BLM 102- Algoritmalar ve Programlama II
2019-2020 Bahar Dönemi

Telefon Rehberi Yönetim Sistemi

Proje Teslim Raporu
10 Haziran 2020

İrem Kalkanlı, Aysen İpek Çakır, Deniz Uzun, Özlem Çalı

Öğretim Elemanı: Dr. Vecdi Emre Levent, emre.levent@fbu.edu.tr, İzinsiz Kopyalanamaz

1 Giriş

1.1 Projenin Amacı

Bir telefon rehberinde bulunması gereken kayıt ekleme, kayıtları gösterme, kayıtları modifiye etme, kayıt arama ve silme kabiliyetlerine sahip olan bir sistem geliştirilmesi amaçlanmıştır. Geliştirilen sistem, kullanıcıya komut satırı arayüzü sunarak rehberin kullanımını sağlayacaktır. Tüm kayıtların bir dosyada tutulması ve gerektiğinde dosyadaki bilgilerin güncellenmesi amaçlanmıştır.

1.2 Proje Ekibi

İrem KALKANLI
Okul numarası:190301007
Doğum Tarihi:15.01.2000
Doğum Yeri: İstanbul
Mezun Olduğu Lise: Özel Batı Ataşehir Doğa Bilim Anadolu Lisesi

Özlem ÇALI:
Okul numarası:190301002
Doğum Tarihi:19.05.2000
Doğum Yeri: Hatay
Mezun Olduğu Lise: Necmi Asfuroğlu Anadolu Lisesi

Deniz UZUN:
Okul numarası:190301015
Doğum Tarihi:08.04.2001
Doğum Yeri: İstanbul
Mezun Olduğu Lise: Kavacık Uğur Anadolu Lisesi

Aysen İpek ÇAKIR:
Okul numarası:190301001
Doğum Tarihi:20.03.2001
Doğum Yeri: Malatya
Mezun Olduğu Lise: Fethi Gemuhluoğlu Fen Lisesi

2 Geliştirilen Uygulama

2.1 Kullanılan Araçlar

Tasarım geliştirilirken Microsoft'un derleyicisi olan Visual Studio Community aracı kullanılacaktır.

Visual Studio Community
Visual Studio, birçok programlama dilini kullanarak program, uygulama ya da web sitesi yapabileceğiniz bir IDE yani entegre geliştirme ortamıdır. Microsoft Windows için bilgisayar programları, web siteleri, web uygulamaları, web hizmetleri ve mobil uygulamalar geliştirmek için kullanılır.

Visual Studio, Windows API, Windows Forms, Windows Presentation Foundation, Windows Store ve Microsoft Silverlight gibi Microsoft yazılım geliştirme platformlarını kullanır. Hem yerel kod hem de yönetilen kod üretebilir.

Visual Studio, IntelliSense'i (kod tamamlama bileşeni) ve kod yeniden düzenleme işlemini destekleyen bir kod düzenleyici içerir. Entegre hata ayıklayıcı, hem kaynak düzeyinde hata ayıklayıcı hem de makine düzeyinde hata ayıklayıcı olarak çalışır. Diğer yerleşik araçlar arasında bir kod profili oluşturucu, GUI uygulamaları oluşturmak için form tasarımcısı, web tasarımcısı, sınıf tasarımcısı ve veritabanı şeması tasarımcısı bulunur. Neredeyse her düzeyde işlevselliği artıran eklentileri kabul eder.

2.2 Tasarım

Tasarım yapılırken nesneye yönelimli programlama (sınıflar, kalıtım, çok biçimlilik) ve modüler programlama (operatör aşırı yükleme, şablonlar, istisna idaresi) yaklaşımlarını bu tasarımda nerelerde, hangi nesneye yönelik programlama başlığını kullanacağımızı ayarladık.

Yapılma Aşamaları

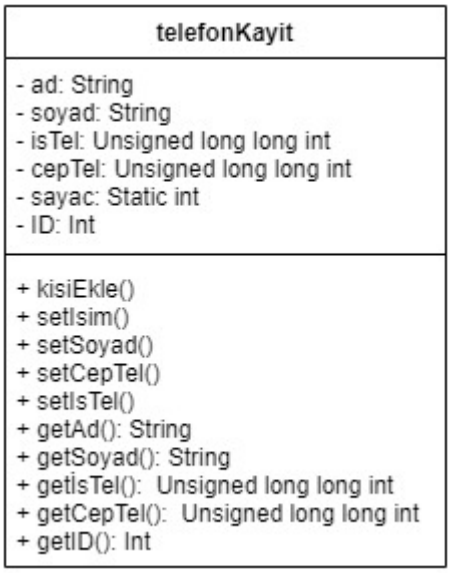
Tasarımı yaparken, projemizde kullanım kolaylığı açısından bir sınıf oluşturarak hazırlamayı tercih ettik. Bunun için telefonKayit.h adında bir adet header dosyası, telefonKayit.cpp adında bir adet cpp dosyası ve en son main.cpp adında dosyalar oluşturarak kodu yazmaya başladık.

```
#pragma once
#include<iostream>
#include<fstream>
#include<string>
#include<iomanip>

using namespace std;
class telefonKayit
{private:
    string ad;
    string soyad;
    unsigned long long int isTel;
    unsigned long long int cepTel;
    static int sayac;
    int ID;

public:
    void kisiEkle();
    void setIsim();
    void setSoyad();
    void setCepTel();
    void setIsTel();
    string getAd();
    string getSoyad();
    unsigned long long int getİsTel();
    unsigned long long int getCepTel();
    int getID();
};
```

telefonKayit.h

Kodu yazarken öncelikle bize gerekli olan değişkenler tanımlandı. Gerekli olan kütüphaneler header dosyasına eklendikten sonra telefonKayit isminde bir sınıf oluşturuldu. Private olarak kullanıcıdan alınan ad, soyad, cep telefonu, iş telefonu tanımlandı. Sayac static int olarak tanımlandıktan sonra ID değişkeni de eklendi. Public olarak da yapılacak işlemlerde değişkenleri tanımlamak için kişi ekle adlı fonksiyon yaratıldı. Aynı şekilde private olarak tanımlanan verileri almak için set ve get fonksiyonları tanımlandı.

