PROJE 3

Egemen Çakır

Kocaeli Bilgisayar Mühendisliği
Email: cakiregemen0@gmail.com

I. GIRIS

Bu projede, ögrencilerin isletim sistemlerinin temel kavramlarını (thread yönetimi ve senkronizasyon mekanizmalarını) anlamaları ve bu kavramları bir simülasyon uzerinde uygulamaları hedeflenmektedir.Projede bir restoran yonetim sistemi gerceklestirmeniz beklenmektedir. Bu proje bir restoranın gunluk isleyisini simule eden bir multithread uygulamasıdır. Buradaki amac gercek zamanlı bir ortamda thread senkronizasyonu ve kaynak yonetimi konseptlerini uygulamaktır.Restoranda, musteri grupları, garsonlar, ascılar ve bir kasa elemanı bulunmaktadır. Her bir rol, farklı gorevleri ve sorumlulukları olan ayrı bir thread olarak islenecektir.Ayrıca projenin kuralları:

- SUREC:
  - Müsterilerin sıraya girmesi.
  - Garsonların siparis alması.
  - Ascıların siparisleri hazırlaması.
  - Musterilerin siparisleri odemesi.
- THREADLER:
  - Her musteri, restorana giris yaparken bir thread olarak olusturulur.
  - Her garson, musterilere hizmet vermek icin ayrı bir thread'de calısır.
  - Her ascı, yemek hazırlama surecini yonetmek icin kendi thread'inde calısır.
  - Kasa islemleri, odeme alma ve hesap kapatma (masanın uygun duruma gelmesi) islemleri icin ayrı bir thread'de gerceklestirilir.
- SENKRONIZASYON:
  - Musteri thread'leri arasında masa secimi ve oturma sırası icin senkronizasyon gerekir.
  - Garson thread'leri, siparisleri alırken ve islerken diger garsonlarla koordineli calısmalıdır. Aynı masaya birden fazla garsonun bakması onlenmelidir.
  - Ascı thread'leri, sınırlı sayıda ocakta yemek hazırlarken birbirleriyle uyum icinde olmalıdır. Aynı siparisi birden fazla ascının hazırlaması engellenmelidir.
  - Kasa thread'i, her seferinde yalnızca bir siparisin ¨odemesini isleyebilir.

Ayrıca bu projede ele alacagımız iki problem vardır.Bunlar :

- 1. PROBLEM:
  - Uygulama calıstırıldıgında musterilerin gelme sırası belirlenerek projenin ozellikleri ve sureler dikkate alınarak simulasyon gerceklestirilir.
  - Islem sırası geldiginde kullanıcı onayıyla islemler yurutulur.
  - Her adımda kullanıcıdan bir sonraki asama icin onay beklenir
  - Simulasyon baslangıcında musteri senaryosu belirlenir. Bu asamada, zaman sınırlamalarına baglı kalmaksızın musterilerin oncelik duzeylerinin belirlenmesi hedeflenmektedir.
  - Restoranda baslangıcta 6 masa, 3 garson, 2 ascı ve 1 kasa bulunmaktadır.
  - Aynı anda en fazla 10 musteri gelebilir.
- 2. PROBLEM:
  - Program baslatıldıgında belirli bir sure tanımlanır. Bu sure icerisinde musteri akısı hesaplanarak restoranın kaynakları belirlenerek maksimum musteri sayısına hizmet verme stratejisi gelistirilir.
  - Musteri sayısı iki sekilde belirlenebilmelidir:
    - Sabit akıs modeli: Toplam sure, musteri gelme aralıgı uygulamada belirlenmelidir.
    - Rastgele akıs modeli: Belirlenen sure icerisinde musterilerin gelisi rastgele aralıklarla ve sayıda olacak sekilde belirlenir.
  - Bir problem icin bulunan cozum senaryolarını ve en iyi cozumu gostermeniz beklenmektedir.

