**:ASSIGNMENT 1:**

**1. GİRİŞ / AMAÇ:**

Bu deneyde, okuma-yazma dosyaları, diziler, matrisler, özyineleme ve dinamik bellek ayırma dahil C dili hakkında bilgi edinmeniz beklenir. Temel C sözdizimi hakkında ön bilgi alınır.

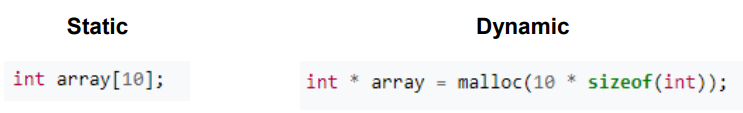
**2. ARKA PLAN BİLGİ**

**2.1. C DİLİ**

C, zorunlu bir prosedür dilidir. Hafızaya düşük seviyeli erişim sağlamak, makine talimatlarına verimli bir şekilde harita veren dil yapıları sağlamak ve minimum çalışma süresi desteği sağlamak için nispeten basit bir derleyici kullanılarak derlenecek şekilde tasarlanmıştır. Düşük seviyedeki yeteneklerine rağmen, dil, platformlar arası programlamayı teşvik etmek için tasarlanmıştır. Standart ve uyumlu bir yazılı C programı, kaynak kodunda çok az değişiklik olan çok çeşitli bilgisayar platformları ve işletim sistemleri için derlenebilir. Dil, gömülü mikrodenetleyicilerden süper bilgisayarlara kadar çok çeşitli platformlarda kullanılabilir hale geldi.

**2.2. DİZİLER**

C'deki dizi türleri geleneksel olarak derleme zamanında belirtilen sabit, statik bir boyuttadır. Bununla birlikte, standart kütüphanenin malloc fonksiyonu kullanılarak çalışma zamanında bir bellek bloğu tahsis etmek de mümkündür. C'nin dizilerin ve işaretçilerin birleşmesi, beyan edilen dizilerin ve bu dinamik olarak tahsis edilmiş simüle edilmiş dizilerin neredeyse birbirinin yerine geçebileceği anlamına gelir.



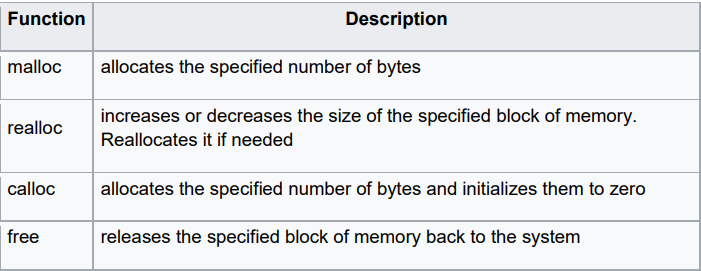
Dizilere her zaman işaretçiler aracılığıyla erişildiği için, dizi erişimleri genellikle temeldeki dizi boyutuna göre denetlenmez. Dolayısıyla, dizi sınırları ihlalleri dikkatsizce yazılmış kodlarda mümkündür ve oldukça yaygındır ve yasadışı bellek erişimi, veri bozulması, arabellek taşmaları ve çalışma zamanı istisnaları da dahil olmak üzere çeşitli yankılara yol açabilir. Sınır kontrolü isteniyorsa, manuel olarak yapılmalıdır.

**2.3. ÇOK BOYUTLU ARRAY**

C, çok boyutlu dizileri bildirmek için özel bir hüküm içermez, bunun yerine, aynı şeyi etkili bir şekilde gerçekleştiren dizilerin dizilerini bildirmek için tip sistemi içinde tekrarlamaya dayanır. Ortaya çıkan "çok boyutlu dizilim" in indeks değerleri, sıralı sırayla arttıkça düşünülebilir. Çok boyutlu diziler, matrisleri depolamak için sayısal algoritmalarda yaygın olarak kullanılmaktadır. C dizisinin yapısı bu özel göreve çok uygundur. Bununla birlikte, diziler yalnızca işaretçi olarak geçtiğinden, dizinin sınırları bilinen sabit değerler olmalı ya da bunları gerektiren herhangi bir alt yordama açıkça aktarılmalı ve dinamik olarak boyutlandırılmış diziler dizisine çift indeksleme kullanılarak erişilemez. (Bunun için bir geçici çözüm diziyi, sütunlara işaretçilerin ek bir "satır vektörü" ile ayırmaktır.).

