

GitHub Linki:

<https://github.com/lanbis/Sinema-Rezervasyon>

Proje Tanıtım Videosu

https://youtu.be/TQoK8_RAsIo

Genel Bilgiler

Programın geliştirilme süresince programın mümkün olduğu kadar esnek bir şekilde kodlanmasına dikkat edilmiştir. indirim.txt ve imdb.txt dosyalarına müdahale edebilmek mümkündür ve yazı formatı bozulmadığı sürece programın hata verip çökme ihtimali çok düşüktür. Program, kullanıcının suçsuzluğu kanıtlanana kadar kullanıcı suçludur anlayışı ile kodlanmıştır. Dolayısı ile kullanıcı tarafından yapılacak tüm girişler ve işlemler kontrol edilip, program üzerinde oluşacak olan tüm açıklar engellenmeye çalışılmıştır. Proje dokümanında istenilen tüm özellikler fazlası ile yerine getirilmiş olup tüm kontrolleri sağlanmıştır ve ek opsiyonlar eklenmiştir.

Değişkenlerin ve Fonksiyonların Tanıtımı

Opsiyon olarak eklenen özelliklerin içerikleri “Opsiyonlar” bölümünde açıklanmaktadır.

Programda kullanılan kütüphaneleri main dosyasının en üstünde görebilirsiniz. Bu kütüphanelerin hangi amaç ile kullandıkları rapor içerisinde açıklanmıştır.

```
import datetime
import biletIslem
from datetime import date
from fpdf import FPDF
from time import sleep
import random
import webbrowser
from pyfiglet import Figlet
```

Program başlangıcında ilk olarak Salon sınıfından s1 nesnesi oluşturulmaktadır ve constructor içinden otomatik olarak salon_olustur() fonksiyonu çağırılmaktadır.

```
class Salon:
    def __init__(self):
        self.salon_olustur()
        self.imdb_oku()
        self.dosya_oku()
```

Bu fonksiyon, salon için gerekli olan değişkenlerin tanımlanması ve yapılacak atamalar içindir. Fonksiyon içerisinde “ciro[]”, “kalan_koltuk[]”, “matris[][]” değişkenleri tanımlanmıştır. Bu değişkenlerin ilk değer ataması yapılmıştır. “ciro[]” değişkeni, 4 kategorinin ciro bilgilerini bir dizi olarak tutmaktadır. “kalan_koltuk[]” değişkeni 4 kategorinin boş koltuk sayısını bir dizi olarak tutmaktadır. “matris[][]” değişkeni ise salon koltuklarını ifade etmektedir. 2 katlı bir dizidir ve varsayılan olarak 0 değeri ataması

yapılmıştır. 0 = boş , 1 = dolu koltuk olarak ifade edilmektedir.

```
def salon_olustur(self):
    self.ciro = [0, 0, 0, 0]
    self.kalan_koltuk = [100, 100, 100, 100]
    self.matris = [0] * 20
    for i in range(20):
        self.matris[i] = [0] * 20
```

Program başlangıcında ikinci olarak imbd_oku() fonksiyonu çağırılmaktadır. Bu fonksiyonun anlatımı opsiyonlar bölümünde açıklanmıştır.