telefonKayit sınıfının UML tablosu aşağıda gösterildi.



telefonKayit.cpp

```cpp
#include<iostream>
#include<fstream>
#include<iomanip>
#include"telefonKayit.h"
#pragma once

using namespace std;
int telefonKayit::sayac = 0;

void telefonKayit::setIsim() {
    cout << "Isim:" << endl;
    cin >> ad;
}
void telefonKayit::setSoyad() {
    cout << "Soyisim:" << endl;
    cin >> soyad;
}
void telefonKayit::setCepTel() {
    cout << "Cep telefonu:" << endl;
    cin >> cepTel;
}
void telefonKayit::setIsTel() {
    cout << "Is telefonu:" << endl;
    cin >> isTel;
}
```

telefonKayit.cpp dosyasında, istenilen değişkenleri kullanabilmek için gerekli olan kütüphaneler tanımlandıktan sonra telefonKayit.h dosyası da tanımlandı. Private olarak tanımlanmış değişkenleri kullanıcıdan almak için set fonksiyonlarında kullanıcının giriş yapması sağlanıldı.

```
void telefonKayit::kisiEkle() {
    system("cls");
    setIsim();
    setSoyad();
    setCepTel();
    setIsTel();
    sayac++;

    ID = sayac;
}

string telefonKayit::getAd() {
    return ad;
}
string telefonKayit::getSoyad() {
    return soyad;
}
unsigned long long int telefonKayit::getCepTel() {
    return cepTel;
}
unsigned long long int telefonKayit::getİsTel() {
    return isTel;
}
int telefonKayit::getID() {
    return ID;
}
```

Get fonksiyonları ile kullanıcının girdiği veriler alındı. Kişi ekle fonksiyonu ile, girilen kişi bilgileri orada tanımlandı. Girilen her kişi için, girdiği sıra numarası, ID numarasına atandı.Ve her girişte sayac 1 arttırıldı.

Main.cpp

```
#include<iostream>
#include<fstream>
#include<iomanip>
#include<vector>
#include <algorithm>
#include"telefonKayit.h"
#pragma once
using namespace std;
int main() {
    telefonKayit tel[100];
    fstream dosya;
    static int kisiSayisi = 0;
    char ch;
    int num;
```

Gerekli olan kütüphaneler tanımlandıktan sonra telefonKayit sınıfında kayıt oluşturabilecek 100 kişilik bir obje dizisi yaratıldı. Farklı komutlarda yapılan işlemleri dosyaya kaydetmek için telefonRehberi adlı txt dosyası oluşturuldu. Kişi sayısı static olarak tanımlandıktan sonra gerekli olan argümanlar da eklendi.

```
do
{
    system("cls");
    cout << "Yapmak istediginiz islemi seciniz!" << endl;
    cout << endl;
    cout << "******" << endl;
    cout << "1. Kayit Ekleme" << endl;
    cout << "2. Kayitlari Gosterme" << endl;
    cout << "3. Kayit Arama" << endl;;
    cout << "4. Kayit Modifiye Etme" << endl;;
    cout << "5. Kayit Silme" << endl;;
    cout << "6. Cikis" << endl;;
    cout << "******" << endl;
    cin >> ch;
    system("cls");
```

Kodun do-while ile yapılması seçildi. Öncelikle yaratılan 6 kategoriden, kullanıcıdan yapmasını istediği komutu seçilmesi istendi.

```
switch (ch)
{
case '1': {
    tel[kisiSayisi].kisiEkle();
    cout << "\n\nKayit olusturuldu.Ana ekrana donmek icin herhangi bir tusa basiniz.";
    cin.ignore();
    cin.get();
    kisiSayisi++;
    break; }
```

Her bir durumun farklı bir değer olduğu bir değer listesi hazırlamak için switch-case yaklaşımını kullanılması tercih edildi. Girilen değişken durumlardan birine eşit olduğunda, durumu izleyen ifadeler yürütülür.
Kullanıcının girişi 1 olursa, kişi ekle fonksiyonunda yaratılan ad, soyad, cep telefonu, iş telefonu değişkenlerini kullanıcıdan alındı. Son olarak kişi sayısı girilen kişi kadar arttırıldı. İşlem bittikten sonra durumdan çıkmak için break komutu kullanıldı.

```
case '2': {
    vector <string> deneme;
    string t;
    string yedek2;
    for (int i = 0; i < kisiSayisi; i++) {
        t = tel[i].getAd();
        for (int j = 0; j < t.size(); j++) {
            t[j] = tolower(t[j]);
        }
        deneme.push_back(t);
    }
    sort(deneme.begin(), deneme.end());
```