Projenin ozellikleri ise :

- Her masaya bir musteri oturmaktadır.
- Masalar dolu oldugunda yeni gelen musteri bir bekleme listesine alınır. Bos masa oldugunda musteri sırasına gore uygun musteri masaya yerlesir.
- Proje kapsamında musteriler iki farklı kategoride degerlendirilebilir: Normal musteriler ve oncelikli musteriler (65 yas ve uzeri).Oncelikli musteriler restorana geldiklerinde,normal musterilerin onune gecmelidir. Bu durum masalara oturma sırasında oncelikli musteri avantajı saglamalıdır.
- Bos masa varken garsonlar bekleme durumundadır. Masa doldukca garsonlar sırayla musterilerden siparis alır. Aynı anda gelen olursa rastgele siparis alımı yapabilir.
- Her garson aynı anda sadece bir musterinin siparisini alabilir.
- Ascılar siparis gelene kadar bekler. Siparis geldikten sonra yemek yapmaya baslarlar. Yemek hazır oldukca sıradaki siparisleri hazırlamaya baslarlar.
- Siparis alındıgında, her bir ascı aynı anda en fazla 2 yemek hazırlayabilir.

- Siparis hazır olduktan sonra musteriler yemege baslar. Musteriler yemek yerken, kasa islemleri icin beklemelidir.
- Yemek biten masanın odemesi kasa tarafından alınır.
- Kasa her seferinde sadece 1 siparisin odemesini gerceklestirebilir.
- Odeme tamamlandıktan sonra ilgili masa uygun duruma gelir.

Proje sayesinde java yazalım dilini kullanarak threadleri kullanmayı ve threadlerin senkronizasyon ve detaylarını ogrendim.Bu proje thread ve senkronize işlemlerini uygulama ve ögrenme acısından oldukca önemlidir.

II. Adımlar

Adım 1

Oncelikle giris sayfamı yapmak için main class olusturdum ve JFrame den extends ettim.Bu sayfada iki problemi ayrı ayrı ele almak için iki buton koydum ve butonlara actionListener ekleyip kullanıcının hangi probleme gitmesi gerekiyorsa o probleme ait sayfanın acılması için methodlar ekledim.Problemler için Problem1Page ve Problem2Page classlarını olusturup JFramden extends ettim.

Adım 2

Bu adımda birinci problemi yapmak için oncelikle kasiyer , ascı , garson , masa ve musteri adında classlar olusturdum ve Thread abstract sınıfından extends ettim ve override etmem gereken public void run() methodunu override ettim.Bu sınıfın olmasının sebebi threadi baslattıgımızda bu method calısır yani bir threadin bir şey yapmasını istiyorsak bu sınıfın içine yazarız.Bu sınıfları doldurmadan once Problem1Page sınıfını düzenlemeye basladım.Sırayla yaptıgım seyler:

- Sayfanın üstüne 4 tane buton koydum ve bazı durumları gerceklestirmeleri için actionListener ekledim.Bu butonlar backBtn , settingBtn , startBtn ve beforeBtn:
  - backBtn : Bu buton ile giris sayfasını donmek için kullandım.
  - settingBtn : Bu butonda bazı sabit deger ve süreler vardı ve bizden bunların degiştirilebilir olması istenmisti.Bende jDialog ile mecvut pencerenin üstüne pencere acıp bu degerleri güncelleme imkanı verdim.
  - beforeBtn: Bu buton ile simülasyon işlemi yapmadan once belli senaryolar belirlememiz gereklidir.Bu buton ile kullanıcıdan gelecek müsteri sayısını alıyorum ve ArrayList¡ArryList¡Integer¿¿ senaryoList arrayListeme aktardım.
  - startBtn: Bu buton ile simülasyon işlemini baslattım fakat kullanıcı bir tane bile senaryo secmediyse JOptionPanel.showMessageDialog() ile kullanıcıya en az bir tane senaryo secmesini soylüyorum , eger tüm kosullar ugunsa simüle için yazdıgım kodları çalıstırıp problem biri gerceklestirmeye calıstım.
- Sayfanın sag tarafına üc adet buton koydum bunlar simüle anında garson , ascı ve kasiyer ile ilgili bazı bilgileri goruntüleyebilecegimiz sayfalar ve bu bilgileri tablolarda goruntüleyebiliyoruz.
- Son olarak kalan yerlere üc adet panel yerlestirdim.Bunun sebebi asci , kasiyer ve masaların simüle anında farklı yerlerde olmasını istedigim için boyle bir düzen yapısını sectim.

