©Ali Galip Bayrak - ODTÜ Bilgisayar Topluluğu

ODTÜ Bilgisayar Topluluğu Üniversite Öğrencileri Arası X. Geleneksel Programlama Yarışması Ön Eleme Soruları

Topluluğumuz, 1997'den bu yana geleneksel olarak düzenlediği programlama yarışması serisine bu sene onuncusunu ekliyor. Programlama yarışması, Ulusal Bilim Olimpiyatları formatında, C, C++ ve Java dilleri üzerinden yapılan ve soruları bilgisayar bilimleri alanının temel problemlerinden ilham alan bir yarışmadır. Yarışmamız, dünyadaki benzerleri arasında (ACM, Tübitak, IOI, vs...) Linux platformunda düzenlenmiş yarışmaların ilki olma ayrıcalığına sahiptir.
Ön eleme sorularının son gönderim tarihi 13 Nisan 2007'dir. Ön katılımcılar arasından bu sorular yoluyla belirlenecek yaklaşık 20 finalist, 29 Nisan 2007 tarihinde ODTÜ Bilgisayar Mühendisliği Bölümü'nde düzenlenecek olan finale çağrılacaktır. Özel ödüllü soruyu en iyi çözen yarışmacı ve final sonucunda ilk üç dereceyi alan finalistler; ödüllerini aynı akşam ODTÜ 'de düzenlenecek olan törende alacaklardır.
Sorular için süre ve bellek sınırlarına, soruların girdi-çıktı kısıtlarına ve diğer teknik detaylara web sayfamızdan "http://yarisma.cclub.metu.edu.tr" erişebilirsiniz.
Her türlü sorularınız ve daha ayrıntılı bilgi için yarisma@cclub.metu.edu.tr adresine mail atabilirsiniz.

PERVANELER

Yıl 2200... Küresel ısınma sonucu yaşanılmaz bir yer haline gelen dünyada, bu olaya çözüm bulmak isteyen bilim adamları bir araya gelip fikirlerini paylaşmak isterler. Bunun üzerine dünyanın dört bir yanından gelen bilim adamları İstanbul'da toplanır. Ünlü fizikçi Barış Tanrıkulu gökyüzüne dev pervaneler yerleştirilip soğutmanın sağlanabileceğini düşündüğünü belirtir. Bunu şu şekilde açıklar; dev pervaneler yerleştirildikten sonra birisi hareketе geçirilecek, bu pervane dönerken çarptığı başka pervaneleri harekete geçirecek ve bu şekilde tüm pervanelerin dönmesi sağlanacak. Bunun üzerine bilim adamları bunun çok ciddi bir hava akımı yaratabileceğini, bu yüzden bir anda pervanalerden yalnızca birisinin çalışması gerektiğini, bu pervane bir başka pervaneye çarptığında çarpanın durup çarpılanın devam etmesi gerektiğini belirtirler. Daha sonra uzun süren tartışmaların ardından fikir kabul edilir. Her ülkenin üzerine dev bir pervane gelecek şekilde bir tasarım yapılmaya başlanır. Fakat bu tasarımı yaparken iki önemli faktöre dikkat edilmesi gerekmektedir:

• Bütün pervaneler bir şekilde hareketе geçebilmeli (hiç çalışmayacak olan bir pervane olmamalı)

• Bütün pervaneler **mümkün olan en yakın zamanda** bir kez bile olsa çalışmış olmalı

Sizden istenen bir program yazarak; konumları ve boyları verilen pervanelerin hepsinin dönmesinin mümkün olup olmadığını ve mümkünse ilk enerjinin hangi pervaneye ve hangi yönde verilmesi gerektiğini bulmanız.

Varsayımlar:

• Pervanelerin sayısı n'dir ($1 \le n \le 200$). Pervaneler 0'dan (n-1)'e kadar tamsayılar ile numaralandırılmıştır.

• Pervaneler iki boyutlu bir düzlemde merkezleri tamsayı koordinatlara gelecek şekilde yerleştirilmiştir. Pervanelerden herbiri, arasında tam 180 derece bulunan iki adet kola sahiptir. Kol uzunlukları da tamsayı ile belirtilecektir.

• Sizden istenilen olabilecek en az çarpma kullanarak bütün pervaneleri en az bir kez harekete geçirmenizdir.

• Bir pervane bir diğerine çarptığında durmaktadır. Dolayısıyla sistemde bir anda sadece bir pervane dönmektedir.

• Pervaneler kendi etrafında dönmeyi 60 adımda tamamlarlar (Her adımda 6 derece dönmektedir). Adımlar 0'dan 59'a kadar numaralandırılmıştır. 0 numaralı adım +x eksenidir ve adım numaraları saat yönünün tersinde artmaktadır.

