

# Nüfus Yönetim Sistemi

İrem Bozkurt, Pinar Sefer, Şevval Can, Zehra KOYUNCU

Fenerbahçe Üniversitesi

Endüstri Mühendisliği

İstanbul, Türkiye

e-mail: {irem.bozkurt, sevvai.can , zehra.koyuncu, pinar.sefer} @fbu.edu.tr,

## Özetçe

Nesne yönelimli programlama, Her işlevin nesneler olarak soyutlandığı bir programlama yaklaşımıdır. NYP destekleyen programlama dilleri yüksek seviye diller olarak adlandırılır. (**wikipedia nesneye yönelimli programlama, 2021**) Bu çalışmada, nesneye yönelimli programla mantığı kullanılmıştır ve bir nüfus yönetim uygulaması yapılmıştır. Nesneye yönelimli programlamanın özelliği olan dört ana başlık göz önünde bulundurulmuştur ve çözüm yolları bu başlıklara göre izlenmiştir. Bu başlıklar şu şekildedir.

- Soyutlama
- Kapsülleme
- Miras Alma
- Çok Biçimlilik

**Anahtar Kelimeler** — *Class, Decorator, Kalıtım, Tasarım kalıpları, Çok biçimlilik*

## Abstract

**Object oriented programming is a programming approach in** which each function is abstracted as objects. Programming languages that support NYP are called high level languages. (Wikipedia object oriented programming, 2021) In this study, the logic of object oriented programming has been used and a population management application has been made. Four main headings, which are characteristic of object oriented programming, have been taken into consideration and solution ways have been followed according to these titles. These headings are as follows..

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

**Keywords** — *Class, Decorator, Inheritance, Design patterns, PolymorphismGiriş*

Bir nüfus yönetim sistemi uygulamasıdır. Nüfus yönetim uygulamasında bulunması gereken verileri işler ve kullanır. Geliştirilen yazılımda kullanıcıdan alınan kimlik numarası kullanılarak Adı, Soyadı, Baba adı, Anne adı, Doğum yeri, Medeni durumu, Kan grubu, Kütük Şehir, Kütük İlçe , İkametgah Şehir , İkametgah İlçe bilgilerini kullanır ve kullanıcının istediği işlemi yapar.

## I. SİSTEM MİMARİSİ

Microsoft'un derleyicisi olan Visual Studio Community ve Pycharm kullanılmıştır.

## II. KULLANILAN YAZILIM

Kullanıcı programı ilk çalıştırdığında; karşısına seçim ekranı çıkar. Seçmek istediği komutun üzerine gelip enter tuşuna bastığı zaman komutun özelliğine göre kullanıcıyı yeni bir ekran karşılar. Bu seçimin yapılabilmesi için PyInquirer kütüphanesi kullanılmıştır.

```
Terminal: Local x +
(venv) C:\Users\irem\PycharmProjects\projeson>py dnm.py
Toplam 0 kayıt bulundu
? Seciniz (Use arrow keys)
> Çıkış
Yeni Kayıt Ekleme
Arama
Kişi Güncelleme
Kişi Silme
Tüm veritabanını listeleme
Terminal Python Console Q Find Run TODO
Occurrences found in comments, strings and non-code files
```

### Kayıt ekranı

Kullanıcı yeni kayıt ekleme işlemini seçerse karşısına yanda verilen arayüz çıkar.

Uygulama ilk başlatıldığında sisteme kayıtlı kaç kişi olduğu bilgisini ekrana basar.

Kullanıcının yapacağı bütün işlemler kimlik numarası üzerinden sağlanmaktadır. Kullanıcı kimlik numarasını girdikten sonra eğer yeni kayıt işlemi yapıyorsa ve sisteme kayıtlı bir kimlik numarası girmişse kimlik numarasının başkasına ait olduğuna dair bir hata mesajı alır. Yeni bir kimlik numarası giriyorsa kayıt işlemi için gerekli diğer bilgileri girmesi istenir. Arama, Kişi Güncelleme, Kişi Silme işlemlerinin tamamı kimlik numarasına göre yapılır.