Genel olarak Problem1Page sayfasının genel düzeninin olusturduktan sonra simüle kodunu yazmadan bu sınıfta kullanmak için olusturdugum sınıfları doldurmaya basladım.

Adım 3

Sırasıyla musteri , masa , garson ,ascı ve kasiyer sınıfları :

- musteri classı : constructor ile integer id numarası , Masa türünde emptyMasa ve integer türünde iki adet x ve y kordinat bilgileri alıyor.Yemek yeme metodu ve masa secme metotlarını yazdım ve run methodunda cagırdım.Yazdıgım ve doldurdugum methodlar:
  - masaSecim methodu: Bu fonksiyonda Masa.remove() methodunu cagırıyorum ve eger bos bir masa varsa o masayı donduruyor ve return olarak true donuyor eger musterinin bekleme suresi gecen bir durum olmus ve hayla bir masaya oturmamıssa Masa.remove() methodu null deger doner ve bu durumda return false doner.
  - yemekYeme methodu:Bu methodda oncellikle yemek yeme süresi kadar bekler ve daha sonra static olarak tanımlı yemegiBitenler arrayList'ine ekleme yapar ve Kasiyer sınıfında wait() ile beklyen Kasiyer threadlerini notifyAll() ile uyandırıyorum.
  - run methodu: Burada oncelikle masaSecim methodunu cagırıyorum ve donen degeri alıyorum eger donen deger false ise return diyerek threadi bitiriyorum aksi halde yemekYeme methodunu cagırıyorum.
- masa classı : constructor ile integer id numarası ve integer türünde iki adet x ve y kordinat bilgileri alıyor.masaRemove metodu ve masaAdd metotlarını yazdım.Yazdıgım methodlar:
  - masaAdd methodu: Parametre olarak aldıgı masa degişkenini alır ve masaList arrayListine ekler ayrıca belli bir duruma gore Problem1Page.lockRemove ile senkronize olmus yapıda bekleyen yapıları uyandırmak için notifyAll() methodunu cagırıyorum.
  - masaRemove methodu:Bu methodda oncellikle masaList arrayListini eleman sayısını alıyorum ve eger 0 ise musteriyi bekleme suresi kadar bekletiyorum eger bu süreden once uyandırılırsa bos masayı kapmak için ugresır kaparsa retrun ile doner eger beklem suresinde once uyanmaz ise return null doner.Ayrıca masayı aldıktan sonra return ile donmeden once siparisiAlınmayanlar arrayListine ekleme yapar ve uyuyan garsonlar için lock u alır ve notifyAll() methodunu calıstırır.
- garson classı : constructor ile integer id numarası ve integer türünde iki adet x ve y kordinat bilgileri alıyor.Siparis alma metodunu yazdım.Yazdıgım method:

  - siparisAlma methodu:Oncelikle siparisialinmayanlar listesinin eleman sayısını alıyorum ve eger 0 ise wait durumuna geciriyorum.Eger 0 degilse siparis alma suresi kadar bekleyip siaprisiAlinmayan listsinden bir kişi alıp siparisiAlinanalar listesine ekleme yapıyor ve uyuyan ascıları uyandırıyorum.
- ascı classı : constructor ile integer id numarası ve integer türünde iki adet x ve y kordinat bilgileri alıyor.Yemek hazirlama metodunu yazdım.Yazdıgım method:
  - yemekHazirlama methodu:Oncelikle siparisialinanlar listesinin eleman sayısını alıyorum ve eger 0 ise wait durumuna geciriyorum.Eger 0 degilse yemek hazirlama suresi kadar bekleyip duruma gore 1 veya 2 kişini yemegini hazırlamak üzere siaprisiAlinanlar listsinden kişi/kişileri alıp siparisiHazirlananlar listesine ekleme yapıyor ve uyuyan kasiyerleri uyandırıyorum.
- akasiyer classı : constructor ile integer id numarası ve integer türünde iki adet x ve y kordinat bilgileri alıyor.odeme metodunu yazdım.Yazdıgım method:
  - odeme methodu:Oncelikle siparisiHazirlananlar listesinin eleman sayısını alıyorum ve eger 0 ise wait durumuna geciriyorum.Eger 0 degilse odeme suresi kadar bekleyip duruma gore 1 kişinin yemegin ödemesini almak üzere siaprisiHazirlananlar listsinden alıp odemeYapanlar listesine ekleme yapıyor ve Masa.masaAdd() methodunu cagırıyorum.Ayrıca hiç bir müsteri kalmayınca ilk senarya için cagırdıgım problem1Run() methodunu eger uygulanacak bir senerya varsa cagırıp sırada hangi senaryo varsa onu simüle ettiriyorum.