Kullanıcın girişi 2 olursa rehberdeki tüm kişilerin bilgileri isim sırasına göre çıktı alındı. Tüm kullanıcıların ad fonksiyonu geçici bie stringine atanarak bütün harfleri

küçük harf haline getirilerek vektöre atanmıştır. Atanan vektörde sort fonksiyonu sayesinde alfabetik sıraya göre dizilmiştir.

```
for (int i = 0; i < kisiSayisi; i++) {
    int top = 0;
    for (int i2 = 0; i2 < deneme[i].size(); i2++) {
        top = top + (int)deneme[i].at(i2);
    }
    for (int j = 0; j < kisiSayisi; j++) {
        int top2 = 0;
        yedek2 = tel[j].getAd();
        for (int j1 = 0; j1 < yedek2.size(); j1++) {
            top2 = top2 + (int)yedek2.at(j1);
        }
```

Burada da deneme vektöründeki tüm adların harfleri tek tek ascii karşılıkları bulunarak toplandı.

```
        if (abs(top - top2) % 32 == 0) {
            cout << "****" << endl;
            cout << "ID:" << tel[j].getID() << endl;
            cout << "Isim:" << tel[j].getAd() << endl;
            cout << "Soyisim:" << tel[j].getSoyad() << endl;
            cout << "Cep Telefonu:" << tel[j].getCepTel() << endl;
            cout << "Is Telefonu:" << tel[j].getİsTel() << endl;
            cout << "****" << endl;
            cout << endl;

        }
    }
}
cout << "\n\nKayit gosterildi.Ana ekrana donmek icin herhangi bir tusa basiniz.";
cin.ignore();
cin.get();
break; }
```

Aranan stringin harflerinin ascii karakter toplamlarıyla, bulunması istenen stringin harflerinin ascii karakter toplamlarının, farkının mutlak değeri alındı. Bu farkın, 32'ye göre modunun alınıp sıfıra eşitlenmesinin sebebi büyük harf -küçük harf duyarlılığını sağlanmasıdır.

```
case '3': {aramaM:
    cout << "\n\tArama turunu seciniz" << endl;
    cout << "1.ID ile arama" << endl << "2.Isimle arama" << endl << "3.Soyadla arama" << endl << "4.Cep telefonuyla arama" << endl
        << "5.Is telefonuyla arama" << endl << "6.Ana menuye geri don" << endl;
    cin >> num;
    switch (num) {
    case(1):
    {
    IDAM:
        int aranacakID;

        cout << "Aramak istediginiz ID'yi giriniz:" << endl;
        cin >> aranacakID;
        int i;
        for (i = 0; i < kisiSayisi; i++) {
            if (aranacakID == tel[i].getID()) {
                cout << "ID:" << tel[i].getID() << endl;
                cout << "Isim:" << tel[i].getAd() << endl;
                cout << "Soyisim:" << tel[i].getSoyad() << endl;
                cout << "Cep Telefonu:" << tel[i].getCepTel() << endl;
                cout << "Is Telefonu:" << tel[i].getİsTel() << endl;
                cout << "\n\nAranan kisi bulundu.";

                break;
            }
        }
```

3.durum kullanıcının rehberden birini araması için tasarlanmıştır. Öncelikle aranacak kişinin hangi bilgisine göre arayacağını 6 tane komut belirleyerek tekrar switch-case yaklaşımını kullanmak için kullanıcının bunlardan birini seçmesi istendi.Bu komutlar şu şekildedir.
1.ID ile arama
2.İsimle arama
3.Soyadla arama
4.Cep telefonuyla arama
5.İş telefonuyla arama
6.Ana menüye geri dön

Eğer 1’e basıldıysa aranacakID adlı değişken yaratarak, kullanıcının aradığı kişinin ID numarasına eşitlendi. Telefon rehberindeki kişilerin hepsinin ID‘si, aranacakID ‘ye eşit mi diye taratıldı.Eğer eşitse bu kişinin tüm bilgileri ekrana yazdırıldı.

```
    if (i == kisiSayisi) {
    hata1:
        int i1;
        cout << "\aAranan ID bulunamadi" << endl << "1.Ana menuye geri donun" << endl << "2.Arama menusune geri donun" << endl
            << "3.ID arama menusune geri donun" << endl << "4.Cikis" << endl;
        cin >> i1;
        if (i1 == 1) {
            break;
        }
        else if (i1 == 2) {
            goto aramaM;
        }
        else if (i1 == 3) {
            goto IDAM;
        }
        else if (i1 == 4) {
            goto cikis;
        }
        else {
            cout << "\aHatali giris, tekrar deneyiniz!" << endl;
            cin.ignore();
            cin.get();
            goto hata1;
        }
    }

    else {
        cout << "Ana ekrana geri donmek icin herhangi bir tusa basiniz." << endl;
        cin.ignore();
        cin.get();
        break;
    }
}
```

Eğer öyle biri yoksa ekrana aranan kişi bulunamadı yazdırılıp, kullanıcının seçmesi istenen yeni komutlar yaratıldı. Kullanıcının girdiği sayıya göre o duruma gidilecek ve o durumdaki işlemler yapılacaktır.

```
case(2): {AAM:
string aranacakAd;
string olanAd;
cout << "Aramak istediginiz ismi giriniz:" << endl;
cin >> aranacakAd;
int aranacakAdTop = 0;
int olanAdTop = 0;
for (int j = 0; j < aranacakAd.size(); j++) {
    aranacakAdTop = aranacakAdTop + (int)aranacakAd.at(j);
}
int i;
for (i = 0; i < kisiSayisi; i++) {
    olanAd = tel[i].getAd();
    for (int j = 0; j < olanAd.size(); j++) {
        olanAdTop = olanAdTop + (int)olanAd.at(j);
    }
    if (abs(olanAdTop - aranacakAdTop) % 32 == 0) {
        cout << "ID:" << tel[i].getID() << endl;
        cout << "Isim:" << tel[i].getAd() << endl;
        cout << "Soyisim:" << tel[i].getSoyad() << endl;
        cout << "Cep Telefonu:" << tel[i].getCepTel() << endl;
        cout << "Is Telefonu:" << tel[i].getİsTel() << endl;
        cout << "\n\nAranan kisi bulundu.";

        break;
    }
    olanAdTop = 0;
}
```