• Bir pervane, bir adım içerisinde en fazla bir pervaneye çarpabilir. Yani bir pervanenin bir adımında birden fazla pervaneye çarpacağı bir girdi verilmeyecektir. Bir pervane, bir diğer pervaneye çarptığında başladığı adımı **tamamlamayacak**, adıma başladığı açıya dönüp duracaktır. Dolayısıyla pervaneler sabitken mutlaka 60 açıdan birinde bulunurlar. (örn: 29. adımından 30. adımına geçerken başka bir pervaneye çarpan pervane, 29. adımına geri döner ve durur.)

• Bir pervane diğer pervaneyle çarpıştığı anda iki pervanenin arasındaki iç açı (pervanelerden ilkinin merkez noktası, çarpma noktası ve pervanelerden ikincisinin merkez noktası arasındaki açı) 90 dereceden küçük eşitse çarpan pervane diğerini aynı yönde döndürecektir. Eğer bu açı 90 dereceden büyükse ters yönde döndürecektir.

• Herhangi bir pervanenin bir kolunun diğer bir pervanenin merkezine çarpabileceği bir girdi verilmeyecektir.

• Bir girdinin birden fazla en iyi çözümü bulunabilir, bu durumda en iyi çözümlerden herhangi birisini üretmeniz yeterli olacaktır.

• Bir girdinin çözümü bulunmayabilir. Bu durumda farklı bir çıktı üretilecektir.

©Ali Galip Bayrak - ODTÜ Bilgisayar Topluluğu

KURYE

Atasay ve Yiğit, Bilgisayar Topluluğu ve Verimlilik Topluluğu altında yürüttükleri büyük projelerde bazı anlaşmalara varmıştır. Rekabet ortamında birlikte çalışmanın gerektirdiği gizlilik problemini aşmak için de kurye görüşmelerinde kullanılacak bir yöntem geliştirmişlerdir. İki topluluğun da kampus içindeki noktalar arasında kullanabileceği kendine özel yollar vardır. İki kurye kendilerine özel iki topluluk merkezinden çıkıp belirli bir buluşma noktasında **aynı anda** buluşacaktır. Her bir kurye kendi merkezinden çıktıktan sonra önceden belirlenmiş bir zamanda buluşma noktasında olacak şekilde kendi topluluğuna ait yollardan ve noktalardan geçerek ilerleyecektir fakat hiçbir noktada (görülmemek amacı ile) beklemeyecektir. Buluşma zamanı **olabilecek en erken** zamanda olmalıdır. Sizin de Bilgisayar Topluluğu'na yardım etmek için 13 Nisan 2007 saat 23:59'a kadar, bu türde buluşmalar için kuryelerin izlemesi gereken yolları bulan bir program yazmanız gerekmektedir.

Varsayımlar:

- Kampus içindeki nokta sayısı *n*'dir ($1 \le n \le 100$). Noktalar 0'dan (n-1)'e kadar tamsayılar ile numaralandırılmıştır.
- Bir kurye bir birim zamanda iki noktayı birbirine bağlayan bir yol ilerleyebilmektedir.
- Yollar çift yönlüdür, yani *a* numaralı noktadan *b* numaralı noktaya yol varsa, *b*'den de *a*'ya yol vardır.
- Kuryeler noktalarda veya yollarda bekleyemezler.
- Bir kurye geçtiği noktadan veya yoldan tekrar geçebilir.
- Kuryeler belirli bir anda buluşma noktası dışındaki bir noktada bir araya gelseler de birbirlerini görmemiş gibi yollarına devam edeceklerdir.
- Bir girdinin birden fazla en iyi çözümü bulunabilir, bu durumda en iyi çözümlerden herhangi birisini üretmeniz yeterli olacaktır.
- Bir girdinin çözümü bulunmayabilir. Bu durumda farklı bir çıktı üretilecektir.

UZAYLI BÖCEKLER

Ünlü biyoloji profesörü Gözde Kaymaz yeryüzüne düşen bir meteorun kalıntılarında bulunan bir tür organizma kolonisini araştırmakla görevlendirilmiştir. Çalışması sırasında bu uzaylı böceklere "uzbik" adını vermiş, onların arasındaki ilginç iletişimi ve sosyal yapıyı keşfetmiştir. Uzbikler tek başına veya çeşitli sayılarda yan yana geldiklerinde yaşayabilmekte fakat bu sayı belirli bir sınırı aştığında aralarındaki denge bozulmakta ve birbirlerini yok etmektedirler. Bu yaratıklar iki boyutlu bir düzlem üzerinde belirli noktalarda bulunmaktadır. Uzbikler yerleri değiştirilmeden cam kutular içerisine alınacak ve bu kutuların içine de araştırmalar süresince onların yaşamasını sağlayacak madde ortamını düzenleyen aletler takılacaktır. Cam kutuların boyutlarının bir önemi bulunmamaktadır fakat her kutuya birer adet takılacak olan aletlerin maliyeti çok fazladır. Sizden istenilen bütün uzbikleri olabildiğince az sayıda cam kutu kullanarak kapatmak ve böylece araştırmanın maliyetini olabildiğince düşürmektir.

