Bilgisayar Mühendisliği Bölümü

Yazılım Laboratuvarı I 2020-2021 Güz Dönemi Proje II

Asansörlerdeki Talep Yoğunluğunun Multithread ile Kontrolü

190201105 Alperen İleri-170201093 İlayda Dişiaçık

I.Proje Tanımı ve Aşamaları

I-I.Tanım

"Asansörlerin Talep Yoğunluğunun Multithread ile Kontrolü" adlı programda 5 katlı bir AVM'nin 5 adet asansörü vardır.Asansörlerin taşıyabileceği maksimum insan sayısı 10 ve katlar arası geçiş hızları 200 ms'dir.Bu asansörlerden biri sürekli çalışmaktadır.Diğer dördü ise yoğunluk durumuna göre aktif veya pasif haldedir.AVM'ye 1-10 arasındaki rastgele müşteri 500 ms zaman aralıklarıyla girmektedir.Giren müşteriler rastgele bir kata gitmek için kuyruğa alınır.AVM'den 1-5 arasındaki rastgele müşteri 1000 ms zaman aralıklarıyla çıkmaktadır. Çıkmak isteyen müşterileri rastgele bir kattan (1-4), zemin kata gitmek için kuyruğa alınır.Kuyruktaki müşteriler talep ettikleri katlara taşınır.Kuyrukta bekleyen kişi sayısı maksimum kapasitenin iki katına çıktığında yeni asansör aktif edilir.Kuyrukta bekleyen kişilerin sayısı kapasitenin altına indiğinde asansörlerden biri pasif hale gelir.

I-II.Aşamalar

Proje iki sınıftan oluşmaktadır. **Elevator sınıfı**, bulunulan katı gösterecek currentFloor, hedef katı gösterecek destination, asansörlerin kapasitesini gösterecek capacity, asansörlerin içindeki müşteri sayısını gösterecek count_inside, elevatorID integer parametreleri ve asansörlerin aktifliğini gösterecek active, anlık çalışıp çalışmadığını gösterecek mode boolean parametrelerinden ve bu parametrelerin getter() ve setter() metodlarından oluşturulmuştur.

Yazlab12 sınıfı, asansörleri tutan vektörlerden, bir asansörün katlarındaki müşterileri tutan ve beş asansörün içindeki müşterileri tutan hashmaplerden ve asansörlerin isterlere göre çalışmasını sağlayacak fonksiyonlardan oluşmuştur. Bu fonksiyonlar:

musteriOlustur():1-10 arasında rastgele bir müşteri sayısı ve 1-4 arasında rastgele bir kat oluşturulmuştur. Müşterilere gidecekleri kat bilgisi atanmıştır. Yoğunluk kontrolü yapılarak her asansör için maksimum kapasite iki katına ulaştığında bir sonraki asansör

aktif edilmiştir. Aksi durumda ana asansör hariç, diğer asansörler pasif durumunda tutulmuştur.

musteriCikisi():1-5 arasında rastgele bir müşteri sayısı ve 1-4 arasında rastgele bir kat oluşturulmuştur. Müşterilerin çıkış yapmak isteyeceği kat bilgisine göre kontroller yapılmıştır. Buna göre katlardan müşteriler eksiltilmiştir. Eğer katta bulunan müşterilerden daha fazla sayıda müşteri çıkış yapmak isterse kattaki müşteri sayısı sıfırlanır.

