**T.C.**

**FIRAT ÜNİVERSİTESİ**

**TEKNOLOJİ FAKÜLTESİ**

**YAZILIM MÜHENDİSLİĞİ**

**YMH 319 Programlama Dilleri**

**Check Word (Tez Kontrol)**



**Cihan GÖNEN-170541010**

**Doç. Dr. Fatih ÖZKAYNAK**

**Şubat – 2021**

Bu dökümanda projemi geliştirirken ve kod yazarken dikkat ettiğim noktalara değineceğim.

Öncelikle kodlama yaparken değişken ataması önemli bir husustur. Ben projemi geliştirirken buna dikkat etmeye çalıştım. Örnek vermek gerekirse tablolar listesinin başlangıç değeri için ‘tablolarBaslangic’ değişken adını kullanırken şekiller listesinin başlangıç değeri için ‘sekillerBaslangic’ , tezin giriş bölümünün başladığı değer için ise ‘girisBaslangic’ değişkenini kullandım. Gördüğümüz üzere belli bir kuralara göre isimlendirme yapılmıştır. Kelimenin ikinci kısmı türünü yani başlangıç değeri olduğunu belirtirken ilk kısmı da hangi başlangıç değeri olduğunu vurgulamaktadır.

Diğer bir programlama yaklaşımı olarak programın tamamında kaç eleman olduğu belli olmayan bir dizi oluşturmaktansa string değişkenler oluşturulmuştur. Programın gidişatına göre bu string değişkenine değerler boşluk bırakılarak aktarılmış ve en sonunda dizi oluşturulup string değişkeni içerisindeki değerler boşlukları kullanarak ayrılıp diziye aktarılmıştır.

Programda gerekli yerlerde karışık olan diziler sıralama algoritmaları kullanılarak sıralanmış ve işlem yapılması daha kolay hale getirilmiştir. Örneğin atıf listesini kullanmadan önce insertion sort sıralama algoritmasını kullanmayı tercih ettim. Burada insertion sort algoritmasının tercih edilmesinin sebebi ise dizinin boyutunun çok büyük olmaması ve tersten sıralı olmasının çok çok düşük bir ihtimal olmasıdır. Insertion sort algoritmasında en kötü durum dizi tersten sıralı olunca oluşur.

Girintiler ve boşluklar gerekli yerlerde kullanıldı. Bu sayede kodun okuması ve anlaşılması kolaylaştırılmış oldu. Açıklama satırı kullanılırken ise yazılmış kodun yarısı kadar olmasına dikkat edildi. Benzer algoritmalarla çalışan fonksiyonlar için özellikle farklı olan yerlere açıklama satırı eklendi ki açıklama satırı da gereksiz fazla olmasın.

Programın geliştirilmesi sırasında sorulan bazı sorular programın iyileştirilebilmesi için çok önemli yer tutmuştur. Kendime sorduğum bu sorular çok basit olmakla beraber beni kodu sorgulamaya ve daha efektif bir algoritma düşünmeye itmiştir. Örneğin "Bu kodu neden yazdım?" veya "Bu kod nasıl daha efektif yazılabilir?" gibi sorular ne kadar basit gözükse de detaylı bir şekilde sorulup cevaplandığında ve aksiyon alındığında programın kalitesi gözle görülür biçimde artmıştır.

Yapısal programlama yaklaşımıyla satırlarca kod gerektiren problemler fonksiyonlara bölüştürülmüş ve gerektiğinde bu fonksiyonlar tekrar kullanılıp karmaşıklığın önüne geçilmiştir. Kod tekrarının önüne geçmeme yarayan bu yapı aynı zamanda kodun okunmasını kolaylaştırıp hataların tespitinde ve onarılmasında da faydalı olmuştur.

Ekstra olarak projenin çalıştırılması ve kullanımı README text dosyasında projeyle birlikte sunulmuştur.