Kullanıcı isime göre arama yapacaksa, aranacakAd adlı değişken oluşturularak girilen isim aranacakAd'a eşitlendi. Telefon rehberindeki tüm isimler taranarak bu kişinin rehberde olup olmadığı kontrol edildi. Burada da benzer bir yaklaşımla büyük

harf küçük harf duyarlılığın olması önlendi. Eğer bu kişi varsa tüm bilgileri ile ekrana yazdırıldı.

```
if (i == kisiSayisi) {
hata2:
    int i1;
    cout << "\aAranan ad bulunamadi" << endl << "1.Ana menuye geri donun" << endl << "2.Arama menusune geri donun" << endl
        << "3.Ad arama menusune geri donun" << endl << "4.Cikis" << endl;
    cin >> i1;
    if (i1 == 1) {
        break;
    }
    else if (i1 == 2) {
        goto aramaM;
    }
    else if (i1 == 3) {
        goto AAM;
    }
    else if (i1 == 4) {
        goto cikis;
    }
    else {
        cout << "\aHatali giris, tekrar deneyiniz!" << endl;
        cin.ignore();
        cin.get();
        goto hata2;
    }

}
else {
    cout << "Ana ekrana geri donmek icin herhangi bir tusa basiniz." << endl;
    cin.ignore();
    cin.get();
    break;
}
```

Eğer öyle biri yoksa ekrana aranan kişi bulunamadı yazdırılıp, kullanıcının seçmesi istenen yeni komutlar yaratıldı. Kullanıcının girdiği sayıya göre o duruma gidilecek ve o durumdaki işlemler yapılacaktır.

```
    case(3): {SAM:
    string aranacakSoyad;
    string olanSoyad;
    cout << "Aramak istediginiz soyadi giriniz:" << endl;
    cin >> aranacakSoyad;
    int aranacakSoyadTop = 0;
    int olanSoyadTop = 0;
    for (int j = 0; j < aranacakSoyad.size(); j++) {
        aranacakSoyadTop = aranacakSoyadTop + (int)aranacakSoyad.at(j);
    }
    int i;
    for (i = 0; i < kisiSayisi; i++) {
        olanSoyad = tel[i].getSoyad();
        for (int j = 0; j < olanSoyad.size(); j++) {
            olanSoyadTop = olanSoyadTop + (int)olanSoyad.at(j);
        }
        if (abs(olanSoyadTop - aranacakSoyadTop) % 32 == 0) {
            cout << "ID:" << tel[i].getID() << endl;
            cout << "Isim:" << tel[i].getAd() << endl;
            cout << "Soyisim:" << tel[i].getSoyad() << endl;
            cout << "Cep Telefonu:" << tel[i].getCepTel() << endl;
            cout << "Is Telefonu:" << tel[i].getİsTel() << endl;
            cout << "\n\nAranan kisi bulundu.";

            break;
        }
        olanSoyadTop = 0;
    }
```

Kullanıcı soyada göre arama yapacaksa, aranacakSoyad adlı değişken oluşturularak girilen soyad aranacakSoyad'a eşitlendi. Telefon rehberindeki tüm soyadlar taranarak bu kişinin rehberde olup olmadığı büyük harf küçük harf duyarlılığı olmadan kontrol edildi. Eğer bu kişi varsa tüm bilgileri ile ekrana yazdırıldı.

```
    if (i == kisiSayisi) {
    hata3:
        int i1;
        cout << "\aAranan soyad bulunamadi" << endl << "1.Ana menuye geri donun" << endl << "2.Arama menusune geri donun" << endl
            << "3.Soyad arama menusune geri donun" << endl << "4.Cikis" << endl;
        cin >> i1;
        if (i1 == 1) {
            break;
        }
        else if (i1 == 2) {
            goto aramaM;
        }
        else if (i1 == 3) {
            goto SAM;
        }
        else if (i1 == 4) {
            goto cikis;
        }
        else {
            cout << "\aHatali giris, tekrar deneyiniz!" << endl;
            cin.ignore();
            cin.get();
            goto hata3;
        }

    }
    else {
        cout << "Ana ekrana geri donmek icin herhangi bir tusa basiniz." << endl;
        cin.ignore();
        cin.get();
        break;
```

Eğer öyle biri yoksa ekrana aranan kişi bulunamadı yazdırıp, kullanıcının seçmesini istediğimiz yeni komutlar yarattık. Kullanıcının girdiği sayıya göre o duruma gidilecek ve o durumdaki işlemler yapılacaktır.

```
case(4): {CTAM:
int aranacakCepTel;
cout << "Aramak istediginiz cep telefonunu giriniz:" << endl;
cin >> aranacakCepTel;
int i;
for (i = 0; i < kisiSayisi; i++) {
    if (aranacakCepTel == tel[i].getCepTel()) {
        cout << "ID:" << tel[i].getID() << endl;
        cout << "Isim:" << tel[i].getAd() << endl;
        cout << "Soyisim:" << tel[i].getSoyad() << endl;
        cout << "Cep Telefonu:" << tel[i].getCepTel() << endl;
        cout << "Is Telefonu:" << tel[i].getİsTel() << endl;
        cout << "\n\nAranan kisi bulundu.";

        break;
    }
}
```