**2.4. DİNAMİK BELLEK TAHSİSATI**

Bir tahsis talebini yerine getirme görevi, yeterli büyüklükteki kullanılmamış belleğin bir bloğunu bulmayı içerir. Bellek istekleri, yığın veya serbest depo adı verilen büyük bir bellek havuzundan bölümler ayrılarak sağlanır. Herhangi bir zamanda, yığının bazı bölümleri kullanımdadır, bazıları ise "serbest" (kullanılmamış) ve dolayısıyla gelecekteki tahsisler için kullanılabilir durumdadır. C dinamik bellek ayırma işlevleri stdlib.h başlığında tanımlanır.

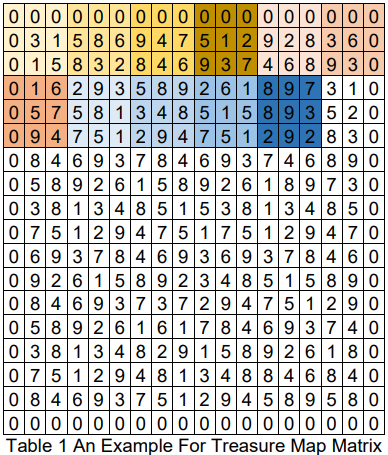


**2.5. ÖZYİNELEME**

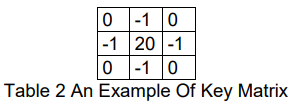
Özyineleme, maddelere benzer şekilde tekrarlama işlemidir. Programlama dillerinde, bir program aynı işlevin içinde bir işlevi çağırmanıza izin veriyorsa, o zaman bu fonksiyonun özyinelemesini çağırır. C programlama dili özyineyi, yani kendini çağırmak için bir işlevi destekler. Fakat özyinelemeyi kullanırken, programcıların fonksiyondan bir çıkış koşulu tanımlamaya dikkat etmeleri gerekir, aksi takdirde sonsuz bir döngüye girer.

**3. DENEYİM**

Bu deneyde, bir matris olarak tasarlanan bir hazine haritası içinde gizli hazinenin bulunması amaçlanmıştır. Bu amaçla, harita ve anahtar veriler dosyadan okunmalıdır. Daha sonra anahtar, kurallara bağlı olarak harita üzerinde hareket ettirilmelidir. Harita matrisinin ve anahtar matrisinin çarpılmasının sonucu, hazinenin yeri hakkında bir fikir verecektir. C programlama dilinde tüm bu işlemleri gerçekleştirmek için matris yapısını kullanmalısınız. Matrisleri saklamak için dinamik bellek ayırma kullanmalısınız.

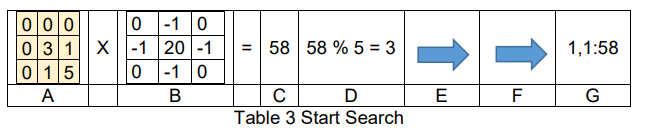


Hazine haritası, Tablo 1'de gösterildiği gibi bir matris olarak tasarlanmıştır. Harita matrisinin boyutu sabit değildir, ancak çalışma zamanında verilecek olan giriş dosyasına göre belirlenmelidir. Harita matrisinin her elemanı bir sayıdır. Kenarlara yerleştirilmiş matris elemanları sıfırlardır. Bu size haritanın sınırlarını gösterir.



