**Adım Adım Case Study Cevaplarının Açıklamaları**

**Soru-1**

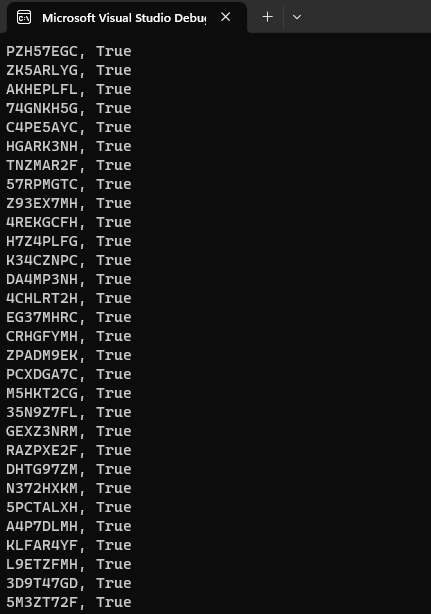
Burada **luhn algoritması**nı kullandım. Basitçe kodu oluşturmak için elimizde bulunan dizi elemanlarına belli sayısal değerler verip, bunların 7 karakterini rastgele seçip 8. Karakteri hesaplama üzerine kurulu. Öyle bir 8. Karakter seçilmeli ki toplam değer 10’a tam bölünsün zaten algoritmanın bir diğer adı da mod 10 algoritması.

1. Kodun içermesi gereken karakter dizisini tanımladım, içinde barındırdığı her bir karakterin değeri de kendisinin index değeri olacak şekilde ilgili algoritmaya göre kodu ürettim.
2. Üretmiş olduğum kodları bir hash set içerisine ekledim. Buradaki maksadım kodların **unique** üretilmesini garantilemek için üretme aşamasında bir kodu 2 kez üretme ihtimalini engellemekti.
3. Üretilen kodların doğruluğunu da algoritma sayesinde herhangi bir yere sormadan belli şartlara göre yapılan toplam işlemini tekrarlayarak 10’a tam bölünüp bölünmemesinden kontrol edebildim.

Kısaca kodları random kullanarak ürettim ki, bir örüntü yakalanıp üretme algoritması en azından kolay tahmin edilemesin.

Unique’liği sağlamak adına kod üretim aşamasında bir hashset’e ilgili kodu atıp atmadığıma baktım. Ve kodların üretimi ve doğrulanması konusunda da luhn algoritmasını kullandım.

Soru-1 Output’u: (solda üretilen kod, sağda kontrol değeri)



**Soru -2**

1. Saas üzerinden dönen response’u uygun bir modele deserialize ettim.
2. Deserialize sonrası elimdeki ocrItemList’in elemanlarını koordinat bilgilerinin y eksen değerlerinin ortalama toplamları üzerinden sıraladım. (veriyi satırlara bölmek maksatlı yaptığım işin ilk adımı)
3. Response’u ilk incelediğimde gözle gördüğüm bir örüntü vardı (örn: ilk ordinat değerleri 121,123 ise bir sonraki 143,144 değerlerine sonra ki ise 164-169 gibi değerlere sahipti.) Bu durum farklı fişlerde vs. değişeceğinden sıralamış olduğum liste elemanlarının sadece bir ordinat değerini seçip bu elemanlarında birbirleri arasındaki farkın ortalamasını bularak ortalama bir satır değişimi için ordinat ekseninde ne kadar hareket etmem gerektiğini hesapladım. Bu hesabı yaparken ordinat verilerini başta 10 a bölmemdeki maksat veriyi gürültüden kurtarmaktı değer 121,123 ise ben onu 12-12 şeklinde hesapladım ve distinct alarak da tekrar eden değerleri uçurdum.
4. Bir foreach döngüsü ile her bir item için hesapladığım ortalama ordinat değerim ile 3.adımda hesapladığım avarage değerlerini karşılaştırarak veriyi satırlara ayırdım. Ve bu ayırdığım verileri tuple tutan bir liste içerisine ekledim.
5. Daha sonra bu liste üzerinde son kez bir foreach döngüsü kurarak ordinat ekseninde sıraladığım kelimeleri bir kere de x ekseni doğrultusunda sıraladım ve bir stringbuilder değişkenine ekledim.
6. Son olarak stringbuilder değişkeni içerisindeki veriyi ilgili bir .txt dosyasına yazdırdım.

NOT: İlk veride tüm data satır sonları(\n) ile beraber geliyordu, bu veri de doğrudan bir txt dosyasına yazılarak bu iş yapılabilir. Ancak bu ocr tarafında optional olduğundan bunu görmezden geldim. Bu yüzden hesaplamalarımda ilk veriyi es geçerek devam ettim.

Soru-2’nin Output’u :

