PROGRAMLAMA LABORATUVARI 1
1.PROJE

Umut SÜTCÜ 200202038
Ömer ARAN 190202012
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

Özet

Bu doküman Programlama Laboratuvarı 1 dersi 1. Projesi için çözümü açıklamaya yönelik oluşturulmuştur. Dökümanda projenin tanımı, çözüme yönelik yapılan araştırmalar, kullanılan yöntemler, proje hazırlanırken kullanılan geliştirme ortamı ve kod bilgisi gibi programın oluşumunu açıklayan başlıklara yer verilmiştir. Doküman sonunda projeyi hazırlarken kullanılan kaynaklar bulunmaktadır.

Projenin Tanımı

Bu projenin amacı C dilinde dosyalama fonksiyonlarının pratiğinin yapılmasıdır. Bu amaçla ikili dosyaların (binary file) ve metin dosyalarının (text file) kullanıldığı bir uygulama geliştirilecektir. Bu proje kapsamında yoğun indeks (dense index) yapısı gerçekleştirilecektir. Bu indeks yapısında ikili dosya olarak gerçekleştirilen veri dosyasındaki her kayıt için indeks dosyasında bir girdi saklanmaktadır. İndeks dosyasında kayıtlar şeklinde saklanmaktadır. Adres değeri anahtarla ilgili bilginin veri dosyasında hangi offsette saklandığını göstermektedir. Veri dosyasında aynı anahtarla ilgili birden çok kayıt olabileceği için indeks dosyasında aynı anahtara ait birden fazla kayıt bulunabilir. Aşağıdaki görselde bu tür bir indeks yapısı için bir örnek verilmektedir

Anahtar	Adres
1	0
1	4x
2	x
2	3x
2	6x
3	2x
.	
.	
.	

offset	Anahtar
0	1
x	2
2x	3
3x	2
4x	1
5x	.
6x	2
7x	.
8x	.

Görselden de anlaşıldığı gibi indeks dosyasındaki kayıtlar anahtara göre sıralı iken veri dosyasındaki kayıtlar sıralı değildir. Böyle bir indeksleme yapısı veri arama işlemlerinin hızını oldukça hızlandırmaktadır. İndeks dosyası kullanılmadan bir kayıt veri dosyasında aranacak olsa tüm kayıtlar dosya baştan sona okunması gerekir. Veri dosyasındaki kayıtların büyük ve çok sayıda kayıt olduğu düşünüldüğünde veri dosyasının baştan sona okunması çok fazla sayıda disk erişimi gerektirir. Oysa ki indeks dosyası kullanıldığında, dosyalar arama anahtarına göre sıralı olduğundan ikili arama yapılabilir. Diğer taraftan indeks dosyasındaki kayıtlar veri dosyasındaki kayıtlara göre çok daha küçük olduğundan indeks dosyasının belleğe aktarılması çok daha az disk erişimi gerektirir. Bu proje kapsamında veri dosyasında bir okuldaki öğrencilere ait notlar saklanacaktır. Her öğrenci için aşağıdaki struct yapısında bilgi saklanacaktır. Bu yapıda ogrNo, dersKodu ve puan alanları bulunmak zorundadır. Ancak projenizi gerçekleştirirken ihtiyaç duyabileceğiniz başka alanlar da tanımlayabilirsiniz. Bir öğrenci için birden çok kayıt olabileceğini göz önünde bulundurunuz. Veri dosyasının bir ikili dosya (binary file) şeklinde gerçeklenmelidir.

Gerçekleştirilecek fonksiyonlar:

• indexDosyasiOlustur: Bir veri dosyasının henüz indekslenmediğini kabul edip veri dosyasındaki kayıtlar için bir indeks dosyası oluşturulacaktır.

• kayitEkle: Veri dosyasına yeni bir kayıt eklenecektir. Yeni kayıt her zaman veri dosyasının sonuna eklenmelidir. Kayıt eklendikten sonra indeks dosyası güncellenmelidir.

• kayitBul: Verilen bir anahtar için veri dosyasındaki ilk kayıt gösterilecektir. İlgili anahtar için birden çok kayıt bulunabileceği için aranılan kayıt bulunana kadar anahtara ait kayıtlar listelenecektir. İndeks dosyasında anahtara ait ilk kaydın bulunmasında ikili arama (binary search) algoritması kullanılacaktır

• kayitSil: Verilen bir anahtar için ilgili doğru kayıt bulunacak (kayitBul işlemindeki gibi) ve bu kayıt silinecektir. Silme işlemi indeks dosyasına da yansıtılacaktır.