Tablo 2'de gösterildiği gibi, anahtar, tek sayı olan (ör: 3,5,7) derece n olan bir kare matris olarak tasarlanmıştır. n, çalışma zamanında belirlenecek bir değişkendir. Anahtar matris kullanılarak, gizli hazine haritada bulunabilir. Hazineyi bulmak için haritadaki anahtar matrisi kaydırıyoruz. Anahtarın kapsadığı harita matrisinin parçası, haritadaki bir alt matrisi belirtir. Anahtar matrisi elde edilen alt matris ile çarparak, bir sonraki kaymanın yönünü buluruz. Yön, çarpım sonucunun mod beşi alınarak hesaplanır (sonuç% 5). Mod işleminin sonucu şu şekilde yorumlanır:

0.Hazine bulundu, 1.Yukarı, 2.Aşağı, 3.Sağa, 4.Sola



A: Alt matris

B: Anahtar Matris

C: A ve B matrislerinin çarpımı

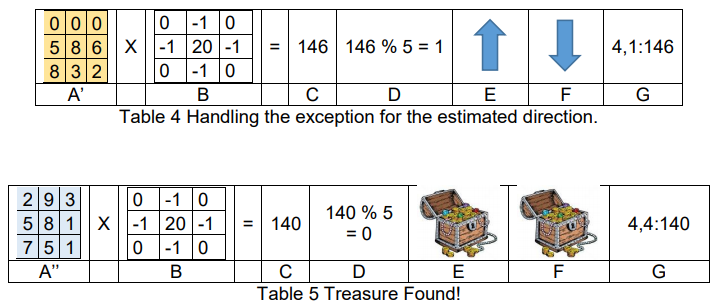
D: Modun kalan kısmı (5, C)

E: Tahmini yön

F: Son yön

G: Çıktı: Alt matrisin ve C'nin merkez hücre adresini yazınız.

Belirlenen yöne kaymanın bir yolu olmayan haritanın sınırına (örneğin en sağ veya en sol) anahtar matris yerleştirilirse, Çizelge 4'te gösterildiği gibi zıt yönde hareket etmelisiniz. Aslında, tahmini yön belirtilen istisna ile değiştirilebilir.



Mod sonucu sıfır ise, hazine Tablo 5'te gösterildiği gibi bulunur. Hazinenin yeri alt matrisin orta noktasıdır. Programınız her adımda alt matrisin orta noktalarını yazdırmalıdır.

**3.1. UYGULAMA**

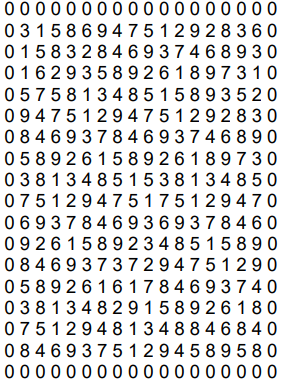
Derlenmiş yürütülebilir programın adı “buluntu” olmalıdır. Programınız giriş / çıkış dosya adlarını komut satırından okumalı, böylece aşağıdaki gibi çalıştırılacaktır: findtreasure [harita matrisinin boyutu] [anahtar matrisinin boyutu] [giriş dosya adı] [çıktı dosya adı]

örn .: ./findtreasure 18x18 3 mapmatrix.txt keymatrix.txt output.txt

Piazza sayfasında örnek giriş ve çıkışı görebilirsiniz. Program, DEV (dev.cs.hacettepe.edu.tr) UNIX makinelerinde çalışmalıdır. Bu yüzden, UNIX laboratuarından birinde derlendiğinden ve çalıştığından emin olun. Eğer derleyemez veya çalıştıramıyorsak, proje sıfır puan almayı riskler. Programı örnek dosyalarında (sağlanan) ve kendi girişlerinizde aynı mekanizmayı kullanarak sınamanız önerilir. Kendi çıktı ve örnek çıktınızı karşılaştırmalısınız. Çıktınız örneklemden farklıysa, proje de sıfır noktası almayı riskler.

