# SAKARYA ÜNİVERSİTESİ – BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

# VERİ YAPILARI DERSİ YAZ OKULU 1. ÖDEVİ RAPORU

Ödeve başlarken derste işlediğimiz recursive fonksiyonları kullanabilirmiyiz diye düşündüm. Ancak daha sonradan recursive fonksiyonların performans açısından düşük olduğu göz önünde bulundurarak ve büyük sayıların faktöriyelinde performans önemli olduğundan bu fikirden vazgeçtim. Ancak yine de ödevde recursive fonksiyonlar kullanmak istedim.

Main.cpp dosyasi içersinde kullanıcıdan faktöriyeli alınacak sayıyı istedim daha sonra negatif bir giriş olduysa ekrana pozitif girmesi konusunda yazı çıkartıyoruz. Eğer girilen sayı 1 ile 11 arasında ise (11 dahil değil) bu işlem yükü açısından büyük bir yük olmadığından aynı zamanda ödevde istenen basamak kaydırmama işlemine gerek olmadığından (normal faktöriyel hesabı) bunu direk bir recursive.cpp içinde yapıp Sonuc.txt ile sonucu çıkartıyoruz.

Son olarak girilen sayının 11 den büyük olması durumu geliyor. Belki 11 12 13 gibi sayıların faktöriyelini bir veri tipine sığdırabiliriz ancak çok büyük sayıların faktöriyeli istendiğinde bunlar yeterli olmayacaktır. Bunun için iki aşamada işlem yapmam gerektiğini düşündüm ilk aşamada basamak kaydırmadan ödevde istenilen çarpma , ikinci işlemimiz ise büyük sayıları tutacak veri tipi. Bunun için hocamızın yaptığı gibi bir ArrayList sınıfı tanımladım ve içinde gerekli methodları tanımladım. ( eleman ekleme , silme, dizi boşluk kontrolü ? vb işlemleri için). Arraylist'e göre sonuçlar diziye eklenecek dizinin boyutu doldukça dizinin kapasitesi arttıralacak . Calculator.cpp dosyasinda basamak kaydirmasiz çarpma işlemini yapildi.

Sonra sonuçlar diziye eklendi ancak şöyle bir sorun oldu. Örneğin dizi yetersiz olduğu için dizi kapasitesi arttırılıyor ancak bu sefer dizinin boyutu bizim sayımızın uzunluğundan daha büyük olabiliyor. Bu durumda dizinin bos indislerine otomatik olarak sıfır atanıyor. Yani 213123 gercek sayımız olsun ancak sonuç.txt te 00000213123 gibi bir değer görüyorduk. Yani baştaki 0 lardan arındırılması gerekiyordu. Bu sorunu da factorial.cpp ye yazılan baştaki 0 ları bulan ve silen bir while döngüsüyle çözdükten sonra programımız tamam gibiydi. Ancak başka gibi sorun vardı…

İşlem yükü arttıkça (daha büyük sayıların faktöriyeli hesaplanması istendikçe) programın çalışma hızı yavaşlıyordu. Basit faktöriyelleri hemen hesaplarken büyük sayılarda gecikmeler yaşanıyordu. Bu sorunun çözümü için "new" komutuyla aldığımız yerleri belleğe "delete" komutuyla ile iade ettik. Heap"ten alınan yerler iade edildi. Ayrıca gereksiz döngü kullanımından kaçınıldı. Mümkün olduğunca az değişken kullanıldı . Gereksiz kod kullanılmasından kaçınıldı . Bunların hepsi programın hızlandırılması içindi . En sonunda şöyle bir sonuç aldım.

2! -> yaklasik olarak aninda , 100! -> 0.6 saniye , 500-> 0.8 saniye , 1000 -> 1 saniye

2000! -> 1.3 saniye , 5000! -> 7 saniye , 10000! -> 25 saniye , 20000! -> 1 dakika 13 saniye

Ve 110000! -> 38 dakika

Görüldüğü gibi yapılan işlemlere rağmen 110bin faktöriyelin hesaplanması 38 dakika kadar tuttu. Bunlar donanımın özelliklerine göre değicektir. Program ödevde istenen 10000 faktoriyeli 24-25 saniye arasında buluyor.

Ödevde başlık ve kaynak dosyaları ayrıldı. Bilgisayara kurduğum g++ kullanıldı, makefile dosyası oluşturuldu.

Ödevde en zorlandığım yer bulunan sayının başındaki 0 lardan kurtulmak ve programı hızlandırmak oldu.

Eksik olan yer olarak programın çok çok büyük sayılarda yavaş hesaplaması .

Dosyalama işlemleri istenilen şekilde yapılmış ve mingw ile derlenmiştir.

Misafir Öğrenci

Yavuz Selim ŞAHİN / [redacted]