**LABİRENT OYUNU**

*Erhan ÖZDOĞAN*

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

ozerhan9@gmail.com

**ÖZET**

Labirent oyunu projesi bir ana fonksiyon ve bu fonksiyonun yanında on dört adet yan fonksiyondan oluşmaktadır.

Labirent oyunu programı kullanıcıdan istediği satır ve sütun sayısına göre tek yönlü bağlantılı liste ile labirent oluşturur bunu yaparken gerekli fonksiyonlarını kullanır. Daha sonra kullanıcıdan labirentin giriş ve çıkış koordinatlarını alır bunu alırken gerekli kontrolleri yapar.

Daha sonra labirenti çözme kısmına geçer bu kısımda yığın kullanarak labirentin içinde dolaşır. Default olarak sağa gider eğer duvarsa yön değiştirir. Yön değiştirmeleri sıralı olarak gerçekleştirir kuzey, doğu, güney, batı şeklindedir. Aynı zamandan her gittiği noktayı bir yığın yapısının içine atar yığın yapısı da yine bağlantılı liste kullanılarak oluşturulmuştur. Eğer program bir çıkmaz noktaya geldiyse yığına attığı son noktayı alarak o noktaya gider ve oradan yön değiştirmeye devam eder. Bu süreç çıkış yolunu bulana kadar veya yığından eleman kalmayana kadar devam eder. Yığında eleman kalmadıysa çıkış yolu yok anlamına gelmektedir. Çıkış yolu bulunduğunda ise yolun koordinatları yığında bulunmaktadır matris üzerinde bu koordinatlar işaretlenir ve programdan çıkılır.

**GİRİŞ**

Labirent oyunu projesi genel olarak kullanıcıdan alınan verilere göre bir labirent oluşturup yine kullanıcıdan alınan verilere göre çıkış yolunun bulunmaya çalışılmasından ibarettir. Projede gerçekleştirilen işlemler kabaca şunlardır:

* Kullanıcıdan verilerin alınması
* Verilere göre labirent oluşturma
* Giriş ve çıkış kapılarına göre labirenti çözme
* Varsa çıkış yolunun gösterilmesi

Bu projede amaç bağlantılı liste ve yığın kullanılarak rastgele oluşturulan labirentler üzerinden çıkışa ulaşmaktır.

Aynı zamanda proje gerçekleştirilirken Programlama derslerinde ve Veri yapıları derslerinde görülen teorik bilgilerin birleştirilmesi ve uygulamada kullanılarak kalıcılığının sağlanması kazanımlarının elde edilmesi amaçlanmıştır.

**TEMEL BİLGİLER**

Labirent oyunu projesinde programlama dili olarak C dili ve geliştirme ortamı (IDE) olarak Code blocks ve DevC++ kullanılmıştır.

**C Programlama Dili:**1970'lerin başında Ken Thompson ve Dennis Ritchie tarafından UNIX işletim sistemi için geliştirilmiş yüksek seviye bir programlama dilidir.Daha çok sistem programlama için kullanılan bir dildir.C# ve C++ dilleri C dilinden türetilmiştir.

**Geliştirme Ortamı (IDE):** IDE bilgisayar programcılarının hızlı ve rahat bir şekilde program geliştirebilmesini amaçlayan, geliştirme sürecini organize edebilen birçok araç ile birlikte geliştirme sürecinin verimli kullanılmasına katkıda bulunan araçların tamamını içerisinde barındıran bir yazılım türüdür.

**PROJE BÖLÜMLERİ**

**1-) ANA FONKSİYON**

Main fonksiyon içinde kullanıcıdan labirenti oluşturmak için satır ve sütun sayıları alınır. Labirenti oluşturacak fonksiyonlar çağırılır. Aynı zamanda ayrı bir matris yine bağlı liste fonksiyonları kullanılarak oluşturulmuştur. Bu labirent ve matris oluşturulurken labirentin boyutu kadar dönen bir döngü içinde bağlı liste fonksiyonları çağırılmıştır. Daha sonra labirentin giriş ve çıkış koordinatları kullanıcıdan istenir ve bu koordinatların doğruluğu kontrol edilir ve labirent çözen fonksiyon çalıştırılır. Labirent çözüldükten sonra ise yığındaki koordinatlara göre matrisin üzerinde gerekli değişiklikler yapılır ve oluşan yol matris üzerinde gösterilerek ekrana yazdırılır.

