Sınıf Planlama Sistemi

160201079 İlknur GÖK,

160201012 Elanur OCAK

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

[ilknurgok.1@hotmail.com](mailto:ilknurgok.1@hotmail.com) , [elaocak45@gmail.com](mailto:elaocak45@gmail.com)

**Özet**

Projede verilen txt dosyası üzerinden verileri okutuldu ve aynı ada ve soyada sahip öğrenciler silindi. Daha sonra kayıt sıraları güncellenip numara ataması yapıldı. 1. öğretim ve 2. öğretim sıralı halde dosyaya yazdırıldı. Eğer kullanıcı yeni kayıt girdiyse kayıtlar listelere yazdırıldı. Bu kayıtlar daha sonra eşit ve en az öğrenci dağılım ilkelerine göre sınıflara atandı.

**1.Giriş**

Projenin genel olarak konusu, verilen txt dosyasının üzerinde istenilen bir kaç dosyalama işleminin uygulanmasını sağlamaktır.

Hazırlanan projenin amacı, verilen txt dosyasında yapılması istenilen dosyalama işlemlerine göre çeşitli dosya oluşturma işlemi ile yeni dosyalar oluşturmak, verilen veya oluşturulan dosyalar üzerinde ekleme, çıkarma, güncelleme ya da istenilen işlemler ile ilgili değişiklik yapılmasını sağlamaktır.

**2.Temel Bilgiler**

Proje C dilinde yazıldı ve CodeBlocks’ta hazırlandı. Dosya oluşturma kısmı Notepad’de tamamlandı.

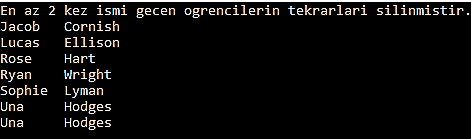
* Code::Blocks, özgür açık kaynak kodlu bir C++ tümleşik geliştirme ortamıdır. wxWidgets tabanlı tamamen özelleştirilebilir arabirimiyle, GNU/Linux, Microsoft Windows, MacOS platformlarında sorunsuzca kullanılabilmektedir.
* Notepad++, Windows işletim sistemi içerisine gömülü olarak gelen Notepad yazılımının yerine kullanılmak üzere C++ ile saf Win32 API ve STL ile geliştirilmiş GPL ile dağıtılan açık kaynak kodlu bir kaynak kod düzenleyicisidir.

**3.Diğer Bölümler**

Projede genel olarak, bir derse kayıt yaptıran öğrenciler üzerinde sınıf planlama sistemi geliştirilmesi istenmektedir. Proje ile üzerinde dosyalama işlemlerinin gerçekleştirilmesi istenen dosya verilmiştir. Dosyada öğrenci numarası, ad, soyad, öğretim türü ve kayıt sırası gibi derse kayıtlı öğrenciler ile ilgili bilgiler yer almaktadır.

*3.1 Kayıt Silme*

İlk olarak istenen işlem, verilen bu dosya içinde bulunan öğrencilerden derse birden fazla kayıt yaptırılanları bulmak ve bu öğrencilerin bilgilerini dosyadan çıkarmayı sağlamaktır. Projemizde ilk olarak, verilen dosyayı taradıktan sonra bulunan bu öğrencilerin aynı ad, soyada vb. sahip olanlarının bilgilerinin silinmesini sağladık. Daha sonra oluşturduğumuz veri\_atama() fonksiyonu ile dosyadan aldığımız öğrenci numarası, ad, soyad vb. bilgileri öğretim türlerine göre ayırıp başka struct yapılarına atamasını yaptık. Ayrıca bu fonksiyon ile ilerideki dosyalama işleminde kullanılabilecek olabileceği için dosyada kayıt silme işlemi sonrasında kalan öğrenci sayısını tespit ettik.



Şekil 1:Kayıt Silme

*3.2 Kayıt Sırası Güncelleme*

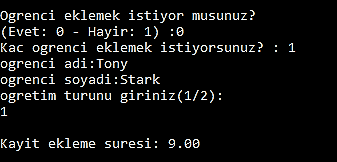
İkinci olarak istenen işlem, tekrarlanan kayıtlar silindikten sonra kayıt sıra bilgilerinin güncellenmesini sağlamaktır. Projemizde kayıt silinmesi işlemi sırasında kaydı fazladan bulunan öğrenciler ile ilgili bilgiler silindiğinde kayıt sıralarını da değiştirmiş olduk.

