

MESLEK YÜKSEK OKULU

Bölüm Adı : Bilgisayar Teknolojileri

Program Adı : Bilgisayar Programcılığı

Ders Adı : Yapay Zeka

Proje Adı : Minimum Yol Problemi

Ad Soyad : Aydın Can SAKARYA

Okul No: 18MY03033

**Yapay Zeka Nedir ?**

En basit ifadeyle yapay zeka (AI), görevleri yerine getirmek için insan zekasını taklit eden ve topladıkları bilgilere göre yinelemeli olarak kendilerini iyileştirebilen sistemler veya makineler anlamını taşır. Yapay zeka pek çok biçimde kendini gösterebilir. Örneğin:

* Sohbet robotları, müşterilerin sorunlarını daha hızlı bir şekilde anlamak ve daha verimli cevaplar vermek için yapay zekadan yararlanır
* Akıllı asistanlar, zamanlamayı iyileştirmek için büyük kullanıcı tanımlı veri kümelerinden kritik bilgileri çekmek için yapay zekadan yararlanır
* Öneri motorları, kullanıcıların izleme alışkanlıklarına göre TV programları için otomatik öneriler sunabilir

Yapay Zeka, herhangi bir özel biçim veya işlevden ziyade süper güçlendirilmiş düşünce ve veri analizi yeteneği ve süreci ile ilgilidir. Yapay Zeka dendiğinde zihinlerde dünyayı ele geçiren çok fonksiyonel, insan benzeri robotlar canlansa da yapay zeka insanların yerine geçmek üzere tasarlanmamıştır. İnsan yeteneklerini ve katkılarını önemli ölçüde geliştirmek üzere tasarlanmış ve bu nedenle oldukça değerli bir ticari varlıktır.

En Kısa Yol Problemi

En kısa yolu bulmak, iki düğüm arasında en maliyetle gidilebilen bir yolun varlığını belirleme problemidir. Çalışmada en kısa yolu bulmak için kullanılan algoritmalara yer verilmiştir.

1. Dijkstra Algoritması

Dijkstra algoritması, bir düğümden diğer tüm düğümlere en kısa yolları hesaplar. Bir başka ifadeyle belirli bir başlangıç noktasına göre en kısa yolu belirleyen algoritmadır. Ağırlıklı ve yönlü graflar için geliştirilmiş olup kenarların ağırlık değeri sıfır ya da sıfırdan büyük bir değer olmalıdır. Eğer kenarların değerleri sıfırdan küçük oluyorsa daha genel algoritmalardan Bellman-Ford kullanılabilir. Dijkstra algoritmasının zaman karmaşıklığı genel olarak 𝑂(𝑀𝑙𝑜𝑔𝑁) şeklinde hesaplanmıştır.

2. Bellman ve Ford

2. Bellman ve Ford Dijkstra gibi bir düğümden diğer tüm düğümlere olan en kısa yolu belirler. Bellman Ford algoritması, Dijkstra farklı olarak negatif değere sahip grafların için de doğru olarak çalışır. Ancak burada da eksi maliyetli çevrimler olmamalıdır.

3. Floyd Algoritması

Graf üzerindeki her bir düğüm için diğer tüm düğümlere olan en kısa yolları4 belirleyen algoritmadır. En kısa yol bulmak için kullanılan en genel algoritmadır. Genellikle yoğun graflarda kullanılması tercih edilir. Grafın komşuluk matrisi şeklinde tutulması durumunda Floyd algoritması 𝑂(𝑁 3 ) karmaşıklığında olmaktadır.

En Kısa Yol Problemi’nin Görüldüğü Yerler

En kısa yol problemi’ni çözen çeşitli algoritmalardan daha sonra bahsedeceğiz, ancak daha önce bu problem nerelerde görülür onlardan bahsetmek istiyorum.

* Navigasyon cihazlarında güzergah ayarlaması sırasında
* Dynamic Routing kullanan Network’lerde en ucuz yada hızlı hat ayarlaması yaparken
* Limancılık, havacılık, postacılık güzergah ayarlaması esnasında
* Hatta Rubik Cube’ü en az hamlede çözmek için dahi kullanılabilir.
* Robot hareket planlaması

Algoritmanın Karmaşıklığı

Algoritmanın karmaşıklığına bakalım. Algoritmanın adımlarını kontrol edelim.

* İlk adımda yeni düğüm ekleniyor ve her düğümün ağırlığı Bellman Ford ile hesaplanıyor. Bu durumun getirdiği karmaşıklık O(VE)’dir.
* Daha sonra negatif kenarlardan kurtarmak için reweighting işlemi yapılıyor. Bu durumun getirdiği karmaşıklık O(E)’dir.
* Her düğüm için Djikstra Algoritması’nın getirdiği karmaşıklık O(V2 logV + VE logV )’dir.

