**Akıllı Fareler**

**1.Problem Tanımı**

Projede bizden istenilen, labirentin içine akıllı fareler yerleştirilip uygun çıkış kapılarını farelerin kendilerinin bulmasıdır. Kapıları bulurken kullanılacak yöntem bize proje dökümanında anlatılmıştır. İlk olarak ileri, sağ, sol yönlerinden oluşan çeşitli varyasyonları kullanılarak farenin çıkış kapısını bulması istenir. Varyasyonların sayısı projede üç olarak belirlendiği için üç adet fare kullanılmıştır.

Fare çıkış kapısını bulurken kendisine belirtilen yöntemi labirent içinde her ayrım noktasına geldiğinde kullanır .Örnek olarak ileri, sağ, sol faresinin ayrım noktalarından herhangi birinde üç seçenek bulunmaktadır. Fare ilk olarak ileri yönünü kullanarak yoluna devam etmeye çalışacaktır. Eğer farenin önünde gidemeyeceği yol varsa tekrar bu ayrım noktasına gelerek ikinci önceliği olan sağ yönünü kullanarak yoluna devam eder. Öncelik sıralarını kullanırken dikkat etmesi gereken en önemli şey en son ayrım noktasının koordinatlarını aklında tutmasıdır. Farenin çıkış kapısını buluncaya kadar izleyeceği yolu bu şekilde özetleyebiliriz.

Ayrıca projede bizden farenin çıkış kapısına giden yollar içerisindeki en kısa olan yolu bulmamız istenmektedir. Daha sonra bu en kısa yolla farenin, kendi bulduğu yolu karşılaştırmamız beklenmektedir ve bu karşılaştırmayı üç fare içinde yapmamız gerekmektedir.