*3.3 Numarası Olmayan Öğrencilere Numara Atama*

Üçüncü olarak istenen işlem, verilen dosya içerisinde üniversiteye yeni kayıt yaptıran öğrencilerin numarası bulunmadığından bu öğrencilere numara atanmasını sağlamaktır. Projemizde öncelikle, kayıt silme işlemini yaparken başka struct yapılarını atadığımız bilgiler için ayrı ayrı, öğretim türlerine göre 1.ogr ve 2.ogr adında dosyalar oluşturduk. Bu dosyalar içine öğrenci bilgilerini atarken aynı zamanda, numarası olmayan öğrencilere verilen kayıt sıralarını dikkate alarak yeni numaralar atanmasını sağladık. Bu bilgileri atama yaparken aynı zamanda dosyalara kayıt sırasına göre sıralı bir şekilde yazdırdık. Ve atamayı yaparken ileride kullanılabilecek olan birinci ve ikinci öğretim öğrenci sayılarını tespit ettik.

*3.4 Yeni Kayıt Ekleme*

Projede ek olarak istenen kısım, yeni öğrenci kayıt girişinin yapılabilmesini sağlamaktır. Projemizde numara atama işleminden hemen sonra, yeni öğrenci kayıt girişinin yapılmasını sağladık. Yeni girişi sağlanan öğrencinin öğretim türüne göre ise daha önceden oluşturduğumuz 1.ogr ve 2.ogr isimli dosyalara yeni öğrenci ile ilgili bilgileri atadık. Bu atamayı yaparken aynı zamanda daha önceden bulduğumuz öğrenci numarası sayısına göre yeni öğrencinin numarasını oluşturmuş olduk. Yeni öğrencinin kayıt sırasını ise daha önceden bulduğumuz kayıt sırasına göre düzenledik. Ve bu sıralı olan iki dosyayı yeni bir dosyaya önce birinci sonra da ikinci öğretim olarak, öğrenci numarası sırasına göre başka bir dosyaya yazdırdık.



Şekil 2:Öğrenci Ekleme

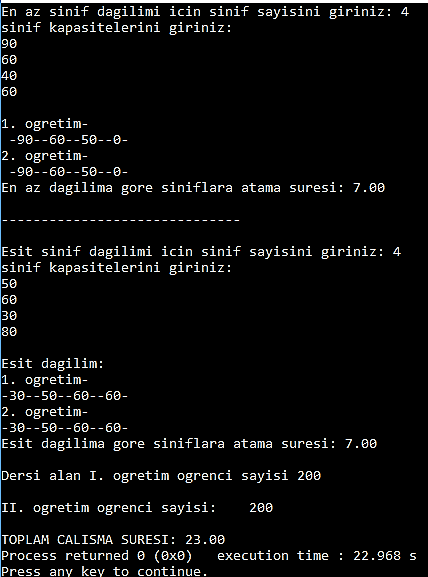
*3.5 Numaraya Göre Sıralama*

Dördüncü olarak istenen işlem, isme göre sıralı verilen dosyanın öğrenci numaralarına göre sıralanmasını sağlamaktır. Projemizde kayıt ekleme kısmından sonra oluşturduğumuz sırala() fonksiyonu ile dosyayı öğrenci numarasına göre sıralamış olduk.

*3.6 Sınıflara Öğrenci Atama*

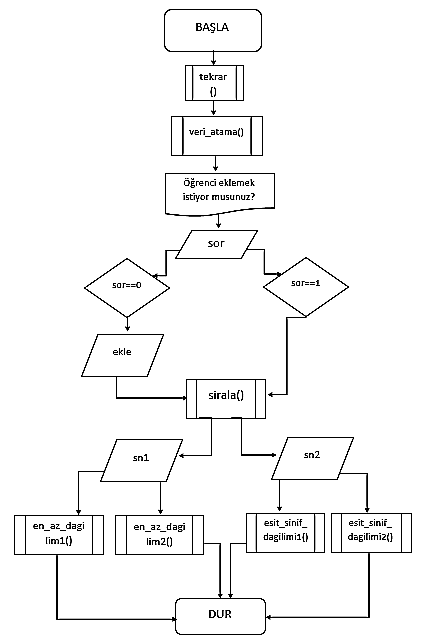
Beşinci olarak istenen işlem, kullanıcıdan alınan sınıf sayısı ve kapasitelerine göre listedeki öğrencilerin dağıtılmasını sağlamaktır.