Program başlangıcında üçüncü olarak dosya_oku() fonksiyonu çağırılmaktadır. Bu fonksiyon “indirim.txt” dosyasının içeriğini okuyup içerisindeki değerleri gerekli değişkenlere atamaktadır. Dosyanın içeriği “dosya” değişkeninde tutulmaktadır. Değişkene atama yapılırken try-except methodu kullanılmıştır. Dosya mevcut değil ise ekrana çıktı verilip program doğrudan exit(0) komutu ile kapatılmaktadır. Dosya içeriğini satırlarda tutmak için “liste” isimli bir liste değişkeni tanımlanmıştır ve dosyanın satırları split edilerek bu listenin içine ataması yapılmıştır. Daha sonrasında “maxBilet”, “ucretler[]” ve “indirim[][]” değişkenleri tanımlanmıştır. “maxBilet” değişkeni, “indirim.txt” dosyasının ilk satırında yer alan maksimum bilet değerini tutması için tanımlanmıştır. “ucretler” değişkeni ise 2 ve 4. satırlar arasında yer alan kategori bilet ücretlerinin değerlerini tutması için tanımlanmıştır. “indirim” değişkeni ise 6 ve 17. satırlar arasındaki indirim değerlerini tutması için tanımlanmıştır.

```
def dosya_oku(self):
    try:
        self.dosya = open("indirim.txt", "r")
    except:
        print("İndirim Bilgileri Okunamadı Program Kapatılıyor !")
        exit(0)
    self.liste = []
    self.liste += self.dosya.read().split()
    for i in self.liste:
        i = i.strip()
    self.ucretler = []
    self.indirim = [0] * 12
    for i in range(12):
        self.indirim[i] = [0] * 3
    for i in range(0, len(self.liste), 1):
        if i == 0:
            carpan = 1
```

Bu değişkenlerin ataması için bir döngü kullanılmıştır. Bu döngü satırların uygun bir biçimde okunup gerekli değişkenlere atanması için özel bir algoritma içermektedir. Bu algoritma oluşturulurken, kullanıcının “indirim.txt” dosyasındaki değerleri daha sonrasında değiştirebileceği varsayılarak tasarlanmıştır. Örnek olarak 1.kategorinin ücreti 200 TL yerine 5 TL veya 3000 TL yapabilmek mümkündür veya Maksimum bilet değerini 30 yerine 5 veya 70 yapabilmek mümkündür. Aynı şekilde indirim aralığındaki mevcut sayılarda değiştirilebilir. Kullanıcı “M” ifadesi yerine sayısal bir değer girebilmektedir.

```

for i in range(0, len(liste), 1):
    if i == 0:
        carpan = 1
        maxBilet = 0
        for h in range(len(liste[i]) - 1, 1, -1):
            maxBilet += int(liste[i][h]) * carpan
            carpan *= 10
    elif 1 <= i <= 4:
        carpan = 1
        sayi = 0
        for h in range(len(liste[i]) - 1, 1, -1):
            sayi += carpan * int(liste[i][h])
            carpan *= 10
        ucretler.append(sayi)
    elif 5 <= i <= 16:
        indirim[i - 5][0] = int(liste[i][2]) * 10 + int(liste[i][3])
        if liste[i][5] == 'M':
            indirim[i - 5][1] = 100
        else:
            indirim[i - 5][1] = int(liste[i][5]) * 10 + int(liste[i][6])
        try:
            indirim[i - 5][2] = int(liste[i][8]) * 10 + int(liste[i][9])
        except:
            indirim[i - 5][2] = int(liste[i][7]) * 10 + int(liste[i][8])
return

```

Programın açılışında dördüncü olarak muzik_cal() fonksiyonu kullanılmaktadır. Bu fonksiyonun içeriği opsiyonlar bölümünde açıklanmıştır.

Programın açılışında beşinci olarak kapak_yazdir() fonksiyonu kullanılmaktadır. Bu fonksiyonun içeriği opsiyonlar bölümünde açıklanmıştır.

Programın açılışındaki gerekli fonksiyonlar çalıştırıldıktan sonra while döngüsü çalıştırılmaktadır. Bu while döngüsü kullanıcı 0 değerini girene kadar devam etmektedir. 1,2,3,4,5,6 değerlerinde ise çeşitli özellikler yer almaktadır.

```

*****
** ANA MENÜ **
*****
1. Rezervasyon Oluştur
2. Salonu Yazdır
3. Yeni Etkinlik Oluştur
4. Ciro Hesapla
5. Bilgileri PDF'e Aktar
6. Rezervasyon İptal Et
0. Çıkış
*****

