Define Use Case

1. İlk olarak program açıldığında 8 adet boş semester objesi açılacak ve bunlara identifier atanacak.
2. createCourse fonksiyonuyla döngü halinde kurslar okunup kurs objeleri açılacak ve bu objelere belli özellikler sırasıyla tanımlanacak. Bundan sonra ise bu kurslar, semester objelerinin içindeki listelere okunurken girilecek.
3. Dönemlere kurslar tanımlandıktan sonra ilk dönemden itibaren random oluşturma işlemleri başlayacak. İlk dönemdeki öğrenci sayısı belli bi aralıkta random olarak belirlendikten sonra o dönemde okula yeni başlamış gibi öğrenci kayıtlarına başlanacak ve programın simülasyon kısmı başlamış olacak.
4. Belli sayıda olan öğrenciler o sayıya gelene dek belli bi döngü içinde oluşturulmaya başlanacak. Random isimler alacak, numaralar belli bir sırada artacak ve her öğrenci oluşturulurken belli dersler alacak. (Bu öğrenciler dersleri aldığı gibi kalıp kalmadığı gözükecek mi bilmiyorum). Fakat öğrenciler bana göre dersleri alırken kalamaz çünkü öğretmen onayından geçmesi lazım. Yani her öğrencinin normal şekilde diğer döneme geçerken önceki derslerden kalıp kalmadığı belli olacak.
5. Üstten alma olmayacak. Her öğrenciye o dönemin dersleri verilecek, fakat öğrenciler random sayıda olacağı için bazı öğrenciler kontenjan bulamayabilir (!). Bunu kontrol edecek bölüm kurs objesinin kendisi olacak ve her öğrenci kayıt olurken o kursun kontenjanı kontrol edileceği için bazı öğrencilere kurs verilmeyecek. Karşılaşacağımız ilk hata bu olacak ve bu hata öğrenciye kaydolması gereken, output logunda görmemiz gereken bir hata. (ACABA BU OUTPUTLAR ÖĞRENCİYİ OLUŞTURDUĞUMUZDA MI EMPTY OLARAK OLUŞTURULUP BELLİ AŞAMALARDA DOLDURULACAK, YOKSA ÖĞRENCİ OBJESİNDE PROGRAM SONUNA KADAR TUTULACAK MI?)
6. Diyelim ki OUTPUT’lar öğrenci oluşturulduğunda empty olarak oluşturulacak. Bu outputlara yazılacak ilk şey bu öğrencilerin kişisel bilgileri olacak ki biz bunlara sahibiz. O zaman ilk alınan hatalar yazılacak, programın en sonunda da her öğrencinin içinde gezebileceğimiz bir döngüyle transkripti yazacağız. Yani bu aldığımız ilk hata öğrenci output’una yazılacak, ayrıca Department outputta da gözükeceği için ERROR gibi bir class’a sahip olup bunları kurslara göre oluşturacağız daha sonra bunlar için hata türleri olacak ve bunlar ana ERROR class’ından inherit edecek. QUOTA error, veya advisor vermedi gibi vs. Ayrıca bunların tek tek hali öğrencinin kendi outputuna da not alınacak.
7. Daha sonra ilk oluşturulan öğrencileri 8. döneme kadar aynı şekilde kaydetmeye devam edicez. Bir dönemden diğerine geçerken önceki dönemdeki derslerinden kalıp kalmadıkları belli olacak ve bunlara göre derslere kayıt olabilecekler. Ayrıca kredi ve diğer istenilenleri her ders kayıt bölümünde hoca kontrol edecek.
8. Son döneme gelene dek kurs ve öğrenciler arasında ilişki hep devam edecek. Öğrencilerin başka döneme geçerken notları kayıt edilecek, çan eğrisine göre harf notları verilecek.
9. En son program bittiğinde öğrenci outputunda sırasıyla öğrenci bilgileri, error kayıtları ve son olarak dönemlerde sırasıyla GPA harf notları gözükecek.