Kullanıcı cep telefonuna göre arama yapacaksa, aranacakcepTel adlı değişken oluşturularak girilen numara aranacakcepTel'e eşitlendi. Telefon rehberindeki tüm numaralar taranarak bu kişinin rehberde olup olmadığı kontrol edildi. Eğer bu kişi varsa tüm bilgileri ile ekrana yazdırıldı.

```
if (i == kisiSayisi) {
hata4:
    int i1;
    cout << "\aAranan cep telefonu bulunamadi" << endl << "1.Ana menuye geri donun" << endl << "2.Arama menusune geri donun" << endl
        << "3.Cep telefonu arama menusune geri donun" << endl << "4.Cikis" << endl;
    cin >> i1;
    if (i1 == 1) {
        break;
    }
    else if (i1 == 2) {
        goto aramaM;
    }
    else if (i1 == 3) {
        goto CTAM;
    }
    else if (i1 == 4) {
        goto cikis;
    }
    else {
        cout << "\aHatali giris, tekrar deneyiniz!" << endl;
        cin.ignore();
        cin.get();
        goto hata4;
    }

}
else {
    cout << "Ana ekrana geri donmek icin herhangi bir tusa basiniz." << endl;
    cin.ignore();
    cin.get();
    break;
}
}
```

Eğer öyle biri yoksa ekrana aranan kişi bulunamadı yazdırılıp, kullanıcının seçmesi istenen yeni komutlar yaratıldı. Kullanıcının girdiği sayıya göre o duruma gidilecek ve o durumdaki işlemler yapılacaktır.

```
case(5): {ITAM:
int aranacakIsTel;
cout << "Aramak istediginiz is telefonunu giriniz:" << endl;
cin >> aranacakIsTel;
int i;
for (i = 0; i < kisiSayisi; i++) {
    if (aranacakIsTel == tel[i].getİsTel()) {
        cout << "ID:" << tel[i].getID() << endl;
        cout << "Isim:" << tel[i].getAd() << endl;
        cout << "Soyisim:" << tel[i].getSoyad() << endl;
        cout << "Cep Telefonu:" << tel[i].getCepTel() << endl;
        cout << "Is Telefonu:" << tel[i].getİsTel() << endl;
        cout << "\n\nAranan kisi bulundu.";

        break;
    }
}
```

Kullanıcı iş telefonuna göre arama yapacaksa, aranacakIsTel adlı değişken oluşturularak girilen numara aranacakIsTel'e eşitlendi. Telefon rehberindeki tüm numaralar taranarak bu kişinin rehberde olup olmadığı kontrol edildi. Eğer bu kişi varsa tüm bilgileri ile ekrana yazdırıldı.

```
if (i == kisiSayisi) {
hata5:
    int i1;
    cout << "\aAranan is telefonu bulunamadi" << endl << "1.Ana menuye geri donun" << endl << "2.Arama menusune geri donun" << endl
        << "3.İs telefonu arama menusune geri donun" << endl << "4.Cikis" << endl;
    cin >> i1;
    if (i1 == 1) {
        break;
    }
    else if (i1 == 2) {
        goto aramaM;
    }
    else if (i1 == 3) {
        goto ITAM;
    }
    else if (i1 == 4) {
        goto cikis;
    }
    else {
        cout << "\aHatali giris, tekrar deneyiniz!" << endl;
        cin.ignore();
        cin.get();
        goto hata5;
    }
}

else {
    cout << "Ana ekrana geri donmek icin herhangi bir tusa basiniz." << endl;
    cin.ignore();
    cin.get();
    break;
}
```

Eğer öyle biri yoksa ekrana aranan kişi bulunamadı yazdırılıp, kullanıcının seçmesi istenen yeni komutlar yaratıldı. Kullanıcının girdiği sayıya göre o duruma gidilecek ve o durumdaki işlemler yapılacaktır.

```
    case(6): {
        break;
    }

    }

    break;
    }
```

Kullanıcı bu komutla ana menüye geri dönmektedir.

```
case '4': {M:
    int ID2, i;
    cout << "Modifiye edilecek ID'i giriniz:" << endl;
    cin >> ID2;
    char s1, s2, s3, s4;
    int uyari = 0;
    for (i = 0; i < kisiSayisi; i++) {
        int IDvar = tel[i].getID();
        if (IDvar == ID2) {
            uyari = 1;
            break;
        }
    }
```

4. durum, kullanıcının telefon rehberinde birini ID'si ile modifiye etmek için tasarlanmıştır. Kullanıcı modifiye etmek istediği kişinin ID'sini girecek ve bu ID; ID2'ye eşitlenecektir. Burada rehberdeki tüm kişilerin ID'si taranarak ,ID2'ye eşit olup olmadığı kontrol edilmektedir.

```
if (uyari) {
Name:
    cout << "Isim modifiye edilecek mi?(E/H)" << endl;
    cin >> s1;
    if (s1 == 'e' || s1 == 'E') {
        cout << "Yeni ";
        tel[i].setIsim();
    }
    else if (s1 == 'h' || s1 == 'H') {

    }
    else {
        cout << "\aHatali giris,tekrar deneyiniz!" << endl;
        goto Name;
    }
```