**3.2. GİRİŞ / ÇIKIŞ FORMATI**

Harita matrisi dosyasında sayılar boşluklara ve satır sonlarına göre ayrılır. Hatalar için girişi kontrol etmeniz gerekmez. Örnek harita matrisi dosyası şu şekilde verilebilir:



Programınız çıktı dosyasına çıktılarını yazmalı, böylece çıktı dosyasındaki her satır alt-matrisin orta noktasını ve çarpmanın sonucunu içerecektir. Yukarıdaki giriş dosyası verilirse, çıktı dosyası aşağıdaki gibi olmalıdır:



**3.3. TASARIM BEKLENTİLER**

Çözümünüz için dinamik bellek ayırma kullanmalısınız. Spagetti veya statik diziler yazmak, tüm ödevinizi “kabul edilemez” hale getirebilir ve büyük (tam değilse de) dereceli kayba maruz bırakabilir.

**3.4. VALID PLATFORMLAR**

Kodunuz gcc sürüm 5.4.0'a karşı derlenecektir. Standart olmayan diğer kütüphanelerin / özelliklerin kullanılabilirliğini varsaymamalısınız. Farklı bir derleyici kullanırsanız, kodunuzun gcc'nin belirtilen sürümü ile derleme sorunu içermediğinden emin olmak sizin sorumluluğunuzdadır.

**4.2. RAPORLAR**

Programınızla ilgili bir rapor yazmalısınız. Raporda yer alması gereken konular:

 Problem tanımı

 Yöntemler ve çözüm

İmplemented Uygulanan ve uygulanmayan işlevler

**4.3. DERECELENDİRME**

Gönderdiğiniz kaynak kodunun Linux OS altında gcc derleyicisi ile derlendiğinden emin olun. Derleme veya çalışma zamanı hataları üreten uygulamalar

derecelendirilir.

You Gönderdiğiniz program beş farklı girişle test edilecektir. Her doğru çıktı dosyası için 10 puan, toplam 50 puan elde edebilirsiniz.

Source Kaynak kodunuz "Dinamik bellek ayırma, yineleme, çok boyutlu dizi ve yorum satırları" olmalıdır. Her bir madde için toplam 20 puan 5 puan kazanabilirsiniz.

Topic Her konu için 10 puan, rapor için toplam 30 puan elde edebilirsiniz.

Report Rapordan alabileceğiniz puanlar, yürütme ve kod gözden geçirme bölümlerinden aldığınız toplam puanlarla doğrudan ilişkilidir. Örneğin, ilk iki bölümden 35/70 puan alırsanız, rapor bölümünden en fazla 15 puan alabilirsiniz.

**SON SÖZLER:**

Programınızın çıktısı otomatik olarak derecelendirilecektir. Bu nedenle, çıktıdaki herhangi bir fark (en küçük fark bile) örnek çıktısından dolayı bir hataya neden olur ve siz de uygulamadan 0 alırsınız. % 100 doğru çalışmayan bir programın yanlış çalışan bir program olduğunu unutmayın.

Uzunluğundan bağımsız olarak, değişkenlerinize, sınıflarınıza ve işlevlerinize UNDERSTANDABLE adları kullanın.

READABLE SOURCE CODE bloğunu yaz.

Deneylerinizi göndermek için çevrimiçi gönderim sistemini kullanacaksınız. https://submit.cs.hacettepe.edu.tr/ Son Tarih: 04.11.2018 23:59:59. Başka bir gönderim yöntemi (e-posta veya benzeri) kabul edilmeyecektir.

Bu ödevle ilgili herhangi bir dosyayı e-posta ile göndermeyin.

Ödevin derecelenene kadar tüm çalışmalarınızı kaydedin.

Ödev orijinal, BİREYSEL çalışma olmalıdır. Yinelenen ya da çok benzer görevlerin ikisi de cezalandırılacak. Sorunun genel tartışmasına izin verilir, ancak cevapları, algoritmaları veya kaynak kodları PAYLA YAPMAYIN.

Sorularınızı kursun mono sayfası üzerinden sorabilir ve sayfada tartışılan her şeyin farkında olmanız gerekir.