boylelikle yan sınıflarımı tamamlamıs oldum.Problem 1 için simüle etmek için kullandıgım problem1Run () methodunu yazdım . Bu method:

- problem1Run methodu : Bu fonksiyonda oncelikle simüle için tanımladıgım arrayListeler , paneller ve degişkenleri resetliyorum.Daha sonra Useful sınıdında yazdıgım generateRow ve generateCol methodları ile masa , kasiyer ve asci panellerinin gridLayod unun row ve col degerlerini hesaplayıp ayarlıyorum.Daha sonra yine Useful sınıfında yazdıgım her bir thread sınıfı için operaton methodu ile oncelikle sabit degerleri kadar thread üretimi yapıp daha sonra .start() methodu ile tüm threadleri baslatıyorum ve boylelikle simüle işlemi gerçekleşmiş oluyor.

Bu fonksiyonuda yazdıktan sonra bu fonkisyonu hem start methodunda ilk senaryo için hemde kasiyer sınıfında eger hiç bir müsteri kalmadıysa ve mevcut senaryo kaldıysa bu methodu cagırıp simüle işlemini gerçekleştiriyorum.

Adım 4

Bu adımda ikinci probleme gectim.Oncelikle simüle işlemi olmayacagı için sadece Problem2Page sınıfına iki adet buton koydum bunları koyma amacım problemi sabit akıs mı yoksa rastgele akıs şeklinde mi gerçekleştirecegini secmesi için koydum.Sabit akıs butonuna bastıgında kullanıcıdan oncelikle zaman aralık bilgisini daha sonra sabit müsteri gelme zaman araslıgını alıyorum ve en son olarak her aralıkta sabit gelecek musteri sayısını alıyorum ve bu bilgiler ile olusturdugum sabitAkısStrateji methodunu calıstırıp max kazancı ve ideal masa, garson gibi degerleri JOptionPanel ile ekranda gosteriyorum.sabitAkısStrateji methodu:

- sabitAkısStrateji methodu: Bu method kullanıcının giridigi bilgiler ile oncelikle toplam gelecek musteri sayısını elde eder.Daha sonraki işlemleri son musteri üzerinden uygular burada oncelikle top süreye musterilerin kac defa gelecegi hesap edilir daha sonra son musterinin bekleme suresi elde edilip bu süreyi bir kisinin masadan kalması için gerekli zamana bolundugnda beklemesi geek garson sırası elde edilir ve bu degerler ile masa , garson ,ascı degerleri bulunup aralrından en fazla kazanc saglayanı elde edilir.

Rastgele akıs ile cozmek isterse bu sefer kullanıcıdan sadece toplam sure alınır ve rastgeAkısStrateji methodu ile bu süreyi aşmayacak şekilde Random sınıfı kullanılarak süreler ve gelecek süreler üretilip daha sonra sabit akısta oldugu gibi son müsteri üzerinde max kazanc ve olasıklar blunup en ideali secilir ve ekrana gosterilir.Boylelikle ikinci problemide bitimis olup genel olrak projeyi bitirdim.

III. Özet

Bu projenin amacı, threadleri senkronize bir biçimde kullanarak iki problemide cozmekti. Bu projeyi yaparken threadleri kullanmayı ve senkronize işelmlerini ogrenmeyi ve uygulamayı ogrendim.