Modifiye edilecek kişi bulunduktan sonra, kişinin hangi bilgisinin modifiye edileceği kullanıcıya sorulur. Burada kişinin ismi modifiye edilecek mi diye sorulup evet ise yeni isim alınacak, hayır ise işlem yapılmadan devam edilecektir. Cevaba uygun olmayan girişlerde hata çıktısı verilecektir.

```
Surname:
    cout << "Soyad modifiye edilecek mi?(E/H)" << endl;
    cin >> s2;
    if (s2 == 'e' || s2 == 'E') {
        cout << "Yeni ";
        tel[i].setSoyad();
    }
    else if (s2 == 'h' || s2 == 'H') {

    }
    else {
        cout << "\aHatali giris,tekrar deneyiniz!" << endl;
        goto Surname;
    }
```

Bu kodda kişinin soyadı modifiye edilecek mi diye sorulup evet ise yeni soyad alınacak, hayır ise işlem yapılmadan devam edilecektir. Cevaba uygun olmayan girişlerde hata çıktısı verilecektir.

```
Phone1:
    cout << "Cep telefonu modifiye edilecek mi?(E/H)" << endl;
    cin >> s3;
    if (s3 == 'e' || s3 == 'E') {
        cout << "Yeni ";
        tel[i].setCepTel();
    }
    else if (s3 == 'h' || s3 == 'H') {

    }
    else {
        cout << "\aHatali giris,tekrar deneyiniz!" << endl;
        goto Phone1;
    }
```

Burada kişinin cep telefonu modifiye edilecek mi diye sorulup evet ise yeni numara alınacak, hayır ise işlem yapılmadan devam edilecektir. Cevaba uygun olmayan girişlerde hata çıktısı verilecektir.

```
Phone2:
    cout << "Is telefonu modifiye edilecek mi?(E/H)" << endl;
    cin >> s4;
    if (s4 == 'e' || s4 == 'E') {
        cout << "Yeni ";
        tel[i].setIsTel();
    }
    else if (s4 == 'h' || s4 == 'H') {

    }
    else {
        cout << "\aHatali giris,tekrar deneyiniz!" << endl;
        goto Phone2;
    }
    uyari = 0;
}
```

Burada kişinin iş telefonu modifiye edilecek mi diye sorulup evet ise yeni numara alınacak, hayır ise işlem yapılmadan devam edilecektir. Cevaba uygun olmayan girişlerde hata çıktısı verilecektir.

```
if ((ID2 > kisiSayisi) || (i == kisiSayisi)) {
    cout << "\aHatali ID girisi,tekrar deneyiniz." << endl;
    cin.ignore();
    cin.get();
    goto M;
}
else {
    cout << "Kisi modifiye edilmistir!" << endl;
    cin.ignore();
    cin.get();

    break;

}

break; }
```

Eğer kullanıcının girdiği ID yanlışsa, modifiye etme kısmına geri dönüp en başa alınması sağlanmıştır. İşlem tamamlandıktan sonra ekrana ‘Kişi modifiye edilmiştir’ yazdırılmıştır.

```
case '5': {
Sil:
    cout << "\n\tSilinecek ID: "; cin >> num;
    int uyarı = 0;
    int IDal;
    for (int k = 0; k < kisiSayisi; k++) {
        IDal = tel[k].getID();
        if (IDal == num) {
            uyarı = 1;
        }
    }
    if (uyarı) {

        for (int i = num - 1; i < kisiSayisi; i++) {
            tel[i] = tel[i + 1];
        }
        cout << "Kisi silinmistir!" << endl;

        cin.ignore();
        cin.get();
    kisiSayisi--;

    }
    else {
        system("cls");
        cout << "\aHatali ID girisi,tekrar deneyiniz!" << endl;
        goto Sil;
    }
    break;
}
```

5. durum, kullanıcının telefon rehberinde birini ID'si ile silmek için tasarlanmıştır. Kullanıcı silmek istediği kişinin ID'sini girecek ve bu ID;
IDal 'a eşitlenecektir. Burada rehberdeki tüm kişilerin ID'si taranarak ,IDal'a eşit olup olmadığı kontrol edilmektedir. Eğer böyle biri varsa, rehberden silinecek ve kişi sayısı 1 azaltılacaktır. Eğer böyle biri yoksa hata çıktısı verilip 5.duruma geri dönülecektir.

```
        case '6': { cikis:
        dosya.open("telefonRehberi.txt");
        if (dosya.is_open()) {
            for (int i = 0; i < kisiSayisi; i++) {
                dosya << "****" << endl;
                dosya << "ID:" << tel[i].getID() << endl;
                dosya << "Isim:" << tel[i].getAd() << endl;
                dosya << "Soyisim:" << tel[i].getSoyad() << endl;
                dosya << "Cep Telefonu:" << tel[i].getCepTel() << endl;
                dosya << "Is Telefonu:" << tel[i].getİsTel() << endl;
                dosya << "****" << endl;
            }
        }
        else
            cout << "Dosya acilamadi" << endl;
        dosya.close();

        exit(0); }
        default:    cout << "\a";

        }
    } while (ch != '7');

    getchar();
    return 0;
}
```

6.durum kullanıcının bu tasarımdan çıkması için tasarlanmıştır. Kullanıcı bu tasarımdan çıkmadan önce yapılan tüm girişleri dosyaya kaydedecek ve sonra dosya kapatılacaktır. Sistemde olası hatalı durumlarda \a komutuyla alarm sesi vermesi sağlanmıştır.

