Bu tasklarda, belirli bir konu üzerinde JavaScript kullanarak karmaşık bir uygulama geliştirmeniz beklenmektedir.

Önemli Not: Bu görevde herhangi bir yapay zeka aracı (ChatGPT, blackbox, gemini vb. gibi) kullanımı yasaktır. İstediğiniz gibi internette araştırma yapabilir ve bilmediğiniz kavramları araştırabilirsiniz. Gönderimlerinizi Githubtan yapacaksanız private repo açmanız gerekiyor. Tavsiyem zip haline getirerek mail olarak göndermenizdir.

Mail : ramazan.bakir.software@gmail.com

Teslim Süresi: Bu görevi tamamlamak için **salı akşam 19:30**'a kadar zamanınız var.

Başarılar dilerim!

Ramazan BAKIR.

TASK 1 :

"012345678" ifadesindeki her karakterin, bir (div1)'den diğerine (div2) belirli gecikmelerle hareket etmesini animasyonlu olarak yapmamız gerekiyor.

Animasyon sıralaması :

Tek sayı indexli karakterler için :

- Tek indeksli karakterler (1, 3, 5, 7, 9) belirli bir gecikme ile hareket eder: 4 saniye + (karakterin indeksi * 1 saniye).
- Örneğin : Index 3'teki karakter (yani 3), 3+4 = 7 saniye sonra hareket etmesi gerekiyor.

Çift sayı indexli karakterler için:

- Çift indeksli karakterler (0, 2, 4, 6, 8), tüm bitişik tek karakterlerin hareketi tamamlandıktan sonra, ek 1 saniye gecikmeyle hareket eder.
- Örneğin: Index 3'teki karakter (yani 3), 3+4 = 7 saniye sonra geldikten 1 saniye sonra sol tarafına 2 gelmesi gerekiyor.

Beklenen Animasyon Sırası ve Zamanlaması:

- "1" karakteri, 5 saniye sonra div2'ye taşınır (1*1 + 4 saniye)
- "0" karakteri, "1" karakterinin hareketi tamamlandıktan sonra div2'ye taşınır (6 saniye sonra)
- "3" karakteri, 7 saniye sonra div2'ye taşınır (3*1 + 4 saniye)
- "2" karakteri, "3" karakterinin hareketi tamamlandıktan sonra div2'ye taşınır (8 saniye sonra)

Kalan karakterler için benzer bir örüntü izleyin.

Gecikmelere ve hareket sırasına rağmen, tüm karakterler div2'de orijinal sıralarında ("012345678") görünmelidir.

Önemli bir nokta : en son 8 kalıyor o yüzden dikkat etmeniz gerekiyor. Yani 7 geldikten 1 saniye sonra 6 geliyor. 6'dan 1 saniye sonra da 8 gelmesi gerekiyor çünkü rakamlarda 9 yok buna dikkat ediniz.

Video ile anlatım için izleyebilirsiniz : https://ramazanbkr.com/task/task1/record1.webm

Size verilen html ve css kodları için indirebilirsiniz :
https://ramazanbkr.com/task/task1/html-css.txt

TASK 2:

Bir havayolu rezervasyon sisteminde uçuşları yönetmek için bir uygulama geliştireceksiniz. Bu sistemin, belirli bir havalimanından kalkış ve varış yapan uçuşların listesini çekmesi, yeni uçuş eklemesi ve mevcut uçuşları güncellemesi gerekmektedir. Bu işlemler aşağıdaki gibi asenkron olarak gerçekleştirilecektir:

1-) fetchFlights: Belirli bir havalimanı koduna göre uçuş listesini çeken bir fonksiyon yazın. Uçuş bilgileri bir API'dan çekilecektir. (Bu API'yı simüle eden bir fonksiyon oluşturabilirsiniz.)

2-) addFlight: Yeni bir uçuş ekleyen bir fonksiyon yazın. Bu işlem, belirli bir süre sonra tamamlanacaktır ve uçuş başarıyla eklendiğinde bir onay mesajı dönecektir.

3-) updateFlight: Mevcut bir uçuşu güncelleyen bir fonksiyon yazın. Bu işlem de asenkron olarak çalışacak ve uçuş başarıyla güncellendiğinde bir onay mesajı dönecektir.

4-) removeFlight: Belirli bir uçuşu kaldıran bir fonksiyon yazın. Bu işlem de asenkron olarak çalışacak ve uçuş başarıyla kaldırıldığında bir onay mesajı dönecektir.

5-) manageFlights: Yukarıdaki fonksiyonları kullanarak, uçuşları sırasıyla listeleyen, yeni bir uçuş ekleyen, mevcut bir uçuşu güncelleyen ve bir uçuşu kaldıran bir fonksiyon yazın. Bu işlemlerin tümü ardışık olarak gerçekleştirilecek ve her adımda konsola işlem durumu yazdırılacaktır.

Açıklamalar:

fetchFlights: Havalimanı koduna göre uçuşları filtreler ve 2 saniye sonra sonuç döner.

addFlight: Yeni bir uçuş ekler ve 1.5 saniye sonra onay mesajı döner.

updateFlight: Mevcut bir uçuşu günceller ve 1.5 saniye sonra onay mesajı döner.

removeFlight: Belirli bir uçuşu kaldırır ve 1.5 saniye sonra onay mesajı döner.

manageFlights: Bu fonksiyon, yukarıdaki işlemleri sırasıyla gerçekleştirir ve her adımda konsola işlem durumu yazdırır.