Tüm Veritabanı Listeleme seçilirse kişiler.db uzantılı dosya içerisinde bulunan tüm bilgiler ekrana bastırılır.

```
Terminal: Local x +
(venv) C:\Users\irem\PycharmProjects\projeson>py dnm.py
Toplam 0 kayıt bulundu
? Seciniz Yeni Kayıt Ekleme
Kayıt edilecek kişinin kimlik numarasını giriniz:
```

```
Terminal: Local × +
Toplam 0 kayıt bulundu
? Seciniz Yeni Kayıt Ekleme
Kayıt edilecek kişinin kimlik numarasını giriniz:123
Kayıt edilecek kişinin ismini giriniz:ali
Kayıt edilecek kişinin soyadını giriniz:veli
Kayıt edilecek kişinin baba adını giriniz:ahmet
Kayıt edilecek kişinin anne adını giriniz:ayse
Kayıt edilecek kişinin doğum yerini giriniz:istanbul
Kayıt edilecek kişinin medeni durumunu giriniz:bekar
Kayıt edilecek kişinin kan grubunu giriniz:0+
Kayıt edilecek kişinin kütük şehri giriniz:istanbul
Kayıt edilecek kişinin kütük ilçesini giriniz:kadıköy
Kayıt edilecek kişinin ikametgah şehri giriniz:istanbul
Kayıt edilecek kişinin ikametgah ilçesini giriniz:kadıköy
? Seciniz (Use arrow keys)
> Çıkış
Yeni Kayıt Ekleme
Arama
Kişi Güncelleme
Kişi Silme
Tüm veritabanını listeleme
```

İlk olarak gerekli kütüphaneler çağrılmıştır.

Ardından Kayıt isimli bir class oluşturulmuştur. İstenilen tüm kişi bilgileri bu classta tanımlanmıştır. Daha sonra Classmethod kullanılmış ve yeni\_kayıt fonksiyonu oluşturularak kullanıcı bilgileri, kullanıcıdan input olarak alınmıştır ve bu bilgiler return edilmiştir.

```
from pathlib import Path
from functools import wraps
import PyInquirer

class Kayit(object):
    def __init__(self, kimlikNo, yeniİsim, yeniSoyisim, yeniBabaAdı, yeniAnneAdı, yeniDoğumYeri, yeniMedeniHal, yeniKanGrubu, yeniKütükŞehir, yeniKütükİlçe, yeniİkametgahŞehir, yeniİkametgahİlçe):
        self.kimlikNo = kimlikNo
        self.yeniİsim = yeniİsim
        self.yeniSoyisim = yeniSoyisim
        self.yeniBabaAdı = yeniBabaAdı
        self.yeniAnneAdı = yeniAnneAdı
        self.yeniDoğumYeri = yeniDoğumYeri
        self.yeniMedeniHal = yeniMedeniHal
        self.yeniKanGrubu = yeniKanGrubu
        self.yeniKütükŞehir = yeniKütükŞehir
        self.yeniKütükİlçe = yeniKütükİlçe
        self.yeniİkametgahŞehir = yeniİkametgahŞehir
        self.yeniİkametgahİlçe = yeniİkametgahİlçe

    @classmethod
    def yeni_kayıt(cls):
        kimlikNo = input("Kayıt edilecek kişinin kimlik numarasına giriniz:")
        yeniİsim = input("Kayıt edilecek kişinin ismini giriniz:")
        yeniSoyisim = input("Kayıt edilecek kişinin soyadına giriniz:")
        yeniBabaAdı = input("Kayıt edilecek kişinin baba adını giriniz:")
        yeniAnneAdı = input("Kayıt edilecek kişinin anne adını giriniz:")
        yeniDoğumYeri = input("Kayıt edilecek kişinin doğum yerini giriniz:")
        yeniMedeniHal = input("Kayıt edilecek kişinin medeni durumunu giriniz:")
        yeniKanGrubu = input("Kayıt edilecek kişinin kan grubunu giriniz:")
        yeniKütükŞehir = input("Kayıt edilecek kişinin kütük şehri giriniz:")
        yeniKütükİlçe = input("Kayıt edilecek kişinin kütük ilçesini giriniz:")
        yeniİkametgahŞehir = input("Kayıt edilecek kişinin ikametgah şehri giriniz:")
        yeniİkametgahİlçe = input("Kayıt edilecek kişinin ikametgah ilçesini giriniz:")

        return cls(
            kimlikNo,
            yeniİsim,
            yeniSoyisim,
            yeniBabaAdı,
            yeniAnneAdı,
            yeniDoğumYeri,
            yeniMedeniHal,
            yeniKanGrubu,
            yeniKütükŞehir,
            yeniKütükİlçe,
            yeniİkametgahŞehir,
            yeniİkametgahİlçe,
        )

    def __str__(self):
```