3 Sonuçlar

Telefon Rehberi Kayıt Sistemi projesinde Switch- case komutunu yapı taşı olarak seçip tasarıma öyle devam ettik. Yapılan işlemleri dosyaya kaydetmeyi bu proje ile deneyimledik. Yapılan projede bilgi birikimimizi göstermek ve projeyi kalitelileştirmek adına birçok yaklaşım kullanmayı tercih ettik. Vektör kütüphanesi gibi ve default, break, goto gibi fonksiyonları yaklaşımları kullandık.
Ayrıca yazılan her bir bölümde düşünmenin, uğraşıp geliştirmenin, hata ayıklama konusunda deneyimler kazandık.

Hazırlanan sunum video'su adresi: https://www.youtube.com/watch?v=5ZdfOf9IghM

Dosyaların github adresi: https://github.com/iremkalkanli/BLM-102-Projesi-Telefon-Kayit-Sistemi

İrem **Kalkanlı**

Fenerbahçe Üniversitesi · Bilgisayar Mühendisliği · Öğrenci (1.Sınıf)
Atatürk, Metropol İstanbul, Ataşehir Blv., 34758 Ataşehir/İstanbul
(+90) 536-509-8787 | irem.kalkanli@stu.fbu.edu.tr | iremkalkanli | İrem Kalkanlı | İrem Kalkanlı

İş Deneyimi

Eğitim

Fenerbahçe Üniversitesi — *İstanbul, Türkiye*
Bilgisayar Mühendisliği — *Ekim, 2019 - Bek. Mayıs, 2023*
- Fenerbahçe Üniversitesi bilgisayar mühendisliği lisans programında öğrenci(GNO:2.8)

Özel Batı Ataşehir Doğa Bilim Anadolu Lisesi — *İstanbul, Türkiye*
Lise — *Eylül, 2014 - Haziran, 2018*
- Diploma Notu:95.52

Atakent Doğa Koleji — *İstanbul, Türkiye*
Ortaokul — *Eylül, 2012 - Haziran 2014*

Üsküdar Doğa Koleji — *İstanbul, Türkiye*
İlkokul — *Eylül, 2006 - Haziran, 2011*

Yetenekler

Programlama Dilleri	C,C++
Microsoft Office	World, Excel, Power Point
Yazılım	Linux, Windows
Yabancı Diller	İnglizce, Almanca, Çince

Sertifikalar ve Seminerler

2020	**Linux 101 ve Sistem Yönetimi**, Eğitim Sertifakası	*İstanbul, Türkiye*
2020	**CISCO Networking Academy:Siber Güvenliğe Giriş**, Eğitim Sertifakası	*İstanbul, Türkiye*
2020	**CISCO Networking Academy:WiFi Kablosuz Ağlar**, Eğitim Sertifakası	*İstanbul, Türkiye*
2018	**Mindfulness 101**, Katılım sertifikası	*İstanbul, Türkiye*
2017	**Adli Bilimler**, Seminer	*İstanbul, Türkiye*
2014-2017	**Master Business Administration for Teenagers (T-MBA)**, Eğitim sertifikası	*İstanbul, Türkiye*
2016	**First Certificate in English (FCE)**, Not:C, University of Cambridge ESOL Examinations	*İstanbul, Türkiye*
2014	**FİT In Deutsch 1**, Not:47/60,Goethe-Zertifikat A1	*İstanbul, Türkiye*

Projeler

BLM 101 Projesi: FBU CPU — *İstanbul, Türkiye*
Aysen İpek Çakır, Deniz Uzun ve Özlem Çalı ile gerçekleştirilmiştir. — *Ocak, 2020*
- On farklı operasyonu gerçekleştiren bir işlemci tasarımı
- Detaylı bilgi için: BLM-101-Projesi-FBU-CPU

BLM 103 Projesi: Sezar Şifreleme — *İstanbul, Türkiye*
Aysen İpek Çakır ile gerçekleştirilmiştir — *Ocak, 2020*
- C dilinde yazılmış bir sezar şifreleme ve deşifreleme kodudur.
- Detaylı bilgi için: BLM-103-Sezar-Sifreleme

BLM 103 Projesi: Sezar Şifreleme

İstanbul, Türkiye

Aysen İpek Çakır, Deniz Uzun ve Özlem Çalı ile gerçekleştirilmiştir. *Haziran, 2020*

- C++ dilinde yazılmış kayıt ekleme, modifiye etme, arama, alfabetik olarak sıralama ve silme işlemlerini gerçekleştirebilen telefon kayıt sistemidir.
- Detaylı bilgi için: BLM-102-Projesi-Telefon-Kayit-Sistemi

ÖZLEM ÇALI

Bilgisayar Mühendisi

İstanbul/Ataşehir
0531 434 3642
ozlem.cali@stu.fbu.edu.tr
özlem-çalı-1222171b0
github.com/ozlem.cali
LMS:ozlem.cali

HAKKIMDA

Ben Özlem. Fenerbahçe Üniversitesi Bilgisayar Mühendisliği 1.sınıf öğrencisiyim.