```

Kullanıcıdan seçim değeri alınırken try-except metodunun içinde alınmıştır ve kullanıcının string veya boş bir değer girip programı çökertmesi engellenmiştir. Eğer kullanıcı 1 seçimi yapar ise ayrı bir while döngüsü başlamaktadır ve kullanıcıdan hangi kategoriden rezervasyon yaptırmak istediği sorulmaktadır. Kullanıcının string ve boş değer girme durumu kontrol edilmektedir.

```

while self.kategori != 0:
    try:
        self.kategori = int(input("Kategori Seçiminiz (1-4): "))
    except:
        pass

```

Kullanıcı hangi kategoriden rezervasyon yapmak istediğini seçtikten sonra adet_al() fonksiyonu çağırılmaktadır. Bu fonksiyon içerisinde kullanıcının maksimum kaç adet bilet seçebileceği “maxBilet” değişkeni ile belirlenmektedir ve adet girişi yapılması istenmektedir. Eğer kullanıcı 1’den daha küçük veya “maxBilet” değerinden daha büyük bir değer, string veya boş bir değer girerse ekrana hata mesajı yazdırılıp programın ana ekranına geri dönülmektedir.

```
def adet_al(self):
    try:
        print("Adet Giriniz(0-", self.maxBilet, ")\n", end="", sep="")
        tmp = int(input())
        if tmp < 0:
            print("Adet Sayısı 1'den Küçük Olamaz !")
            sleep(0.5)
            tmp = 0
        if tmp > self.maxBilet:
            print("Adet Sayısı ", self.maxBilet, " Sayısından daha fazla olamaz !")
            tmp = 0
            return
        except:
            print("Adet Sayısını Yanlış Girdiniz")
            sleep(0.5)
            tmp = 0
        return tmp
```

Kullanıcı doğru bir adet girişi yaparsa yetersiz_koltuk() fonksiyonu çağırılmaktadır. Bu fonksiyon içerisinde, kullanıcının rezervasyon yaptırmak istediği kategori içerisinde yeterli koltuk sayısı mevcut mudur diye “kalan_koltuk[]” değişkeni içerisinde kontrol edilmektedir. Eğer yeterli yer mevcut değilse ekrana hata mesajı yazdırılmaktadır ve ana menüye geri dönülmektedir.

```
def yetersiz_koltuk(self, adett, n):
    if adett > self.kalan_koltuk[n]:
        print("Yeterli Koltuk Yok. Rezervasyon İşlemi Red Edildi!")
        sleep(0.5)
        return True
```

Eğer yeterli yer mevcut ise “biletIslem” kütüphanesinin içindeki bilet_olustur() fonksiyonu çağırılmaktadır. 1 ve 3. Kategoriler ile 2 ve 4. Kategorilerin bilet oluşturma fonksiyonları birbirinden farklıdır. Bunun nedeni koltukların farklı bir algoritma ile seçilmesinden dolayıdır. Çağırılan bilet_olustur() fonksiyonuna bazı özel değerler gitmektedir. Bu değerler kullanıcının seçtiği kategoriye göre değişmektedir. Bu parametreler kullanıcının seçtiği kategori hangi satır ve sütun numaraları ile başlayıp hangi numara ile bitmektedir, adet bilgisi, kategori değişkeni, “matris[][]” değişkeni, “kalan_koltuk[]” değişkeni olarak belirlenmiştir.

```
biletIslem_100220027.bilet_olustur(0, 10, 5, 15, self.adet, 0, self.matris, self.kalan_koltuk)
```

bilet_olustur() fonksiyonu çalıştırılırken kullanıcının 1 ve 3. Kategorilerden seçim yaptığı düşünülüp ona göre bir algoritma oluşturulmuştur çünkü iki kategoride de koltuk-bilet yerleşimi aynı hızda yapılmaktadır fakat satır, sütun başlangıç değerleri farklıdır. Fonksiyon çalışırken iki döngü kullanılmıştır ilk döngü satırlar için ikinci döngü sütunlar içindir. “n” değişkeni burada kategori değişkenini temsil etmektedir.