IV. Sonuç

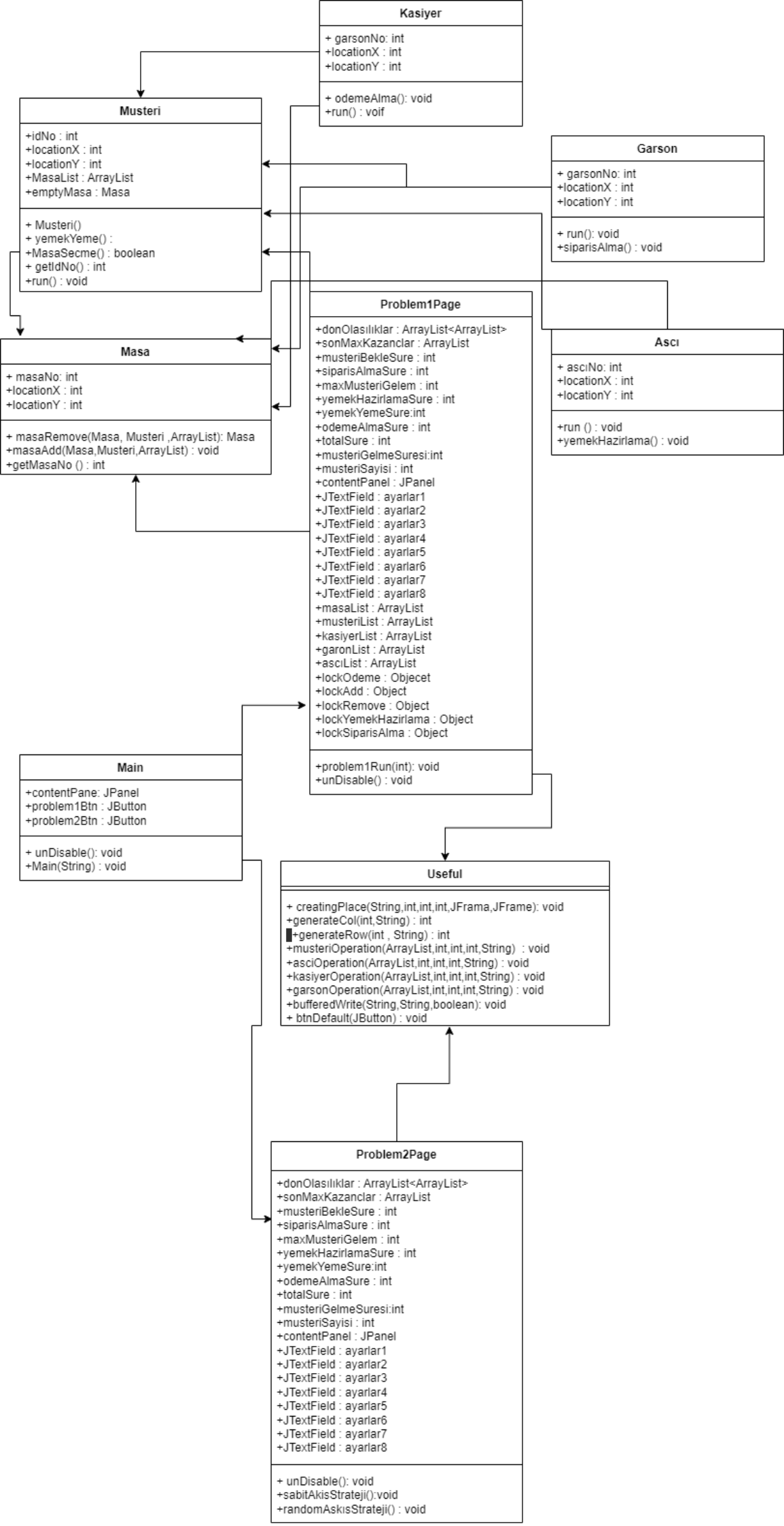
Tüm kontroller yapıldıktan sonra artık iki problemide buyuk cogunlukta cozmus olduk.Ilk problemde baslangıc degerlerine gore simule edebiliyor ve degerleri .txt uzantılı dosyaya yazabildim ve sabit degerleri degiştirebildim.Ikinci Problemde her iki akıs modellerine gore kullanıcıdan aldıgım degerler dogrultusunda max. kazancı elde edebildim.Fakat projeyi yaparken bazı problemlerle karşılaştım bunlar:

- Simule etme asamasında thread senkronu konusunda hatalar ve senkron olmama gibi durumlarla karsılastım.Bu sorunları tekrar tekrar kontrol ve test ile cozdum.

