T.C.

SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR ve BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

PROGRAMLAMA DİLLERİNİN PRENSİPLERİ DERSİ

ÖDEV 4 – RAPOR

JAVA DİLİNDE MULTITHREADING İLE

EŞ ZAMANLI PROGRAMLAMANIN SAĞLANMASI

Hazırlayan Öğrencinin;

Adı: Oğuzhan

Soyadı: Tohumcu

Numarası: B181210397

Serdivan / SAKARYA

Mayıs, 2019

KISA ÖZET

Bu ödevde, bir dosyanın içerisinde bulunan sayıları okuyup, her sütunu kendi içerisinde toplayarak en sonunda oluşan sonucu ekrana yazdıran, bir Java programı yazılmıştır. Okunan dosyanın adı "Sayilar.txt" olarak verilmiştir.

Ödevde kullanılan bazı önemli özellikler:

- BufferedReader & FileReader
- Try-catch & finally blocks
- Array List & List<Integer>
- Implement Runnable class (Multithreading for Parallel Calculation)
- Concurrent Execution (Thread Pool, Override Run Method, Execute and Shutdown)
- Exception Handling (for FileNotFoundException, IOException)

ALGORİTMA, YAZILIM GELİŞTİRME ve SONUÇ

Öncelikle, ödev Eş Zamanlı programlama olduğu için ve Java'da yazılması istendiği için; Java dilinde multithreading yapısının kullanımını öğrendim. Öğrendiklerim üzerinden yola çıkarak ödevin programlamasını gerçekleştirmek adına kendi **algoritmamı** geliştirdim.

Bizden istenen, dosyadan okunan 4 basamaklı sayıların birler, onlar, yüzler ve binler basamaklarındaki sayıları ayrı ayrı toplayarak çıkan sonuç ile sonucun hesaplanması esnasında geçen sürenin, seri ve paralel hesaplamalara göre kaç milisaniye sürdüğünün de ekrana yazılması yönündeydi.

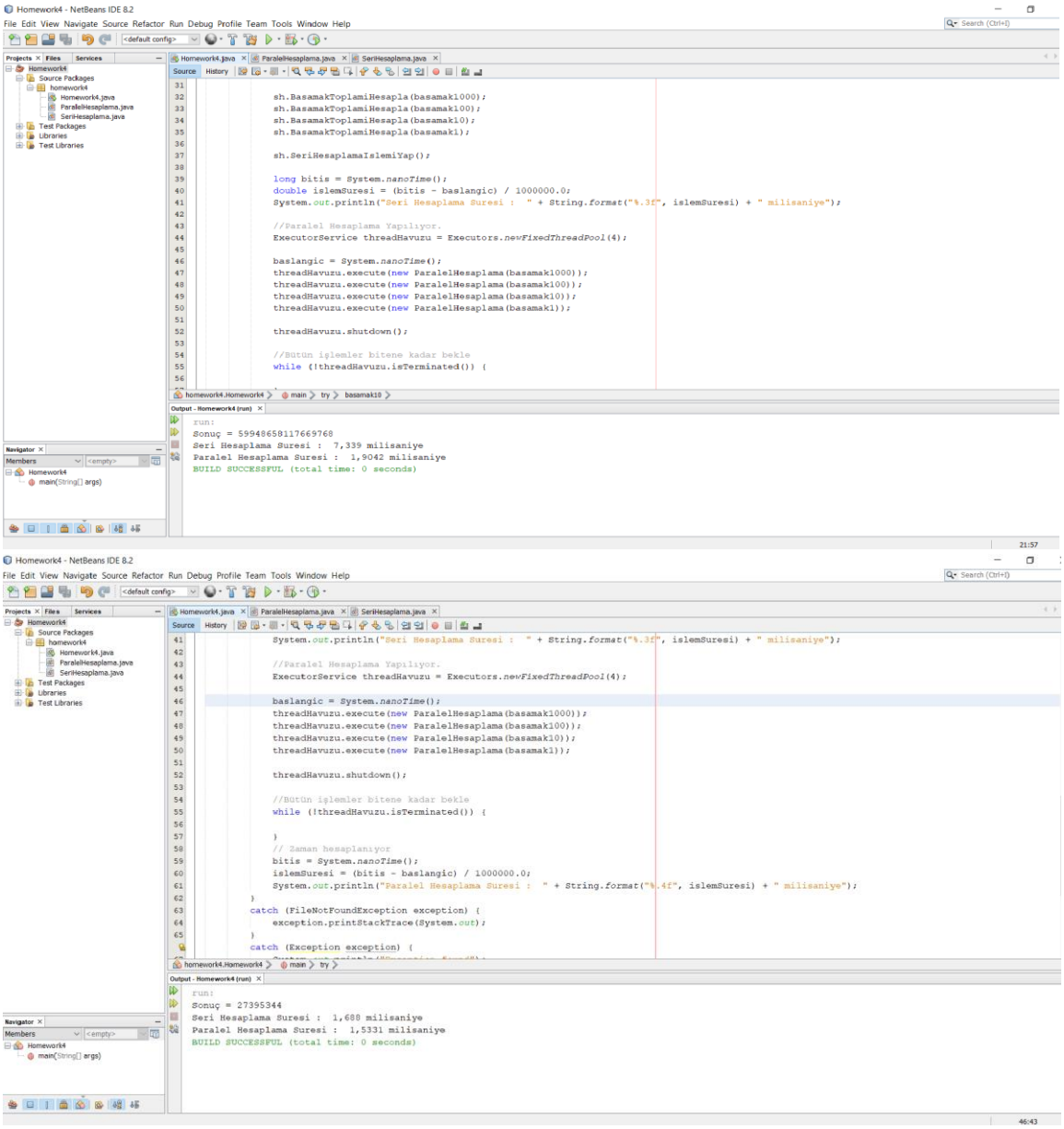
Bu yüzden önce thread kullanılmadan yapılan seri hesaplama için bir sınıf oluşturdum ve bu sınıf içinde dosyadan alınan her bir sayı ile 4 basamak için, 5 farklı integer array list'i tanımladım. Ardından **SeriHesaplama** sınıfının constructor'ını ve basamakların get metodlarını yazdım. Daha sonra, dosyadan okumanın yapıldığı **OkumaIslemi** metodunu, olası bir hataya karşı, try-catch-finally blokları içerisinde exception kontrolü yaparak yazdım. Her bir basamak toplamını, **BasamakToplamiHesapla** metodu ile metoda gönderilen liste üzerinde çeşitli işlemler yaparak buldum. Bu sınıf için, son olarak **SeriHesaplamaIslemiYap** metodunda her basamağın toplamına göre oluşan sonucu hesapladım ve konsola yazdırdım.