Çalıştırılan döngü tüm kategorinin en başından en sonuna kadar devam edecek şekilde ayarlanmıştır fakat adet değişkeni 0 olduğu anda fonksiyon duracaktır. Bu nedenle “maxBilet” değişkeni 100 olarak ayarlanıp 100 bilet satın alınırsa herhangi bir problem yaşanması mümkün değildir. Döngü çalışırken eğer koltuk dolu olarak gözüküyorsa gerekli işlemler yapılmaktadır. Bu işlemler; matris değerinin 1 (koltuğun dolu) olarak ayarlanması, “adet” sayısının 1 azaltılması, kalan_koltuk[] değerinin 1 azaltılması, 0.2 saniyelik bekleme süresi konularak kullanıcının hangi koltukların rezerve edildiğini görmesi ve ekrana yazdırılması daha sonrasında adet değişkeni 0 olursa fonksiyonun durdurulmasıdır.

```
def bilet_olustur(t1, t2, y1, y2, adett, n, matris, kalan_koltuk):
    print("Rezerve Edilen Koltuklar (Sıra-Koltuk): ", end="")
    for i in range(t1, t2, 1):
        print()
        for j in range(y1, y2, 1):
            if matris[i][j] == 1:
                continue
            matris[i][j] = 1
            adett -= 1
            kalan_koltuk[n] -= 1
            sleep(0.2)
            print(i + 1, " - ", j + 1, ", ", end="", sep="")
            if adett == 0:
                print()
                return 0
```

Aynı mantık 2 ve 4. kategoriler için oluşturulan bilet_olustur2() fonksiyonu için de geçerlidir fakat döngü yapısı 2. ve 4.kategorilerin bilet-koltuk yerleşme sırasına göre oluşturulmuştur. “adet” değişkeni 0 olduğu anda fonksiyon tamamlanmaktadır.

```
def bilet_olustur2(t1, t2, y1, y2, y3, y4, adett, n, matris, kalan_koltuk):
    print("Rezerve Edilen Koltuklar (Sıra-Koltuk): ", end="")
    for i in range(t1, t2, 1):
        print()
        for j in range(y1, y2, -1):
            if matris[i][j] == 1:
                continue
            matris[i][j] = 1
            adett -= 1
            kalan_koltuk[n] -= 1
            sleep(0.2)
            print(i + 1, " - ", j + 1, ", ", end="", sep="")
            if adett == 0:
                print()
                return
        for h in range(y3, y4, 1):
            if matris[i][h] == 1:
                continue
            matris[i][h] = 1
            adett -= 1
            kalan_koltuk[n] -= 1
            sleep(0.2)
            print(i + 1, " - ", h + 1, ", ", end="", sep="")
            if adett == 0:
                print()
                return
```

Bilet rezervasyonu başarılı bir şekilde tamamlandıktan sonra fiyat_hesapla() fonksiyonu çalışmaktadır. Bu fonksiyon içerisine kategoriye özel parametreler gitmektedir. Bu parametreler; kategori değişkeni, o kategorinin indirim bilgilerinin hangi indis aralıklarında bulunduğu, adet değeri parametreleridir.

```
self.fiyat_hesapla(0, 0, 3, self.adet)
```