Genel olarak projenin sonuçları böyleydi.

V. Kaynakca

- https://umitsamimi.medium.com/java-threads-980347735de1
- https://emrahmete.wordpress.com/2011/10/06/javada-thread-yapisi-ve-kullanimi-hakkinda-ipuclari/
- https://medium.com/@tarikkaan1koc/javada-paralel-ve-seri-programlama-temel-farklar-ve-senkronizasyon-araçları-f1f38c1dcd90


Kasiyer
+ garsonNo: int
+locationX : int
+locationY : int
+ odemeAlma(): void
+run() : voif
Musteri
+idNo : int
+locationX : int
+locationY : int
+MasaList : ArrayList
+emptyMasa : Masa
+ Musteri()
+ yemekYeme() :
+MasaSecme() : boolean
+ getIdNo() : int
+run() : void
Garson
+ garsonNo: int
+locationX : int
+locationY : int
+ run(): void
+siparisAlma() : void
Masa
+ masaNo: int
+locationX : int
+locationY : int
+ masaRemove(Masa, Musteri ,ArrayList): Masa
+masaAdd(Masa,Musteri,ArrayList) : void
+getMasaNo () : int
Problem1Page
+donOlasılıklar : ArrayList<ArrayList>
+sonMaxKazanclar : ArrayList
+musteriBekleSure : int
+siparisAlmaSure : int
+maxMusteriGelem : int
+yemekHazirlamaSure : int
+yemekYemeSure:int
+odemeAlmaSure : int
+totalSure : int
+musteriGelmeSuresi:int
+musteriSayisi : int
+contentPanel : JPanel
+JTextField : ayarlar1
+JTextField : ayarlar2
+JTextField : ayarlar3
+JTextField : ayarlar4
+JTextField : ayarlar5
+JTextField : ayarlar6
+JTextField : ayarlar7
+JTextField : ayarlar8
+masaList : ArrayList
+musteriList : ArrayList
+kasiyerList : ArrayList
+garonList : ArrayList
+ascıList : ArrayList
+lockOdeme : Objecet
+lockAdd : Object
+lockRemove : Object
+lockYemekHazirlama : Object
+lockSiparisAlma : Object
+problem1Run(int): void
+unDisable() : void
Ascı
+ ascıNo: int
+locationX : int
+locationY : int
+run () : void
+yemekHazirlama() : void
Main
+contentPane: JPanel
+problem1Btn : JButton
+problem2Btn : JButton
+ unDisable(): void
+Main(String) : void
Useful
+ creatingPlace(String,int,int,int,JFrama,JFrame): void
+generateCol(int,String) : int
+generateRow(int , String) : int
+musteriOperation(ArrayList,int,int,int,String) : void
+asciOperation(ArrayList,int,int,int,String) : void
+kasiyerOperation(ArrayList,int,int,int,String) : void
+garsonOperation(ArrayList,int,int,int,String) : void
+bufferedWrite(String,String,boolean): void
+ btnDefault(JButton) : void
Problem2Page
+donOlasılıklar : ArrayList<ArrayList>
+sonMaxKazanclar : ArrayList
+musteriBekleSure : int
+siparisAlmaSure : int
+maxMusteriGelem : int
+yemekHazirlamaSure : int
+yemekYemeSure:int
+odemeAlmaSure : int
+totalSure : int
+musteriGelmeSuresi:int
+musteriSayisi : int
+contentPanel : JPanel
+JTextField : ayarlar1
+JTextField : ayarlar2
+JTextField : ayarlar3
+JTextField : ayarlar4
+JTextField : ayarlar5
+JTextField : ayarlar6
+JTextField : ayarlar7
+JTextField : ayarlar8
+ unDisable(): void
+sabitAkisStrateji():void
+randomAskısStrateji() : void