**2.Sorunlar ve Araştırmalar**

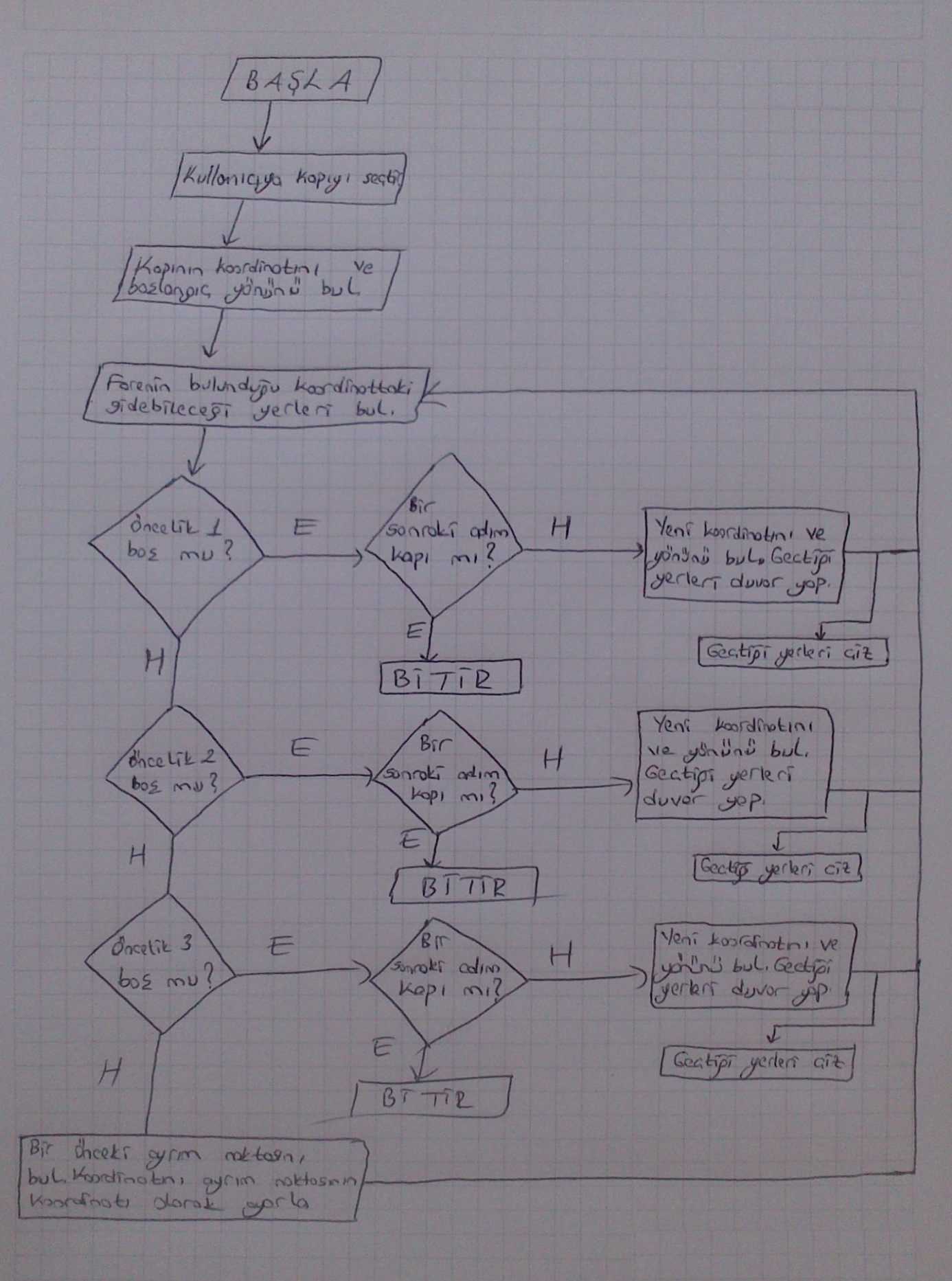
Projeye başladığımızda karşılaştığımız ilk sorun projeyi gerçekleştireceğimiz ortamın belirlenmesi oldu. Önce Windows Phone Application ortamında yazmaya karar verdik. Buna göre öncelikle bu ortamda yapılmış başka bir uygulama inceledik. Uygulamada birçok xaml dosyası olduğunu gördük ancak bizim uygulamamız ekranda yalnızca tek bir görüntü içereceği için başka bir platform üzerinde çalışmaya karar verdik. Karşılaştığımız bu ilk problemin çözümü için internette yapılan diğer Windows Phone uygulamalarına göz attık. Araştırma sırasında XNA Game Stdio platformu altında bulunan Windows Phone Game üzerinde yapılan bir oyun dikkatimizi çekti.***(1)*** Bulduğumuz örneğin kodlarını iyice inceledik ve bizim yapacağımız proje için gerekli olan hemen hemen tüm bilgileri içerdiğini gördük. Son olarak bu platformda çalışmaya karar verdik.

İkinci karşılaştığımız sorun ise grafik ekranına labirentin çizdirilmesi oldu. Problemin çözümü için yukarıda bahsettiğimiz örnek projenin içinde bulunan ekrana çizdirme fonksiyonunu kullandık.

Bir başka sorun ise farenin hareketi sırasında yönünün belirlenmesi ve uygun yönünün çizdirilmesi oldu. Problemin çözümü için internetten çizdirme fonksiyonunu detaylı bir şekilde araştırdık***(2)*** ve kendi uygulamamıza uyarlamak içindeneme-yanılma yöntemini kullandık.

**3.Tasarım**

**3.1. Akış Şeması**



**4.Genel Yapı**

Projede bizden .txt uzantılı dosyadan okuma yapmamız istendiği için c# ta dosyadan okuma işlemini araştırdık ***(3)*** .

Textten okuma işlemini gerçekleştirirken ilk olarak labirent adında 15 \* 25 boyutlu bir dizi tanımladık. Dizi elemanlarının değeri 0 ve 1 olmaktadır. Labirentte duvar var ise 0, yoksa 1 değerini almaktadır. Ardından oluşturduğumuz bu diziyi ısolatedstorage sınıfı yardımıyla sanal text dosyasına yazdırdık. Son olarak da yine ısolatedstorage sınıfı yardımıyla sanal text te oluşturduğumuz dosyadan okuma işlemini gerçekleştirdik.