**2-) DÜĞÜM OLUŞTURMA FONKSİYONU**

Bu fonksiyon labirenti oluşturmak için bağlantılı liste düğümleri oluşturur ve parametre olarak aldığı i indisi , j indisi ve veriyi malloc ile bellekten yer alarak oluşturduğu düğümün değişkenlerine atar ve oluşturduğu düğümü döndürür. Burada parametre olarak aldığı veri labirentin yollarını modeller 1 veya 0 değerleridir 1 ise yol var 0 ise yol yok demektir. Bu fonksiyonun aynısından main fonksiyonunda bahsedilen matris içinde oluşturulmuştur.

**3-) LİSTEYE EKLEME FONKSİYONU**

Bu fonksiyon düğüm oluşturma fonksiyonundan döndürülen düğümü bağlı listeye ekler listeye eklerken listenin boş olup olmama kontrolünü yapar eğer liste boşsa ilk eleman olarak ekler eğer boş değilse global değişken olarak tanımlanan ilk ve son değişkenleri kullanılarak listenin sonuna eklenir. Bu fonksiyonun aynısından main fonksiyonunda bahsedilen matris içinde oluşturulmuştur.

**4-) MATRİS YAZDIRMA FONKSİYONU**

Bu fonksiyon labirentin sütun sayısını parametre oalrak alır. Düğüm tipinde bir değişken oluşturup labirentin ilk düğümünü bu düğüme atar ve bağlantılı liste üzerinden giderek düğümlerin verilerini ekrana yazar parametre oalrak aldığı sütun sayısını ise matris halinde yazmak için yani alt satıra geçmek amacıyla kullanır. Bu fonksiyonun aynısından main fonksiyonunda bahsedilen matris içinde oluşturulmuştur.

**5-) KAPI BUL FONKSİYONU**

Bu fonksiyon parametre olarak x ve y koordinatlarını alır düğüm tipinde değişken oluşturup labirentin ilk düğümünü bu düğüme atar daha sonra düğümlerin üzerinden giderek ilk önce her düğümün i ve j değerinin x ve y parametrelerine eşit olup olmadığına bakar eşitse verisinin 1 olup olmadığına bakar eğer 1 ise true döndürür değilse sonraki false döndürür. Bu fonksiyonun aynısından main fonksiyonunda bahsedilen matris içinde oluşturulmuştur.

**6-) YIĞIN YENİ DÜĞÜM FONKSİYONU**

Bu düğüm yığın düğüm tipinde bir değer döndürür ve parametre olarak düğüm tipinde bir veri alır yığın düğüm tipinde bir değişken oluşturup parametre olarak aldığı veriyi bu değişkene atar ve yığın düğüm tipindeki bu değişkeni geri döndürür.

**7-) YIĞINA EKLEME FONKSİYONU**

Bu fonksiyon parametre olarak düğüm tipinde bir veri alır ve yığın düğüm tipinde bir değişken oluşturup bu değişkene 6 numaralı fonksiyon olan yeni düğüm fonksiyonundan dönen değeri atar. Yeni düğüm fonksiyonu yığına ekleme fonksiyonunda çağırılır. Daha sonra global değişken olan kök değişkenine oluşturulan yığın düğüm tipindeki bu değişken atanır. Böylece veri yığına eklenmiş olur.

**8-) BOŞMU FONKSİYONU**

Bu fonksiyon yığının boş olup olmadığını kontrol eder parametre almaz geriye değer olarak kök değişkeninin değilini döndürür eğer kök null ise geriye true döner null değilse false döner.

**9-) DÜĞÜM AL FONKSİYONU**

Bu fonksiyon parametre almaz ve geriye düğüm tipinde bir değişken döndürür. Yığın düğüm tipinde bir değişken oluşturur ve bu değişkene global değişken olan kök ü atar. Daha sonra kökü yığındaki bir önceki elemana kaydırır ve oluşturulan değişkeni geri ye döndürür ve yığından siler. Bu işlemleri yapmadan önce yığının boş olup olmadığının kontrolünü 8 numaralı fonksiyon olan boşmu fonksiyonunu çağırarak yapar.