Eş zamanlı programlama için, **ParalelHesaplama** adında bir sınıf oluşturdum. Bu sınıfı **multithreading** yapısını gerçekleştirebilmek için, **Runnable** class'ından implement ederek oluşturdum. Ardından sınıfa ait bir adet no parameter constructor ile basamakların toplamının hesabı için çeşitli işlemler yaptığım **BasamakToplamiParalelHesapla** metodunu yazdım. Son olarak, Runnable sınıfına ait **run** metodunu override ettim.

Şimdi sıra yazılan metodların main içerisinde gerçekleştirilmesinde. Bunun için öncelikle sistemin başlangıç zamanını elde ettim ve basamaklara ait 4 ayrı array list tanımladım. Seri hesaplama sınıfından bir instance oluşturup, bu örnek üzerinden dosyayı okuyarak; her sayının basamaklarını elde ettim. Ardından basamak toplamlarını hesaplayarak, seri hesaplama sonucunu buldum. Seri hesaplama süresi için ise, işlem bittikten sonra aldığım sistem zamanından, başta aldığım zamanı çıkararak milisaniye cinsine çevirdim. Paralel kısmında ise, bir thread havuzu üzerinden 4 basamak için 4 farklı **execute** ile **multithreading** işlemini yaptım ve havuzu **shutdown** ile kapattım. Ardından bütün işlemler bitene kadar bekledim ve seri tarafında olduğu gibi paralel hesaplama süresini de hesapladım. Tüm bu işlemleri **try-catch** bloğunda yaparak exception kontrolünü sağladım.

EKRAN ÇIKTILARI

Ekran çıktılarını doğru olarak elde ettim.



REFERANSLAR

[1] https://winterbe.com/posts/2015/04/07/java8-concurrency-tutorial-thread-executor-examples/

[2] https://www.mkyong.com/java/how-to-read-file-from-java-bufferedreader-example/

[3] https://medium.com/gokhanyavas/javada-multithreading-bbc6a9181772

Not: İşbu rapor şahsıma ait olup, yararlandığım kaynaklar yukarıdaki gibidir. Kimseyle paylaşmadığımı belirtmek isterim.