Fonksiyon çalıştıktan sonra bir döngü ile “indirim[][]” değişkeni içerisinde kategorinin indirim değerleri bulunmaktadır ve adet değişkenine uygun olan indirim değeri belirlenmektedir. Daha sonrasında o kategorinin bilet ücreti * bilet adedi – indirim miktarı formülü ile toplam fiyat belirlenmektedir ve ekrana tüm bilgiler yazdırılmaktadır. Eğer kullanıcının satın aldığı bilet sayısı herhangi bir aralığa dahil değil ise indirim kampanyasından yararlanamadığına dair bir mesaj yazdırılmaktadır ve normal ücret tarifesinden ücret hesaplanarak ekrana yazdırılmaktadır ve fonksiyon tamamlandıktan sonra programın ana ekranına dönlülmektedir.

```
def fiyat_hesapla(self, n, l1, l2, adett):
    sleep(0.7)
    for i in range(l1, l2, 1):
        if self.indirim[i][0] <= adett <= self.indirim[i][1]:
            fiyatt = self.ucretler[n] * adett - (self.ucretler[n] * adett) / 100 * self.indirim[i][2]
            print("\nBilet Adeti:", adett,
                  "\nBilet Tutarı: ", self.ucretler[n],
                  "\nToplam Tutar:", self.ucretler[n] * adett,
                  "\nYapılan İndirim:", (self.ucretler[n] * adett) / 100 * self.indirim[i][2],
                  "\nNet Tutar:", fiyatt)
            sleep(1.4)
            self.ciro[n] += fiyatt
            return
    print("İndirim Kampanyasından Faydalanamıyorsunuz.", end="1")
    fiyatt = self.ucretler[n] * adett
    print("\nBilet Adeti:", adett,
          "\nBilet Tutarı: ", self.ucretler[n],
          "\nNet Tutar:", fiyatt)
    sleep(1.4)
    self.ciro[n] += fiyatt
    print()
    return
```

Rezervasyon işlemleri tamamlandıktan sonra kullanıcı 2.seçenek olarak salonu yazdır seçeneğini seçer ise salon_yazdir() fonksiyonu çalışmaktadır. Bu fonksiyon çalıştırıldığında ekrana; tarih, saat, vizyondaki filmin adı yazdırılmaktadır ve 1 saniyelik sleep() komutu kullanılmaktadır. Tanımlanan time, tarih ve saat değişkenleri, gün ve saatin yazdırılması için kullanılmaktadır. Daha sonrasında satır ve indis numaraları ile birlikte ekrana salon bilgileri yazdırılmaktadır. Bilgiler yazdırılırken ve yazdırma tamamlandığında tekrar sleep() fonksiyonu kullanılarak kullanıcının ekrandaki çıktıları daha rahat bir şekilde takip edebilmesi amaçlanmıştır.

```

def salon_yazdir(self):
    time = datetime.datetime.now()
    tarih = str(date.today())
    saat = str(time.hour) + ":" + str(time.minute) + ":" + str(time.second) + "\n"
    print("\nVizyondaki Film:", self.film, end="")
    print("Tarih - Saat:", tarih, saat, end="")
    print(" ", end="")
    sleep(1)
    for i in range(20):
        if i < 9:
            print("0", end="")
            print(i + 1, "", end="")
        for i in range(20):
            print()
            sleep(0.2)
            for j in range(20):
                if j <= 0:
                    if i < 9:
                        print("0", end="")
                        print(i + 1, end="")
                    if self.matris[i][j] == 0:
                        print(" - ", end="")
                    else:
                        print(" X ", end="")
            print()
            sleep(1)

```

```

Vizyondaki Film: Lock, Stock and Two Smoking Barrels
Tarih - Saat: 2022-12-28 20:34:17
 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20
01 X  X  X  X  X  X  X  X  X  X  X  -  -  -  -  -  X  X  X  -  -
02 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
03 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
04 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
05 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
06 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
07 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
08 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
09 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
10 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
11 X  X  X  X  X  -  -  -  -  -  -  -  -  -  -  -  X  X  X  -  -
12 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
13 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
14 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
15 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
16 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
17 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
18 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
19 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -
20 -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -

```

Salonun yazdırılması tamamlandığında kullanıcı ciroları görüntülemek için ciroları görüntüle seçimini seçerse ciro_hesapla() fonksiyonu çalışacaktır. Bu fonksiyon "ciro[]" değişkeni içerisindeki tüm değerleri toplam değişkeni içerisinde toplayarak toplam ciroyu bulmaktadır. Kategorilerin ve tüm salonun cirosunu ekrana yazdırmaktadır. sleep() metodu 1 saniyeliğine çalışmaktadır ve fonksiyonu sonlandırmaktadır.

```
def ciro_hesapla(self):  
    toplam = 0  
    print()  
    for i in range(0, len(self.ciro), 1):  
        print("Kategori", i + 1, "Cirosu:", self.ciro[i])  
        toplam += self.ciro[i]  
        sleep(1)  
    print("Toplam Ciro", toplam)  
    sleep(1)
```

