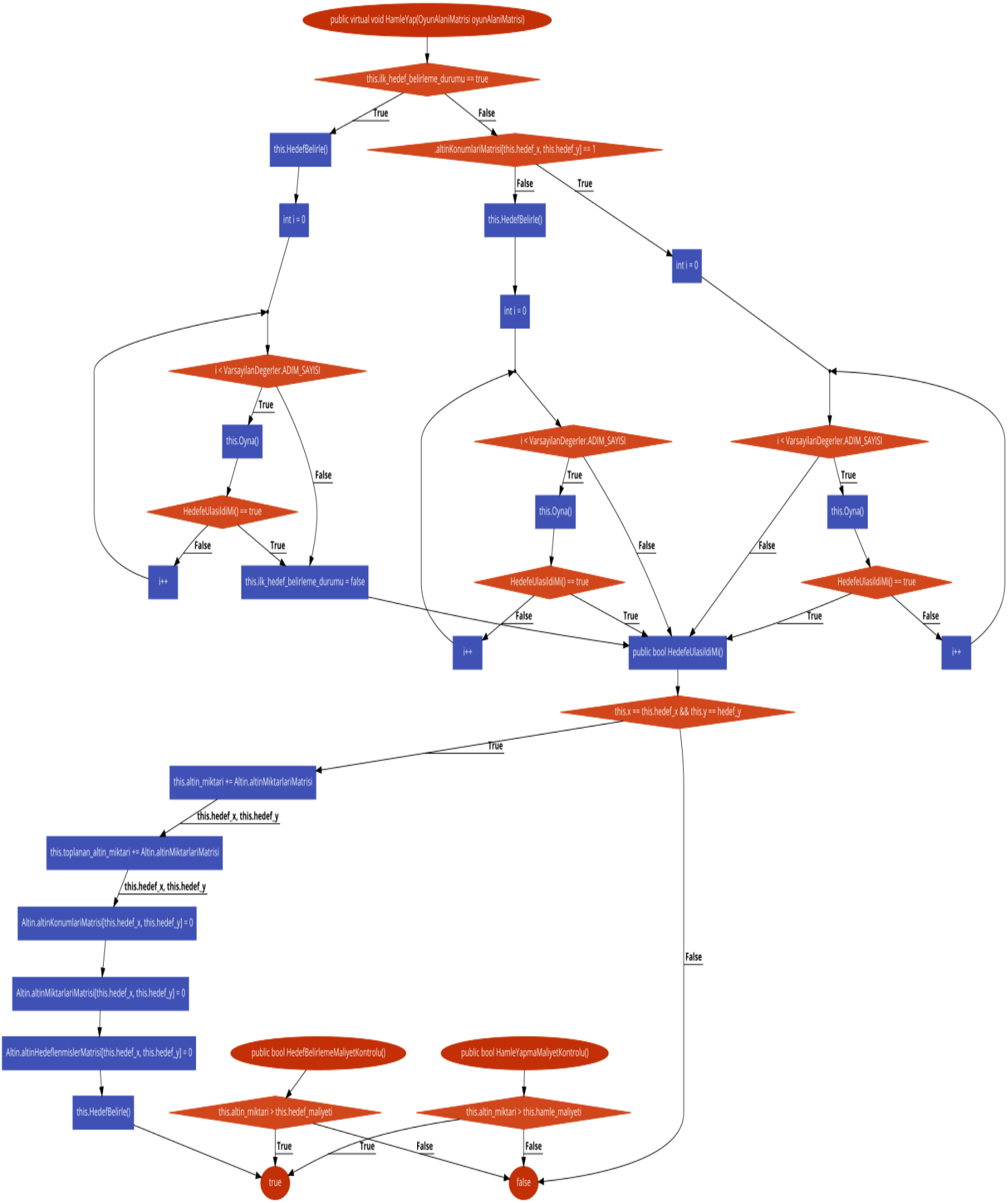
- Çok sayıda winform objesinin programı yavaşlattığını hatta durdurmaya sebep olduğunu öğrendik

- Algoritma becerimize katkıda bulundu

KAYNAKÇA

[1] https://stackoverflow.com/questions/1019793/how-can-i-convert-string-to-int
[2] https://docs.microsoft.com/tr-tr/dotnet/standard/events/
[3] https://stackoverflow.com/questions/3081916/convert-int-to-string
[4] https://www.geeksforgeeks.org/a-search-algorithm/
[5] https://www.youtube.com/watch?v=ySN5Wnu88nE
[6] https://en.wikipedia.org/wiki/Taxicab_geometry

AKIŞ ŞEMASI


public virtual void HamleYap(OyunAlaniMatrisi oyunAlaniMatrisi)
this.ilk_hedef_belirleme_durumu == true
True
False
this.HedefBelirle()
altinKonumlariMatrisi[this.hedef_x, this.hedef_y] == 1
False
True
int i = 0
this.HedefBelirle()
int i = 0
i < VarsayilanDegerler.ADIM_SAYISI
True
this.Oyna()
False
HedefeUlasildiMi() == true
False
True
i++
this.ilk_hedef_belirleme_durumu = false
i < VarsayilanDegerler.ADIM_SAYISI
True
this.Oyna()
False
HedefeUlasildiMi() == true
False
True
i++
i < VarsayilanDegerler.ADIM_SAYISI
True
this.Oyna()
False
HedefeUlasildiMi() == true
True
False
public bool HedefeUlasildiMi()
this.x == this.hedef_x && this.y == hedef_y
True
this.altin_miktari += Altin.altinMiktarlariMatrisi
this.hedef_x, this.hedef_y
this.toplanan_altin_miktari += Altin.altinMiktarlariMatrisi
this.hedef_x, this.hedef_y
Altin.altinKonumlariMatrisi[this.hedef_x, this.hedef_y] = 0
Altin.altinMiktarlariMatrisi[this.hedef_x, this.hedef_y] = 0
Altin.altinHedeflenmislerMatrisi[this.hedef_x, this.hedef_y] = 0
this.HedefBelirle()
False
public bool HedefBelirlemeMaliyetKontrolu()
this.altin_miktari > this.hedef_maliyeti
True
False
public bool HamleYapmaMaliyetKontrolu()
this.altin_miktari > this.hamle_maliyeti
True
False
true
false

UML DİYAGRAMI