Projemizde ilk sınıf ataması işlemi olarak en az sınıf dağılımına göre işlemler yapılmasını sağladık. Sınıflara en az sınıf dağılımı öğrenci ataması işlemini gerçekleştirmeden önce ilk olarak, ileride dağıtım yapılacak olan sınıf sayısı ve bu sınıfların kapasitelerinin kullanıcıdan alınmasını sağladık. Alınan sınıf kapasitelerinin bulunduğu diziyi, en az sınıf dağılımı işlemini uygulayabilmek için büyükten küçüğe doğru sıraladık. Sonrasında sınıf kapasitelerinin daha önceki işlemlerde bulduğumuz öğrenci sayılarından fazla veya az olup olmadığını anlayabilmek için öncelikle sınıf kapasitelerini topladık. Bu toplam ile öğrenci sayılarını karşılaştırdık. Eğer kapasite toplamı, öğrenci sayılarından küçük ise yeni kapasite girişinin yapılmasını istedik ve kapasite toplamı, öğrenci sayısından büyük veya eşit olana kadar kapasite isteme işlemini tekrarlanmadık. Kapasite toplamı, öğrenci sayısından büyük veya eşit olduğunda ise birinci ve ikinci öğretim öğrencileri için ayrı olarak oluşturduğumuz en\_az\_dagilim1() ve en\_az\_dagilim2() fonksiyonlarına geçiş sağladık. Bu iki fonksiyonda ise öncelikle en az sınıf dağılımı işlemini sağlayabilmek için fonksiyona büyükten küçüğe sıralı halde gönderdiğimiz kapasite dizisinin elemanları ile bir diğer fonksiyona gönderdiğimiz öğrenci sayısı ile karşılaştırma yaptık. Öğrenci sayısı, kapasiteden büyük olduğunda kapasite boyutu kadar öğrenciyi bu sınıfa doldurduk ve öğrenci sayısını dolan sınıf kapasitesi kadar azalttık. Daha sonrasında bu işlemi kapasite dizisinin diğer elemanları ile tekrarladık. Öğrenci sayısı sınıf kapasitesinden küçük olduğu durumda ise, kalan öğrenci sayısı kadarı ile bu sınıfa atadık ve bu sınıflara ayırma işlemi için oluşturduğumuz fonksiyonlara göre enazogrdagilimi1 ve enazogrdagilimi2 isimli dosyalar oluşturup ayrılan öğrencileri bu dosyalara yazdırdık.