Ciro gösterme işlemi tamamlandığında kullanıcı yeni bir etkinlik oluşturmak ister ve yeni etkinlik oluştur seçeneğini seçerse `salon_olustur()`, `imdb_oku()`, `dosya_oku()`, `salon_yazdir()` fonksiyonları çalışacaktır ve programda kullanılan tüm değişkenler sıfırlanacaktır, yeni bir film adı seçilecektir fakat **ciro değerleri sabit kalacaktır.** En sonunda 0 seçeneği seçilirse program kapanacaktır.

```
elif self.secim == 3:  
    self.salon_olustur()  
    self.imdb_oku()  
    self.dosya_oku()  
    self.salon_yazdir()  
    continue
```


Opsiyonlar

- Web browser kütüphanesi kullanılarak, program başladığında arka fonda Mozart – Türk Marşı’nın çalması.

```
def muzik_cal():  
    webbrowser.open('piyano.mp3')
```

- IMDB’nin Top 250 film listesi alınarak imdb.txt dosyasına kaydedilmiştir. Fonksiyon başlatıldığında bu dosya okunmaktadır ve “imdb” değişkeninin içine kaydedilmektedir. random kütüphanesi ile içerisinde rastgele 1 film seçilmektedir ve “film” değişkeni içerisinde saklanmaktadır. Bu film yeni etkinlik oluşturuluncaya kadar hafızada tutulmaktadır ve salonu yazdırma işlemi ile PDF oluşturma işleminde en üst satırda gözükmektedir.

```
def imdb_oku(self):  
    try:  
        self.imdb = open("imdb_100220027.txt", "r")  
    except:  
        print("Film Bilgileri Okunamadı Program Kapatılıyor !")  
        exit(0)  
    self.rnd = random.randint(0, 251)  
    self.liste = self.imdb.readlines()  
    self.film = self.liste[self.rnd]
```

Vizyondaki Film: Star Wars

- Salonu yazdırma işleminde satır ve sütun numaraları ile birlikte yazdırılmaktadır. Bu işlemin algoritması, bilet_yazdır fonksiyonu içerisinde tasarlanmıştır.

	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20
01	-	-	-	-	-	X	X	X	X	X	X	X	X	X	X	-	-	-	-	-
02	-	-	-	-	-	X	X	X	X	X	X	X	X	X	X	-	-	-	-	-
03	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
04	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
05	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
06	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
07	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
08	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
09	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
14	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
15	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
18	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

- datetime kütüphanesi kullanılarak mevcut tarih ve saat bilgisi alınıp, salonu yazdırma işleminde görüntülenmektedir.

Tarih - Saat: 2022-12-28 17:48:42

- sleep kütüphanesi kullanılarak ekrana yazdırılan bilgilerin arasında anlamlı bekleme süreleri konularak, kullanıcının ekrandaki yazıları daha kolay bir şekilde takip etmesi amaçlanmıştır.

```
print("\nVizyondaki Film:", self.film, end="")  
print("Tarih - Saat:", tarih, saat, end="")  
print(" ", end="")  
sleep(1)
```

- [illegible]

- ```
def pdf_aktar(self):
 pdf = FPDF()
 pdf.add_page()
 pdf.set_font("Courier", size=12)
 time = datetime.datetime.now()
 tarih = str(date.today())
 saat = str(time.hour) + ":" + str(time.minute) + ":" + str(time.second)
 tmp = "Tarih - Zaman: " + tarih + " " + saat
 pdf.write(5, tmp)
 tmp = "\nVizyondaki Film:" + str(self.film)
 pdf.write(6, tmp)
 pdf.cell(w=150, h=10, txt="SALON BILGILERI", border=1, ln=1, align='C')
 pdf.write(5, "\n ")
 for i in range(20):
 if i < 9:
 pdf.write(5, "0")
 tmp = str(i + 1) + " "
 pdf.write(5, tmp)
 for i in range(20):
 pdf.write(5, "\n")
```