Varsayımlar:

- Düzlemdeki uzbik sayısı *n*'dir ($2 \le n \le 100$). Uzbikler 0'dan n-1'e kadar numaralandırılmıştır.
- Aynı cam kutu içerisinde yer alabilecek maksimum uzbik sayısı *k*'dir ($2 \le k \le 10$).
- Uzbiklerin her biri, iki adet tamsayı ile belirtilen x-y koordinatlarında bulunmaktadır.
- Cam kutular dikdörtgen şeklinde olacak ve kenarları koordinat düzleminin eksenlerine paralel olacaktır.
- Bir uzbik farklı iki kutunun içinde bulunamayacağından herhangi iki kutu kesişemez.
- Programınız, ürettiği çıktıda kullanılan kutu sayısının azlığına (olabilecek en uygun değere yakınlığına) göre puan alacaktır.

ÖZEL SORU
SICAK SOĞUK

Penguen Yoğurt ile Timsah Batlıcan; bir sabah ayaklarından birbirlerine zincirlenmiş bir halde karanlık bir odada uyanırlar. Uyandıklarında ışıklar yanar; yerde bir not ve bir harita bulurlar. Nota göre; kahramanlarımız ılıman, soğuk ve sıcak iklim odalarının bulunduğu bir labirenttedirler. Sıcak odalar Yoğurt'un, soğuk odalar Batlıcan'ın dayanamayacağı ısıdadır. Kahramanlarımızdan biri öldüğünde diğeri yaşam alanına geri gönderilecektir. 1000 oda değiştirme sonucunda labirentten biri çıkamadığı taktirde, her iki kahramanımız da yıkılan labirentte ölecektir. Yoğurtla Batlıcan anlaşarak, sırayla birinin istediği odaya geçmeye karar verirler. İlk seçimi Yoğurt yapacaktır. Sıra kendisine gelen kahraman; kuzey, güney, doğu veya batı odalarından birisini seçmek zorundadır. Sizden istenilen, kendi kahramanınızı yönlendirecek bir kod yazmanız.

Varsayımlar:

- Hikayemiz 2 oyuncu arasındaki bir oyun şeklinde oynanacaktır.
- Labirent, kenarları 1 birim uzunlukta kare şeklinde odalardan oluşan büyük bir dikdörtgen şeklindedir. Dikdörtgenin eni *M*, boyu *N*'dir ($2 \le N$, $M \le 30$).
- Labirentin bazı odaları kapalıdır, buralara girilemez.
- Oyuna Yoğurt başlar.
- Herhangi bir anda yanlış hamle yapan taraf oyunu kaybedecektir. Yanlış bir hamle: kapalı olan bir odaya girmeye çalışmak, basılması gereken karakterler dışında anlamsız karakterler basmak, labirentten dışarı adım atmak, vb. olabilir.
- Oyuncuların herbirinin 20 saniye süre sınırı vardır. Bu süre o oyuncunun tüm hamlelerinin toplamı içindir, hamle başına bir zaman kısıtı yoktur (isterseniz tek bir hamlede sürenizin büyük bir bölümünü kullanabilirsiniz).
- Programlarınız bizim yazacağımız bir hakem programı aracılığıyla oynatılacaktır. Oyuncuların hamlelerinin doğruluğu, oyunun bitirilmesi vs. işleri hakem programı tarafından yapılacaktır. Size gelen hamlelerin doğru olduğunu kabul edebilirsiniz.
- Programınız ilk olarak *sicaksoguk.gir* isimli dosyadan labirentin boyutlarını, oyuncuların başlangıç noktasını ve labirent bilgisini okumalıdır. Daha sonra standart girdiden (stdin) hangi oyuncuyu (Yoğurt/Batlıcan) oynatacağını belirten bir karakter okuyarak oyuna başlamalıdır. Oyun esnasında, sıra kendisinde ise standart çıktıya (stdout) hamlesini basmalı, sıra rakipte ise standart girdiden rakibin hamlesini okumalıdır.