Bu durumda Johnson Algoritması’nın karmaşıklığı O(V2 logV + VE logV) olarak hesaplanır.

(V: Düğümler Kümesi , E: Bağlantılar Kümesi )

Uygulama Ortamı

C# dili ve Visual Studio 2017 uygulamasını kullacağım.

Uygulama Yapım Aşamaları

1) En kısa yol ağacına dahil olan köşeleri, yani minimum mesafesi hesaplanan ve sonlandırılan bir dizi sptSet (en kısa yol ağacı seti) oluşturdum. Başlangıçta, bu küme boştur.

2) Giriş grafiğindeki tüm köşelere bir mesafe değeri atadım. Tüm mesafe değerlerini INFINITE olarak başlattım. Kaynak köşe noktası için uzaklık değerini önce seçilecek şekilde 0 olarak belirledim.

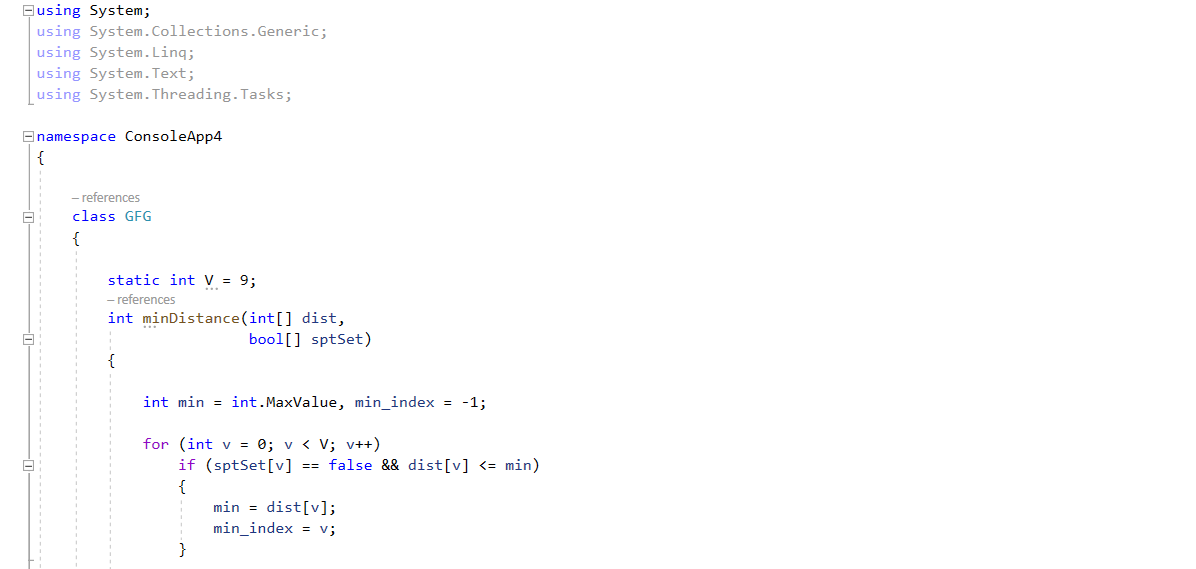
3) sptSet tüm köşeleri içermese de;

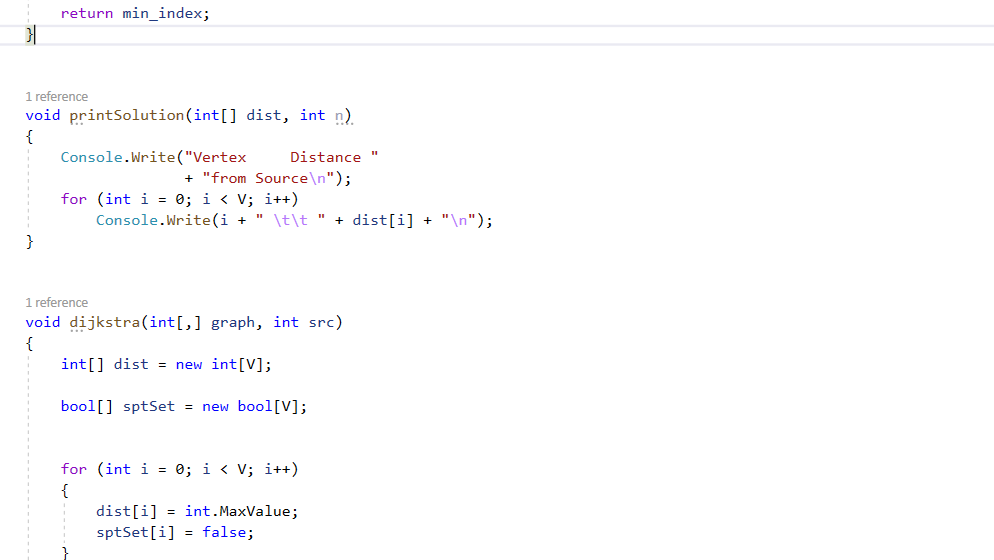
A) sptSet içinde bulunmayan ve minimum mesafe değeri olan bir tepe u seçtim.