Örnek ekran resmi : https://ramazanbkr.com/task/task2/ss.png

Ek yardımcı html & css : https://ramazanbkr.com/task/task2/html-css.txt

Burada ki tasarımda ekleme düzenleme yapabilirsiniz. Ekstra yardımcı olması için verdim. Tasarım konusunda istediğiniz şekilde yapabilirsiniz.

TASK 3:

Bu görevde, Bootstrap'in grid yapısını ve sınıflarını kullanarak belirli bir resmi kullanarak bir piramit yapısı oluşturacaksınız. Herhangi bir JavaScript kullanmadan sadece HTML ve CSS kullanarak bu yapıyı oluşturmalısınız.

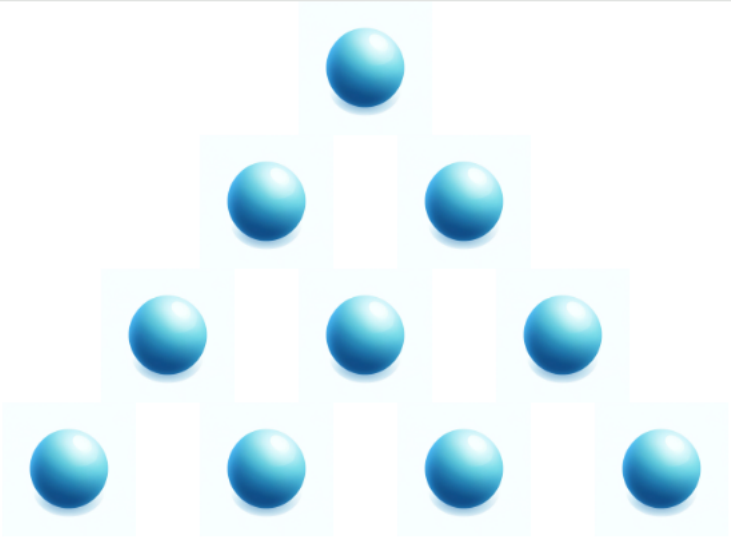
Bootstrap grid sistemi hakkında bilgi edinmek için Bootstrap dokümantasyonunu inceleyebilirsiniz.

Başlangıç için verilen HTML & CSS :

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
  <style>
    .col {  padding: 10px;}
    img:not(.fluid-img) {width:50px; height:auto;}
  </style>
</head>
<body>

<div class="container">

</div>

</body>
</html>
```

İpucu : Css üzerinde bir değişiklik yapmanız gerekmiyor. Tamamen HTML'de bootstrap kullanarak yapmanız gerekiyor.

Sonuç olarak alttaki görselin oluşması gerekiyor :



TASK 4 :

Bu görevde, verilen HTML yapısını ve CSS kodlarını kullanarak belirli bir animasyonlu görseli yeniden oluşturmanız gerekmektedir. Verilen görseli ve animasyonu mümkün olduğunca yakın bir şekilde yapmanız gerekmektedir.

HTML için verilen :

```
<html>
    <body>
        <div class="main"></div>
    </body>
</html>
```

CSS için verilen :

```
.main {

}
```

Sonuç video : https://www.ramazanbkr.com/task/animasyon/video.mov

İpucu : burada yer alan metinleri css'de yazabilirsiniz. Hatırlamak gerekiyor. Görselde arka planda yazan texttir herhangi bir görsel değildir.

TASK 5 :

- Bir şirket, çeşitli projeler üzerinde çalışan çalışanların verimliliğini artırmak için bir performans izleme sistemi geliştirmektedir. Bu sistem, çalışanların projelere ne kadar zaman harcadığını ve hangi görevleri tamamladığını takip eder. Şirketin ihtiyaçlarına uygun bir fonksiyon yazmanız gerekmektedir.
- Çalışanlar ve projeler, belirli bir yapıdaki nesneler olarak temsil edilecektir.

- Her çalışanın belirli görevleri ve her görevin belirli bir süre harcama süresi olacaktır.
- Asenkron bir fonksiyon yazın (calculateTotalTimeSpent), bu fonksiyon bir çalışanın belirli bir projeye harcadığı toplam süreyi hesaplamalıdır.
- Bu fonksiyon, çalışanın görevlerinin sürelerini toplamalıdır ve her görev, bir Promise ile asenkron olarak süreyi döndürecektir.
- Fonksiyon, aynı görev için tekrar hesaplama yapmamak amacıyla memoization kullanmalıdır.
- Ayrıca, fonksiyon recursive olmalıdır; yani, bir görev alt görevler içerebilir ve her alt görev için de süreler hesaplanmalıdır.

Optimizasyon için Memoization kullanarak gereksiz hesaplamaları engelleyebilirsiniz. Promisified yapı ile asenkron işlemleri yönetin ve fonksiyonun doğru sonuçlar döndürmesini sağlayın.

İpucu : Her görevin süresi bir Promise içinde dönecektir. Bu nedenle, Promise.all ve async/await kullanarak asenkron işlemleri yönetin. Memoization kullanarak daha önce hesaplanmış görevlerin sürelerini saklayın ve tekrar hesaplamayın. Recursive bir yapı kullanarak alt görevlerin sürelerini de hesaplayın.