Nüfus yönetimi için nüfus yönetimi isminde bir class oluşturulmuştur.

Dosyadan oku  
Kayıt ekle  
Kişi silme  
Kişi güncelleme  
Listeleme  
Arama

Fonksiyonları bu class içerisinde tanımlanmıştır. Dosyadan oku fonksiyonunda dosya açma ve dosya işlemleri yapılmıştır. Diğer tüm fonksiyonlar ihtiyaç duyulan komutları yapacak şekilde düzenlenmiştir.

```
class NufusYonetimi(object):
    def __init__(self):
        self.dosyaAdi = "kisiler.db"
        self.kayitlar = self.dosyadan_oku()

    def dosyadan_oku(self):
        kayitlar = []
        if Path(self.dosyaAdi).exists():
            with open(self.dosyaAdi, "r") as dosya:
                for line in dosya.read().splitlines():
                    kayitlar.append(Kayit.dosyadan_oku(line))

        return kayitlar

    def kayit_ekle(self):
        yeni_kayit = Kayit.yeni_kayıt()
        for kayit in self.kayitlar:
            if yeni_kayit.kimlikNo == kayit.kimlikNo:
                print("Bu kimlik numarası başka bir kişiye aittir")
                return

        self.kayitlar.append(yeni_kayit)

    def kisi_silme(self, kimlik_no):
        for kayit in self.kayitlar:
            if kayit.kimlikNo == kimlik_no:
                self.kayitlar.remove(kayit)
                return

        print("Kimlik no {}'a sahip kayıt bulunamadı".format(kimlik_no))

    def kisi_guncelleme(self, kimlik_no):
        degisecek = self.degistirilmek_istenen()

NufusYonetimi > baslat()
Terminal Python Console

Program kodu yazılırken nesneye yönelik programlama özellikleri kullanılmıştır. Bu özelliklerden bir tanesi de decoratorlerdir. veritabanı_guncelle fonksiyonu decorator olarak yazılan bir fonksiyondur. decorator içinde tekrar decorator yani wrapper kullanılmıştır. Ayrıca veritabanı_guncelle fonksiyonunda dosya işlemleri yapılmıştır.
```

```
def veritabanini_guncelle(fonksiyon):
    @wraps(fonksiyon)
    def decorator(self, *args, **kwargs):
        fonksiyon(self, *args, **kwargs)

        with open(self.dosyaAdi, "w") as dosya:
            for kayit in self.kayitlar:
                dosya.write(kayit.virgulle_ayir() + "\n")

        return decorator

@veritabanini_guncelle
def secim_yap(self):
    secimler = [
        {
            "name": "secim",
            "type": "list",
            "message": "Seciniz",
            "choices": [
                "Çıkış",
                "Yeni Kayıt Ekleme",
                "Arama",
                "Kişi Güncelleme",
                "Kişi Silme",
                "Tüm veritabanını listeleme",
            ]
        }
    ]
    secim = PyInquirer.prompt(secimler).get("secim", "Çıkış")
```

Uygulama açıldığı zaman kullanıcının karşısına çıkan arayüzün arka planı bu şekildedir. Kullanıcının yaptığı seçime göre uygun method kullanılır.