• kayitGuncelle: Bir anahtar için aranılan kayıt bulunup sadece puan alanı güncellenecektir.

• veriDosyasiniGoster: Veri dosyasındaki tüm kayıtları listeler

• indeksDosyasiniGoster:İndeks dosyasındaki tüm kayıtları gösterir.

• indeksDosyasiniSil:İndeks dosyasını diskten siler.

Araştırma ve Yöntemler

Projede öncelikli olarak bir struct yapısı oluşturup öğrenci bilgilerinde onun içinde tutmamız lazımdı biz de bunun için

```
struct kayit{
   int ogrno
   int derskodu
   int puan
   int derskodu2
   int puan2
}
```

Bu şekilde bir struct yapısı oluşturduk.Birden fazla öğrenci bilgisi olduğu için dinamik bellek ile struct dizisi oluşturduk ve bütün işlemlerimizi bunun içinde yaptık.Projede txt file oluşturup onun üzerinde işlem yapabilmek için ilk önce binary file oluşturup onun onu üzerinden işlem yapılması gerekiyor.Bunun için binary file üzerine araştırmalar yapıp,kullanım yöntemlerini bulduk.Bulduğumuz yöntemlerden bizim için en uygunu binary filedan karakter karakter okuyup bunu txtye yazmak oldu.Bunun için fgetc fonksiyonunu kullandık.Veriler üzerinde arama yapabilmek için binary search kullanmamız gerekiyordu.Bunun için binary search nasıl kullanılır onu araştırdık ve binary search için verilerin sıralı olması gerektiğini fark ettik.Verileri sıralamak için bubble sort algoritmasını kullandık.Binary search algoritmasını kayıtbul,kayıtgüncelle ve kayıtsil fonksiyonlarında kullandık.Kayıt silme işleminde silinecek verilerin üzerine bir sonraki veriyi yazarak diziyi bir sıra geriye kaydırdık. Kayıt bulma işleminde binary search ile öğrenci numarası girilen öğrencinin bilgilerini getirdik.Kayıt güncelle işleminde yeni girilen notu eskisinin üzerine yazarak güncelledik.Kayıt ekle işleminde dizide yeni bir yer açarak öğrencinin bütün bilgilerini oraya girmesini sağladık.Veri ve indeks dosyasını göstermek için fgetc fonksiyonu yardımıyla karakter karakter okuyup ekrana yazdırdık.İndeks dosyasını silme işleminde ise remove komutunu kullanarak indeks dosyasını diskten kaldırdık.

Kullanıcının bu fonksiyonlar arasında istediği gibi dolaşabilmesi için switch case yapısını kullanarak bir kullanıcı arayüzü ayarladık.

Geliştirme Ortamı

Projeyi Windows ve macOS sistemlerde CodeBlocks ve Xcode üzerinde geliştirdik.

Kod Bilgisi

Akış Şeması

Akış şeması ektedir.

Algoritma

Program çalıştığında 50 tane öğrenci veriyor ama bunları henüz txt file olarak yazmamış durumda fonksiyonlardan indeks dosyasısı oluştur seçildiğinde bir txt dosyası oluşturuyor ve bunu bir kontrole sokuyor eğer başarılı bir şekilde oluştuysa bunu bildirerek kullanıcı arayüzüne dönüyor.Kullanıcı arayüzünde seçilene göre gerekli fonksiyona giderek işlemler gerçekleşiyor.Kayıt bul fonksiyonunda binary search kullanılarak aranan öğrenci bulunuyor.Kayıt ekle ,kayıt güncelle ve kayıt sil fonksiyonlarında yapılan değişikler hem veri dosyasına kaydediliyor hem de indeks dosyasına kaydediliyor.Değişiklik yapılan her fonksiyondan sonra indeks dosyası oluştur fonksiyonu tekrar çağırılıyor ve bu sayede yapılan değişikler indeks dosyasına da işlenmiş oluyor.

Kaynakça

1.Dosya fonksiyonları için

https://www.geeksforgeeks.org/basics-file-handling-c/

https://www.bilgigunlugum.net/prog/cprog/c_dosya

https://web.cs.hacettepe.edu.tr/~maydos/Docs/c/dosyalar.pdf

2.Karşılaştığımız birçok hata için

https://stackoverflow.com