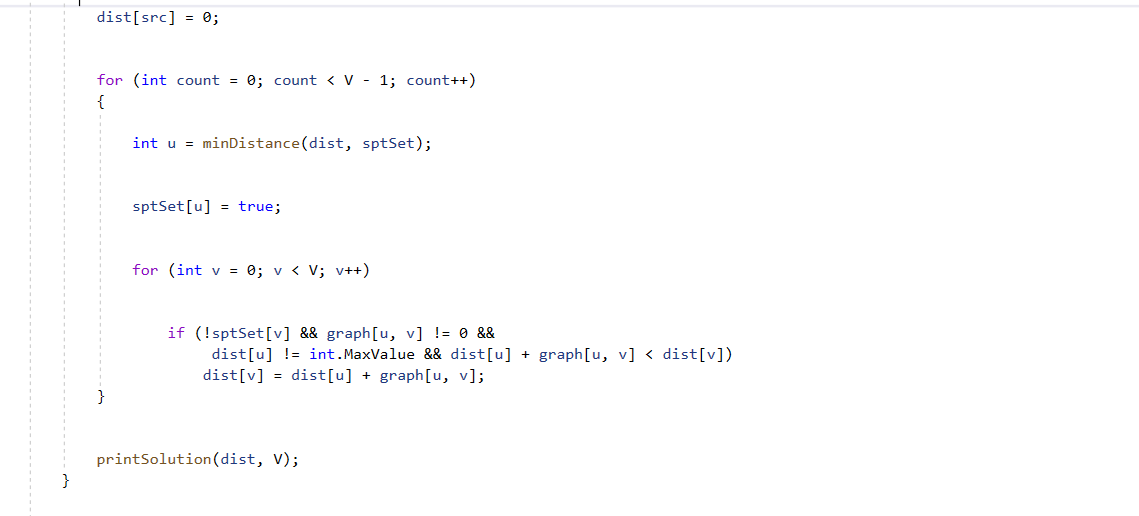
B) u sptSet öğesine ekledim.

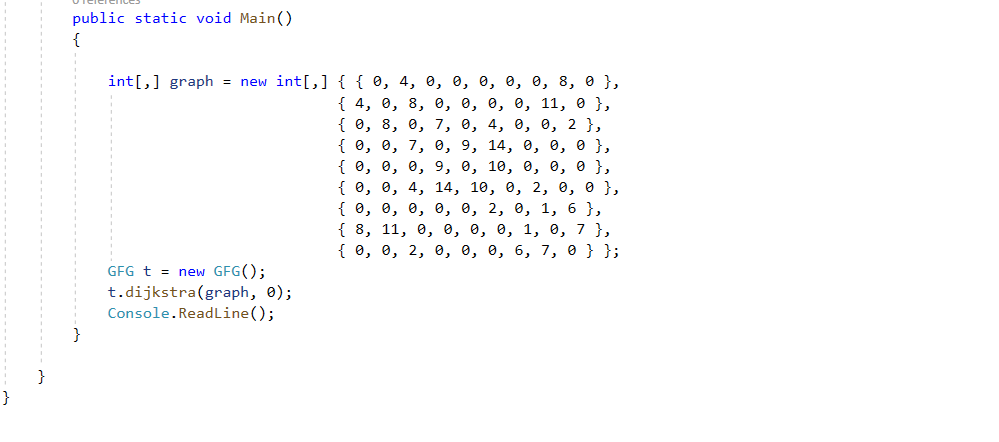
C) Tüm bitişik u köşelerinin mesafe değerini güncelledim. Mesafe değerlerini güncellemek için tüm bitişik köşelerde yenileme yapıyoruz. Her bitişik köşe v için, u'nun mesafe değeri ve u-v kenarının ağırlığı toplamı v'nin uzaklık değerinden küçükse, v'nin mesafe değerini güncelledim.

Proje Kodlarım









Kaynakça

<http://www.zafercomert.com/Medya/2015_05_11_2_121_69f9a888.pdf>

<https://www.oracle.com/tr/artificial-intelligence/what-is-artificial-intelligence.html>

<https://slideplayer.biz.tr/slide/13436554/>