C++
C
Python

SERTİFİKALAR

2012-2014	**Türkiye Satranç Federasyonu**	**Katılım Sertifikası**
2012-2018	**Türkiye Voleybol Federasyonu**	**Katılım Sertifikası**
2018-2019	**Aykut Kence Evrim Konferansı**	**Eğitim Sertifikası**

EĞİTİM

2007-2011	**Koçören Cemalettin Tınaztepe İlköğretim Okulu**	**İlkokul**
2011-2014	**Koçören Cemalettin Tınaztepe Ortaoğretim Okulu**	**Ortaokul**
2014-2018	**Necmi Asfuroğlu Anadolu Lisesi**	**Lise**

PROJELER

YAPAY ZEKA İLE KANSER TESPİTİ **kNN algoritması ile, UC Irvine Üniversitesi veritabanından alınmış göğüs kanser verileri işlenerek, yeni bir kişininverileri sisteme yüklenerek hasta olup olmadığı tahmin edilen bir projedir.** **Deniz Uzun ile tasarlanan bir projedir.**

FBU-CPU PROJESİ **Farklı operasyonları gerçekleştiren işlemci tasarlanmıştır.** **Deniz Uzun, Aysen İpek Çakır, İrem Kalkanı ile yapılan bir projedir.**

TELEFON KAYIT SİSTEMİ **Bir telefon rehberinde bulunması gereken kayıt ekleme, kayıtları gösterme, kayıtları modifiye etme, kayıt arama ve silme kabiliyetlerine sahip olan bir sistem geliştirilmiştir.** **Deniz Uzun, Aysen İpek Çakır, İrem Kalkanı ile yapılan bir projedir.**

YABANCI DİL

İngilizce - Orta Düzey
Almanca - Düşük Düzey
ARAPÇA - Orta Düzey

Aysen İpek Çakır

Feenerbahçe Üniversitesi Bilgisayar Mühendisliği Öğrencisi

05412050361 | aysen.cakir@stu.fbu.edu.tr | aysencakir | aysen-ipek-çakır-a9b7141a4

Eğitim

Fenerbahçe Üniversitesi — *İstanbul,Türkiye*
Bilgisayar Mühendisliği — *Ekim 2019 - Mayıs 2023*

Fetgem Fen Lisesi — *Malatya,Türkiye*
Lise — *Eylül 2015 - Haziran 2019*

Begüm Kartal Ortaokulu — *Malatya,Türkiye*
Ortaokul — *Eylül 2012 - Haziran 2015*

Sümer İlkÖğretim Okulu — *Malatya,Türkiye*
İlkokul — *Eylül 2007 - Haziran 2012*

Yetenekler

Yazılım Dilleri Python, C/C++
Yabancı Diller İngilizce

Sertifikalar ve Seminerler

Malatya,Türkiye
2017 Tübitak Okul Finali — *Mart 2017*
- Katılımcı Sertifikası

Malatya,Türkiye
2018 Tübitak Bölge Finali — *Mart 2018*
- Katılımcı Sertifikası

İstanbul,Türkiye
BİLMÖK Bilgisayar Mühendisliği Öğrencileri Konfersansı — *Şubat 2020*
- Seminer

İstanbul,Türkiye
Siber Güvenlikte Kariyer — *Aralık 2019*
- Seminer

Projeler

Tübitak Biyoloji Projesi: Ananastaki Bromelain maddesinin kanser hücreleri üzerine etkisi

Malatya,Türkiye

Sena Dursun ile gerçekleştirilmiştir

Mart 2018

- Ananastaki Bromelain maddesinin kanser hücreleri ve sağlıklı hücreler üzerindeki etkisini test eden bir proje

BLM 101 Projesi:FBU-CPU

İstanbul,Türkiye

Deniz Uzun,İrem Kalkanlı,Özlem Çalı ile gerçekleştirilmiştir

Ocak 2020

- 10 farklı operasyonu gerçekleştiren bir işlemci tasarımı

BLM 103 Projesi:Sezar Şifreleme

İstanbul,Türkiye

İrem kalkanlı ile gerçekleştirilmiştir

Ocak 2020

- C dilinde yazılmış bir sezar şifreleme ve deşifreleme kodudur.

BLM 102 Projesi:Telefon Kayıt Sistemi

İstanbul,Türkiye

Deniz Uzun,İrem Kalkanlı,Özlem Çalı ile gerçekleştirilmiştir

Mayıs 2020

- C++ dilinde yazılmış bir telefon rehberinde bulunması gereken kayıt ekleme, kayıtları gösterme, kayıtları modifiye etme, kayıt arama ve silme kabiliyetlerine sahip olan bir sistem kodudur.

DENIZ UZUN

İstanbul/Beykoz
0531 280 1890
deniz.uzun@stu.fbu.edu.tr
/deniz-uzun-4b9bb91a1/
github.com/denzuzun
LMS: deniz.uzun

HAKKIMDA

Ben Deniz. Fenerbahçe Üniversitesi Bilgisayar Mühendisliği Bölümü 1.sınıf öğrencisiyim. Okulumuzun Siber Güvenlik Kulübünün başkanlığını yapmaktayım. Yakın zamanda da bir web sitede teknoloji yazarlığı yapmaya başladım.

EĞİTİM

2007 – 2009	**Paşabahçe İlköğretim Okulu** İlkokul
2009 – 2014	**T.E.B Ataşehir Ortaokulu** Ortaokul
2015 – 2017	**Celal Aras Anadolu Lisesi** Lise
2017 – 2019	**Kavacık Uğur Anadolu Lisesi** Lise
2019 – 2023	**Fenerbahçe Üniversitesi** Üniversite

SERTIFIKALAR

2019	**Bahçeşehir Üniversitesi** Siber Güvenlik Big Data Girişimcilik ve Liderlik Blockchain Oyun ve Oyunlaştırma
2020	**Siber Kulüpler Birliği** Linux 101 ve Sistem Yönetimi
2020	**CISCO** Introduction to Cybersecurity Mobility Fundamentals

YETENEKLER

- Linux
- C++
- C

YABANCI DİL

İngilizce

EKİP İLE YAPILAN PROJELER

-YAPAY ZEKA İLE KANSER TESPİTİ YAPAN ALGORİTMA (C)
-BLM101 FBU CPU PROJESİ
-TELEFON KAYIT SİSTEM (C++)