Ardından okumuş olduğumuz dosyamızı içindeki labirent bilgisini tutmak üzere labirent adında 15\*25 boyutunda bir dizi tanımladık. Üç fare içinde ileriki adımlarla labirent üzerinde işlem yapacağımız için labirent\_ilk, labirent\_iki ,labirent\_üç adında üç adet daha dizi oluşturduk.

Farenin labirent içindeki konumunu sürekli olarak elimizde tutmak için koordinat adında 200\*3,çift boyutlu dizimizi oluşturduk.200 değerini vermemizin sebebi farenin çıkış kapısını bulana kadar izleyeceği adım sayısını tahmini değeridir. Farenin bulunduğu konumdan bir sonraki konuma gidip gitmemesini kontrol etmemiz gerekir. Kontrolü yaparken farenin bulunduğu konumdan bir sonraki gidebileceği yerleri aklımızda tutmak için gidebileceği\_yer adında 200\*4 boyutunda dizi tanımladık.

Dizinin ikinci boyutunun ilk indisi ileri, ikinci indisi sağ, üçüncü indisi de sol değerlerini tutmak üzere tasarlanmıştır. Örneğin farenin bulunduğu konumdan bir sonraki ileri konumu eğer boş ise dizinin ikinci boyutunun ilk indisi olan ileri değeri 1, farenin bulunduğu konumdan bir sonraki sağ konumu eğer boş ise dizinin ikinci boyutunun ikinci indisi olan sağ değeri1, farenin bulunduğu konumdan bir sonraki sol konumu eğer boş ise dizinin ikinci boyutunun üçüncü indisi olan sol değeri 1 olacaktır. Aksi takdirde farenin bulunduğu konumdan bir sonraki ileri konumu eğer dolu ise(yani duvar var ise) dizinin ikinci boyutunun ilk indisi olan ileri değeri 0, farenin bulunduğu konumdan bir sonraki sağ konumu eğer dolu ise dizinin ikinci boyutunun ikinci indisi olan sağ değeri 0, farenin bulunduğu konumdan bir sonraki sol konumu eğer dolu ise dizinin ikinci boyutunun üçüncü indisi olan sol değeri 0 olacaktır.

Projede 3 adet fare bulunduğu için ve bu farelerin öncelik sıraları farklı olduğu için, öncelik sıralarını tutmak üzere fare\_öncelik adında 1\*3 cift boyutlu dizisini oluşturduk. Dizimizde ileri değerini 10, sağ değerini 11, sol değerini ise 12 temsil etmektedir. Örneğin farenin öncelik değeri ileri, sağ, sol ise fare\_oncelik dizimizin değerleri 10,11,12 olacaktır.

Farenin bulunduğu konumundaki yönleri farklı olabileceğinden dolayı farenin her adımdaki yönünü tutmak üzere yön adında değişken tanımladık. Bu değişkenimizin değeri fare eğer yukarı yönüne bakarsa 2, sağ yönüne bakarsa 6, sol yönüne bakarsa 4, aşağı yönüne bakarsa 8 değerini almaktadır.

Uygulama açıldığında kullanıcıdan herhangi bir kapı seçimi yapılması beklenmektedir. Eğer kullanıcı herhangi bir kapı seçmediyse fare labirentin içine girmeyecektir. Bu koşulu sağlamak üzere a adında bir değişken tanımladık ve ilk değerini sıfıra eşitledik. Eğer kullanıcı herhangi bir kapı girişine tıklarsa bu değişkenimizin değeri bir olacaktır ve uygulamamızda fare labirentin içine girip dolaşmaya başlayacaktır.