En sonda nüfus yönetimi classı içerisinde bulunan başlat fonksiyonu çağrılır ve program başlar

```
if secim == "Çıkış":
    print("Çıkış seçildi")
    exit(0)

elif secim == "Yeni Kayıt Ekleme":
    self.kayit_ekle()

elif secim == "Arama":
    kimlikno = input("Kimlik no: ")
    self.arama(kimlikno)

elif secim == "Kişi Güncelleme":
    kimlikno = input("Kimlik no: ")
    self.kisi_guncelleme(kimlikno)

elif secim == "Kişi Silme":
    kimlikno = input("Kimlik no: ")
    self.kisi_silme(kimlikno)

elif secim == "Tüm veritabanını listeleme":
    self.veritabani_listele()

else:
    print("Hatalı giriş!!!")

def baslat(self):
    print("Toplam {} kayıt bulundu".format(len(self.kayitlar)))
    while True:
        self.secim_yap()

NufusYonetimi().baslat()
```

### III. SONUÇLAR

Bu projede Nesneye Yönelik Programlama kullanılmıştır. Bu kapsamda classlar, destructor'lar, çok biçimlik ve decoratorler gibi nesneye yönelimli programlama özellikleri kullanılmıştır. Nesneye yönelik programlama bağlamında bir nesne üretilmiştir. Daha sonra yapılacak değişikliklerde tüm kodda değişiklik yapmak yerine nesne üzerinde yapılacak değişikliklerin yeterli olacağı görülmüştür. Oluşturulan bu nesne classlar sayesinde daha sonra başka kodlarda da kullanılabilir hale getirilmiştir. Kod tekrarı yapılmadığı için verimlilik artmıştır.

### PROJE EKİBİ

#### Zehra Koyuncu:

21.02.2000 yılında İstanbul ili Esenler ilçesinde doğdu. 2018 yılında Bursa Erkek Lisesi'nden mezun oldu. Şu an Fenerbahçe Üniversitesi'nde okumaktadır. Endüstri mühendisliği bölümünde lisans eğitimi almakta. C ve Python dili ile ilgilenmektedir. 190302019

#### Şevval CAN:

24.09.1999 yılında Balıkesir ili Altıeylül ilçesinde doğdu. 2018 yılında Balıkesir Uğur Koleji'nden mezun oldu. Şu an Fenerbahçe Üniversitesi'nde Endüstri Mühendisliği bölümünde lisans eğitimi almaktadır. C, Python ve ML ile ilgilenmektedir. 190302028

#### Pınar SEFER

12.07.2001 yılında İstanbul ili Kadıköy ilçesinde doğdu. 2019 yılında Üsküdar Çağrıbey Anadolu Lisesi'nden mezun oldu. Şu anda Fenerbahçe üniversitesinde Endüstri Mühendisliği lisans eğitimi almaktadır. C ve Python ile ilgilenmektedir. 190302039

#### İrem BOZKURT:

04.11.1998 yılında Adıyaman ili Merkez ilçesinde doğdu. 2016 yılında Özel Bil Koleji Fen Lisesi'nden mezun oldu. Şu anda Fenerbahçe Üniversitesi'nde Endüstri Mühendisliği bölümünde anadal, Bilgisayar Mühendisliği bölümünde çiftanadal eğitimi almaktadır. C ve Python ile ilgilenmektedir. 190302010

### REFERANS DOSYALAR

Youtube: <https://www.youtube.com/watch?v=DCri9iLZkk4>

Github: <https://github.com/irembozkurt/Nufus-Yonetim-Sistemi>

### IV. KAYNAKÇA

*vikipedi nesneye yönelimli programlama.* (2021, 1 10).

vikipedi:

[https://tr.wikipedia.org/wiki/Nesne\\_y%C3%B6nelimli\\_programlama](https://tr.wikipedia.org/wiki/Nesne_y%C3%B6nelimli_programlama) adresinden alındı

[1]