İkinci sınıf ataması işlemi olarak eşit sınıf dağılımına göre işlemler yapılmasını sağladık. Sınıflara eşit sınıf dağılımı öğrenci atamasını gerçekleştirmeden önce ilk olarak, ileride dağıtım işlemi yapılacak olan sınıf sayısı ve kapasitelerinin kullanıcıdan alınmasını sağladık. Aldığımız bu sınıf kapasitelerinin bulunduğu diziyi, eşit sınıf dağılımı işlemini uygulayabilmek için küçükten büyüğe doğru sıraladık. Sonrasında sınıf kapasitelerinin daha önceki işlemlerde bulduğumuz öğrenci sayılarından fazla veya az olup olmadığını anlayabilmek için öncelikle sınıf kapasitelerini topladık, bu toplam ile sınıf kapasitelerini karşılaştırdık. Eğer kapasite toplamı, öğrenci sayılarından küçük ise yeni kapasite girişinin yapılmasını istedik ve kapasite toplamı, öğrenci sayısından büyük veya eşit olana kadar kapasite isteme işlemini tekrarladık. Kapasite toplamı, öğrenci sayısından büyük veya eşit olduğunda ise birinci ve ikinci öğretim öğrencilerini için ayrı olarak oluşturduğumuz esit\_sinif\_dagilimi1() ve esit\_sinif\_dagilimi2() fonksiyonlarına geçiş sağladık. Bu fonksiyonlarda ise öncelikle eşit sınıf dağılımı işlemini sağlayabilmek için fonksiyona küçükten büyüğe sıralı halde gönderdiğimiz kapasite dizisinin elemanları ile bir diğer fonksiyona gönderdiğimiz öğrenci sayısı ve sınıf sayısının bölümünün karşılaştırmasını yaptık. Öğrenci sayısı ile sınıf sayısı bölümü, kapasiteden büyük olduğunda öncelikli olarak bu sınıfı kapasitesi kadar öğrenci ile dolmasını sağladık. Öğrenci sayısını dolan sınıf kapasitesi kadar azalttık. Daha sonrasında bu işlemi kalan öğrenci sayısı ve kalan diğer sınıf sayısına göre kapasite dizisinin diğer elemanları ile karşılaştırmasını yaptık. Öğrenci sayısı ile sınıf sayısı bölümü, sınıf kapasitesinden küçük olduğunda ise, bu bölüm sayısı kadar kalan öğrenciyi bu sınıf ya da sınıflara eşit sayıda doldurmuş olduk. Eğer en başta öğrenci sayısı ile sınıf sayısı bölümü, kapasite dizisinin ilk elemanından da küçük ise bütün sınıflara bu bölüm kadar öğrenciyi doldurmayı sağladık. Ve bu sınıflara ayırma işlemi için oluşturduğumuz fonksiyonlara göre esitogrdagilimi1 ve esitogrdagilimi2 isimli dosyalar oluşturup ayrılan öğrencileri bu dosyalara yazdırdık. Son olarak bu derse kayıtlı birinci ve ikinci öğretim öğrencilerinin sayısı ile sınıflara dağıtım işlemi sonuçlarını ekrana yazdırdık.

Şekil 3:Örnek Sınıf Atama Çıktısı

**4.Akış Şeması**



Şekil 4:Akış Şeması

**4.Sonuçlar**

Çalışmamızda genel olarak dosyalama işlemleri ve algoritma mantığı kullanılarak bir uygulama yapılması istenmektedir. Bu proje ile dosyalar üzerindeki kayıtlar için ekleme, çıkarma, güncelleme gibi temel dosya işlemlerinin uygulanması gerekmektedir.

Projede ilk olarak istenen tekrarlanan kayıtların silinmesi işlemini sorunsuz çalıştırdık. İkinci olarak yeni kayıt yaptıran öğrencilere yeni öğrenci numarası atanmasını işlemini de sorunsuz olarak çalıştırdık. Projede bonus kısım olarak verilen yeni kayıt alma kısmında ise alınan yeni kayıtlara numara atanması kısmında hiçbir sorun yaşamadık. Fakat alınan bu kayıtlar için kayıt sırasının güncellenmesi kısmında birkaç sorun yaşadık. Kayıtları aldıktan sonra kodun tekrar derlenmesi ile bu yeni öğrencilerin de kayıt sırasını düzgün atama yaptık. Son olarak istenen sınıflara öğrenci atama işleminde her bir öğretim türü için ayrı dosyalar oluşturup bunlara öğrenci bilgilerini yazdırmasını yapabildik. Fakat her bir sınıf dağılımında ayrılan ve dağıtılması yapılması istenen sınıflar için yeni dosyalar oluşturmayı başaramadık. Bunun yerine her bir sınıf dağılımı sonucunu öğretim türüne göre ayrı olarak tek bir dosya içine yazdırabildik.

Projede gerçekleşen her bir işlem sonrasında geçen süreyi hesaplattık.

**5.Kaynakça**

1. Web Site

<http://www.c4learn.com/c-programs/sorting-elements-of-structure.html>

1. Web Site

<https://stackoverflow.com/questions/21696106/buble-sort-in-c-using-array>

1. Yorulmaz, Muhammet ve Yorulmaz, Seher . “Programlamayı C ile Öğreniyorum”, Palme Yayıncılık, Ankara, 6. Baskı, 2016.
2. <http://bilgisayarkavramlari.sadievrenseker.com/2009/01/01/c-ile-zaman-islemleri/>
3. <http://turgayozgur.com/c-dosyalama-kayit-silme-fonksiyonu.html>
4. <http://www.batuhanduzgun.com/post/2012/02/10/C-ile-Dosya-Islemleri-(EklemeSilmeAramaListeleme).aspx>