Main(): createElevator() ve threadContoller() fonksiyonları tetiklenmiştir.

threadController(): Bu fonksiyon içinde giriş çıkış fonksiyonları için ve her asansör için eş zamanlı çalışabilmeyi sağlayacak threadler tanımlanmıştır. Her asansör için ayrı thread oluşturulmuştur. Bu threadlerin işlevi; elevatorMove() fonksiyonunu çalıştır, asansörün için boşalana kadar döngü içerisinde; asansörü sıra sıra bir kat yukarı çıkar, eğer bulunan katta inecek müşteri varsa indir, bulunan kattaki müşterileri tutan listeye at, asansördeki müşteriyi tutan listeden çıkar. Asansörde müşteri kalmayınca asansörü zemin kata indir ve asansör müsaittir olarak düzenle. Şeklinde bir işleyişe sahiptir ve bunu her asansör threadi kendi içerisinde uygular.

elevatorMove(Elevator elvtr, int index):Müşteri varsa ve asansör aktifse işlem yapacaktır. Müşteri sayısı 10'dan fazla ise asansörlerin içindeki müşteriyi tutan vektöre erişip, müşterinin gideceği kat bilgisi atılmıştır. Müşteri sayısı 10'dan az ise tüm müşteriler asansörlere bindirilmiştir. removeElevator(Elevator elvtr, int index) fonksiyonunu çağırılmıştır.

editHashMap(): Zemin katın asansörlere müşteri taşındıktan sonraki boyutu kadar dönen döngüler ile zemin kattaki müşterileri tutan listenin elemanlarını kaydırma işlemi yapan fonksiyondur.

createElevator(): Asansör oluşturma fonksiyonudur. Her bir asansörün aktiflik pasiflik bilgisi ve ID değeri belirlenmiştir. Asansörlere toplu ulaşımı kolaylaştırmak için; asansörler, asansör tipinde oluşturulan vektörde tutulmuştur. Asansör içerisindeki müşterilerin kontrolü için ise asansör içlerini tutan hash mapler aynı şekilde vektör içerisinde tutulmuştur.

outputScreen(): Ekran bastırma işleminin yapıldığı fonksiyondur. İstenilen arayüze göre düzenlenmiştir.

II.Temel Bilgiler ve Yapılan Araştırmalar

II.I-Proje Sırasında Yararlanılan Teknolojiler

Proje, Java programlama dili kullanılarak NetBeans geliştirme ortamında oluşturuldu. Programı yaparken Java dilinin bize sunduğu kütüphane ve fonksiyonlardan yararlanıldı.

Kütüphaneler;

java.util.HashMap

java.util.Random

java.util.Vector

kullanıldı.

II.II-Yapılan Araştırmalar

Proje için aynı anda birden fazla iş yapmaya yarayan thread yapısı kullanıldı. Bu yapının çalışma mantığı araştırıldı ve örnekleri incelendi. Bir metodu aynı anda birden fazla threadin işletememesi için kullanılan synchronized anahtar kelimesinin nasıl kullanılabileceği araştırıldı ve örnekler incelendi.

III.Genel Yapı

Geliştirdiğimiz projede 7 adet thread kullandık. Bunların 2si giriş çıkış, 5i asansör threadleridir. Program çalıştığında 500ms arayla 1-10 arası rastgele müşteri gelmekte ve ekrana arayüzü basan fonksiyon tetiklenmekte, anlık olarak ise asansör threadleri çalışmaktadır. Eğer zemin katta müşteri yoksa asansörler işlem yapmaz, eğer 2,3,4,5. Asansörler aktif değilse threadleri işlem yapmaz, 1000ms arayla ise 1-5 arası rastgele müşteri çıkışını tetikleyen thread çalışır, şeklinde bir yapı tasarlanmıştır.

VI.Ekran Çıktıları

VII.Kaynakça

https://ufukuzun.wordpress.com/2015/02/22/javada-multithreading-bolum-2-threadlerin-senkronizasyonuna-giris/

https://ufukuzun.wordpress.com/2015/02/26/javada-multithreading-bolum-3-synchronized-anahtar-kelimesi/

https://ufukuzun.wordpress.com/2015/03/29/javada-multithreading-bolum-4-synchronized-kod-bloklari/

https://www.youtube.com/watch?v=-BbRdNwRghg

https://www.youtube.com/watch?v=JceAHRlQsqc

https://www.youtube.com/watch?v=IEJBztermSA

https://www.youtube.com/watch?v=L35a6EVumcc&t=5s