**4.1.Fonksiyonlar**

***ilk\_deger();***

Bu fonksiyon ilk olarak farenin öncelik değerlerini oluşturur. Örneğin farenin önceliği ileri, sağ, sol ise bu değerleri tutmak üzere fare\_oncelik dizisini kullanır. Ardından farenin gidebileceği yerlerin başlangıç değeri olarak -1 e atar. Son olarak farenin koordinatının başlangıç değerlerini -1 e atar.

***gidebileceği\_yer\_son\_eleman\_hesapla();***

Bu fonksiyon, farenin bulunduğu konumdan bir sonraki gidebileceği yerleri tutan gidebileceği\_yer dizisinin son indisi olan 3. indisine, diğer indislerinin toplamını atar. Bu işlemi yapmamızın sebebi farenin herhangi bir çıkmaz yola geldiğinde geri ayrım noktasına atlaması için kolaylık sağlamaktır.

***gidebileceği\_yer\_guncelle();***

Proje içinde farenin konumu sürekli değişeceğinden dolayı gidebileceği yerleri her adımsa güncellemek için kullanılır.

***atla();***

Eğer fare uygulama içinde çıkmaz yola geldiyse bir önceki ayrım noktasına geri dönmesini sağlar.İşlemi gerçekleştirdikten sonra farenin yeni koordinatını bulur ve yeni yönünü tayin eder.

***kontrol();***

Farenin bulunduğu konumdan bir sonraki gideceği yolun herhangi bir çıkış kapısı olup olmadığını kontrol eder.

***menu();***

Uygulamamızın en önemli fonksiyonudur.Farenin adımlarını gerçeklediğimiz yerdir.

İlk olarak farenin bulunduğu konumdaki yönüne bakar.Eğer farenin bir sonraki adımı kapı ise bu fonksiyona hiç girmeden direkt çıkış yapar. Aksi takdirde fonksiyon akışına devam eder.

Farenin gidebileceği yerleri bulur. Farenin bulunduğu koordinatı, yönü ve gidebileceği yerler artık bu fonksiyonumuzda hazır olarak bulunmaktadır. Bundan sonra yapacağımız işlem farenin öncelik sırasına göre adımlarına karar vermek olacaktır.

***cizdirme();***

Bu fonksiyon farenin her adımını piksel cinsinden çizdirmeyi sağlar. İlk olarak kullanıcının herhangi bir kapı seçip seçmediğini “touchcollection” sınıfı yardımı ile kontrol eder. Eğer kullanıcı herhangi bir kapı girişi seçti ise o kapının bulunduğu koordinatı farenin ilk koordinatları olarak atama işlemini yapar. Farenin harekete başladığı andan itibaren bu fonksiyon sürekli olarak çalışır.

Farenin çıkış kapısına kadar gerçekleştireceği adımlar labirent2 adlı dizimizde bulunmaktadır. Fonksiyon içinde bu adımları tek tek kontrol edip farenin bir sonraki adımı arasındaki uzaklık “math.sqrt” fonksiyonu yardımıyla hesaplanır. Ayrıca farenin eski konumundan hangi yöne doğru hareket ettiğini kontrol ederiz. Kontroller sonucu fare eğer bir adım attıysa koordinatını birer birer arttırarak farenin kaymasını sağlarız. Eğer fare çıkmaz yola girip de birden fazla adım atması gerekirse farenin yeni koordinatlarına, atlaması gereken koordinatların değerlerini atarız.

**4.2.Adımlar**

Eğer farenin öncelik sırasındaki ilk değeri ileri komutu ise ve farenin ileri yönündeki koordinatında duvar yoksa fare artık ileri yönündeki koordinata gider. Bu işlemi gerçekleştirdikten sonra farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için gidebileceği yerler dizisindeki ilgili indisler sıfırlanır. Aynı şekilde farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için labirent dizisindeki ilgili indisler sıfırlanır. Ayrıca farenin gittiği yolları tutan labirent2 dizisindeki başlangıç değerleri sıfır olan ilgili indislerin değeri birer artar. Son olarak farenin yeni bulunduğu konumun koordinat değerlerini tutmak üzere koordinat dizisine atama işlemleri gerçekleştirilir ve yeni yön tayin edilir.