Tarih - Zaman: 2022-12-28 17:58:59

Vizyondaki Film: Princess Mononoke

SALON BİLGİLERİ

|    | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 01 | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  |
| 02 | -  | -  | -  | -  | -  | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | -  | -  | -  | -  | -  |
| 03 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 04 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 05 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 06 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 07 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 08 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 09 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 10 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 11 | x  | x  | x  | x  | x  | x  | x  | x  | x  | x  | -  | -  | -  | -  | -  | x  | x  | x  | x  | x  |
| 12 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 13 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 14 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 15 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 16 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 17 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 18 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 19 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |
| 20 | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  | -  |

CIRO BİLGİLERİ

Kategori 1 Ciro:3200.0  
Kategori 2 Ciro:1425.0  
Kategori 3 Ciro:425.0  
Kategori 4 Ciro:570.0

Toplam Ciro: 5620.0

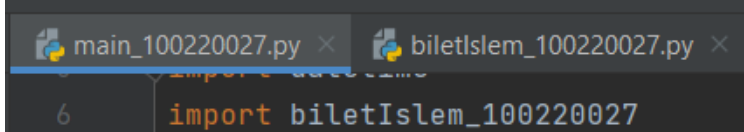
- Bilet iptal etme özelliği eklenmiştir. Bu özellik ile yapılan rezervasyonları iptal edebilmek ve ücret iadesi yapabilmek mümkündür. Ücret iadesi yapılırken bilet tutarının %50'si kadar iade yapılmaktadır ve yapılan iade tutarı ciro bilgilerinden eksiltilmektedir. İptal edilen koltuk numaraları ve iade edilen toplam tutar bilgileri ekrana yazdırılmaktadır. sleep kütüphanesi burada da kullanılarak gerçekçi bir görünüm amaçlanmıştır. Bilet oluşturma algoritması gibi bilet iptalinde de 2 farklı fonksiyon kullanılmıştır. Bilet oluşturma fonksiyonundaki işlemlerin tersi gerçekleştirilmektedir.

```
def bilet iptal(t1, t2, y1, y2, adett, n, matris, kalan_koltuk):
 print("İptal Edilen Koltuklar (Sıra-Koltuk): ", end="")
 for i in range(t1, t2, 1):
 print()
 for j in range(y1, y2, 1):
 if matris[i][j] == 0:
 continue
 matris[i][j] = 0
 adett -= 1
 kalan_koltuk[n] += 1
 sleep(0.2)
 print(i + 1, " - ", j + 1, ", ", sep="")
 if adett == 0:
 print()
 return 0
```

```
İptal Edilen Koltuklar (Sıra-Koltuk):
1 - 6, 1 - 7, 1 - 8, 1 - 9, 1 - 10, 1 - 11, 1 - 12, 1 - 13,

Bilet İade Tutarı, Normal Bilet Ücretinin Yarısı Kadardır !
Bilet Adeti: 8
Toplam Tutar: 1600
İade Tutarı: 800.0
```

- Modüler programlama anlayışı ile bilet işlem fonksiyonları biletIslem.py dosyası adı altında yer almaktadır ve main dosyası içerisinde import edilip çağırılmaktadır.



```
main_100220027.py x biletIslem_100220027.py x
6 import biletIslem_100220027
```

- Sınıf yapısı kullanılmıştır. İstenildiği takdirde yeni nesneler tanımlanarak birden fazla salon ile çalışacak şekilde ayarlayabilmek mümkündür.

```
s1 = Salon()
s1.menu()
```

```
class Salon:
 def __init__(self):
 self.salon_olustur()
 self.imdb_oku()
 self.dosya_oku()
```

- Belirtilen özelliklere ek olarak bazı küçük opsiyonlar daha eklenmiştir...