**10-) DÜĞÜM BELİRLE FONKSİYONU**

Bu fonksiyon struct düğüm tipindedir ve parametre olarak int tipinde x ve y koordinatlarını alır düğüm tipinde bir değişken oluşturarak bu değişken üzerinden bağlantılı listede gezer ve düğümlerin i ve j verilerini x ve y ile karşılaştırır eğer herhangi bir düğümün i ve j verisi parametre olarak alınan x ve y koordinatlarına eşitse fonksiyon bu düğümü geri döndürür. Eğer eşit değilse sonraki düğüme geçer. Bu fonksiyonun aynısından main fonksiyonunda bahsedilen matris içinde oluşturulmuştur.

**11-) LABİRENTİ ÇÖZEN FONKSİYON**

Bu fonksiyon void tipindedir yani geri bir değer döndürmez ve parametre olarak labirentin satır ve sütun değerlerini bunun yanında giriş ve çıkış kapılarının x ve y koordinatları olmak üzere toplam altı değişken alınır.

Struct düğüm\* tipinde bir değişken oluşturur ve bu değişkene giriş koordinatlarını içieren düğüm düğüm belirle fonksiyonu çağırılarak atanır. Ve bir tanede ikinci matris için struct düğüm\* tipinde bir değişken oluşturulur ve bu değişkenede giriş koordinatlarından gelen düğüm atanır ve verisine 1 atanır.

Değişkenler oluşturulup değerleri atandıktan sonra sonsuz bir döngüye girilir sırayla kuzeye, doğu , güney ,batı yönlerine gidilir. Gidilirken döngü içinde bir if kontrolü yapılır bu kontrolde yönlere uygun kontroller yapılır örneğin kuzeye gidilecekse bir üst satıra geçilmesi gerek yani düğümün i indisi bir azaltılması gerekir if koşulunun içinde i indisinin bir eksi değerinin 0 dan büyük eşit olması ve gidilecek indisin verisinin 1 yani yol olması şartlarının yanında ekstra matristeki o noktanın 1 olmama şartı aranır eğer bunlar sağlanırsa nokta yığına eklenir ve üstte oluşturduğumuz değişkenlere bu nokta atanır daha sonra gidilen noktanın çıkış noktamız olup olmamasına bakılır çıkış noktamız ise döngüden çıkılır.

Eğer gidilicek yer kalmadıysa önce yığının boş olup olmamasına bakılır yığın boş ise çözüm yoktur diyerek dönüden çıkılır eğer yığın boş değilse bir önceki düğüme gidilir ve başka yollar aranır.

**SONUÇLAR**

Labirent oyunu projesinde:

* Link list kullanılarak kullanıcıdan alınan bilgilere gör labirent oluşturulmaktadır labirent oluşturulurken dizi kullanılmamıştır.
* Yine bağlı liste ile labirentin düğümleri arasında dolaşılmış ve uygun düğümler yine bağlı liste ile oluşturulan yığına aktarılmıştır. Yığın oluşturulurken dizi kullanılmamıştır.
* Labirent başarılı bir şekilde çıkış yolu bulunmuştur ve konsolda matris üzerinde çıkış yolu gösterilmiştir. Eğer çıkış yolu yoksa gerekli uyarı verilebilmiştir.
* Labirent çözülürken çıkış yolu bulunmuştur ancak bu çıkış yolu en kısa yol olmayabilmektedir.

**KAYNAKÇA**

[1] Yorulmaz,M. ve Yorulmaz,S. “Programlamayı C ile Öğreniyorum”,2016

[2] Gök,O. ve Şahin,S.”Yığın ve Kuyruk Yapıları”, http://embedded.kocaeli.edu.tr/veri\_1718\_6/

[3] Gök,O. ve Şahin,S.”Bağlantılı Listeler”, http://embedded.kocaeli.edu.tr/veriyapilari\_1718\_5/

[4] http://www.codesenior.com/tutorial/Labirent-Cozme-Algoritmalari

[5](labirent çözme Algoritması oluşturmada yararlanıldı)

https://selcukaltintav.wordpress.com/2014/07/02/stack-veri-yapisiyla-problem-cozmeklabirent-algoritmasi/