Eğer farenin öncelik sırasındaki ilk değeri ileri komutu ise ve farenin ileri yönündeki koordinatında duvar varsa farenin ikinci önceliğine bakılır. İkinci önceliğinde bulunan komutun da labirent üzerinde duvar yoksa fare artık sağ yönündeki koordinata gider. Bu işlemi gerçekleştirdikten sonra farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için gidebileceği yerler dizisindeki ilgili indisler sıfırlanır. Aynı şekilde farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için labirent dizisindeki ilgili indisler sıfırlanır. Ayrıca farenin gittiği yolları tutan labirent2 dizisindeki başlangıç değerleri sıfır olan ilgili indislerin değeri birer artar. Son olarak farenin yeni bulunduğu konumun koordinat değerlerini tutmak üzere koordinat dizisine atama işlemleri gerçekleştirilir ve yeni yön tayin edilir.

Farenin öncelik sırasındaki ikinci değeri sağ komutu ise ve farenin sağ yönündeki koordinatında duvar varsa farenin üçüncü önceliğine bakılır. Üçüncü önceliğinde bulunan komutun da labirent üzerinde duvar yoksa fare artık sol yönündeki koordinata gider. Bu işlemi gerçekleştirdikten sonra farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için gidebileceği yerler dizisindeki ilgili indisler sıfırlanır. Aynı şekilde farenin geçmiş olduğu yollardan bir daha geçmesini engellemek için labirent dizisindeki ilgili indisler sıfırlanır. Ayrıca farenin gittiği yolları tutan labirent2 dizisindeki başlangıç değerleri sıfır olan ilgili indislerin değeri birer artar. Son olarak farenin yeni bulunduğu konumun koordinat değerlerini tutmak üzere koordinat dizisine atama işlemleri gerçekleştirilir ve yeni yön tayin edilir.

Farenin öncelik sırasındaki üçünü değeri sol komutu ise ve farenin sol yönündeki koordinatında da duvar varsa fare bir önceki ayrım noktasına direkt olarak atlar.

Yukarıda anlattığımız adımları yön 2 (yukarı) için anlattık. Yani farenin yüzü yukarıya dönük durumu söz konusudur.

Farenin yüzü sola (yön 4) doğru ise ileri yönünü sol, sağ yönünü ileri, sol yönünü aşağı olarak belirleriz. Yukarıdaki adımları aynen uygularız.

Farenin yüzü sağa (yön 6) doğru ise ileri yönünü sağ, sağ yönünü aşağı, sol yönünü yukarı olarak belirleriz. Yukarıdaki adımları aynen uygularız.

Farenin yüzü aşağı (yön 8) doğru ise ileri yönünü aşağı, sağ yönünü sol, sol yönünü sağ olarak belirleriz. Yukarıdaki adımları aynen uygularız.

**5.Kazanımlar**

Windows Phone için XNA Game Stdio platformunu kullanarak mobil oyun uygulaması geliştirdik. XNA Game Stdio’da şekil çizdirmeyi, şekilleri hareket ettirmeyi, şekillere yön vermeyi öğrendik.

**6. Referanslar**

**Article in a Popular Magazine**

**(1)** [**http://www.yazgelistir.com/makale/windows-phone-7-de-xna**](http://www.yazgelistir.com/makale/windows-phone-7-de-xna)

**(2)** [**http://msdn.microsoft.com/en-us/library/microsoft.xna.framework.graphics.spritebatch.draw.aspx**](http://msdn.microsoft.com/en-us/library/microsoft.xna.framework.graphics.spritebatch.draw.aspx)

**(3)** [**http://stackoverflow.com/questions/13568888/windows-phone-reading-level-from-txt-file**](http://stackoverflow.com/questions/13568888/windows-phone-reading-level-from-txt-file)