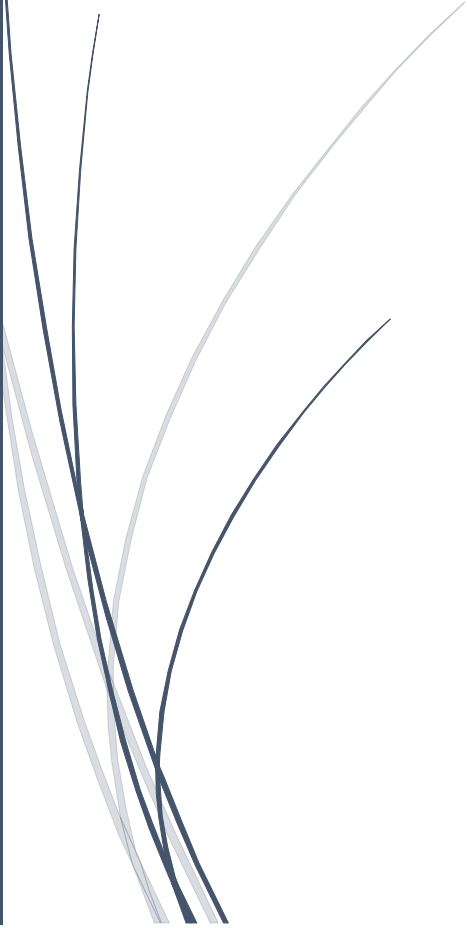




08.06.2023

# BIL104 Final Projesi

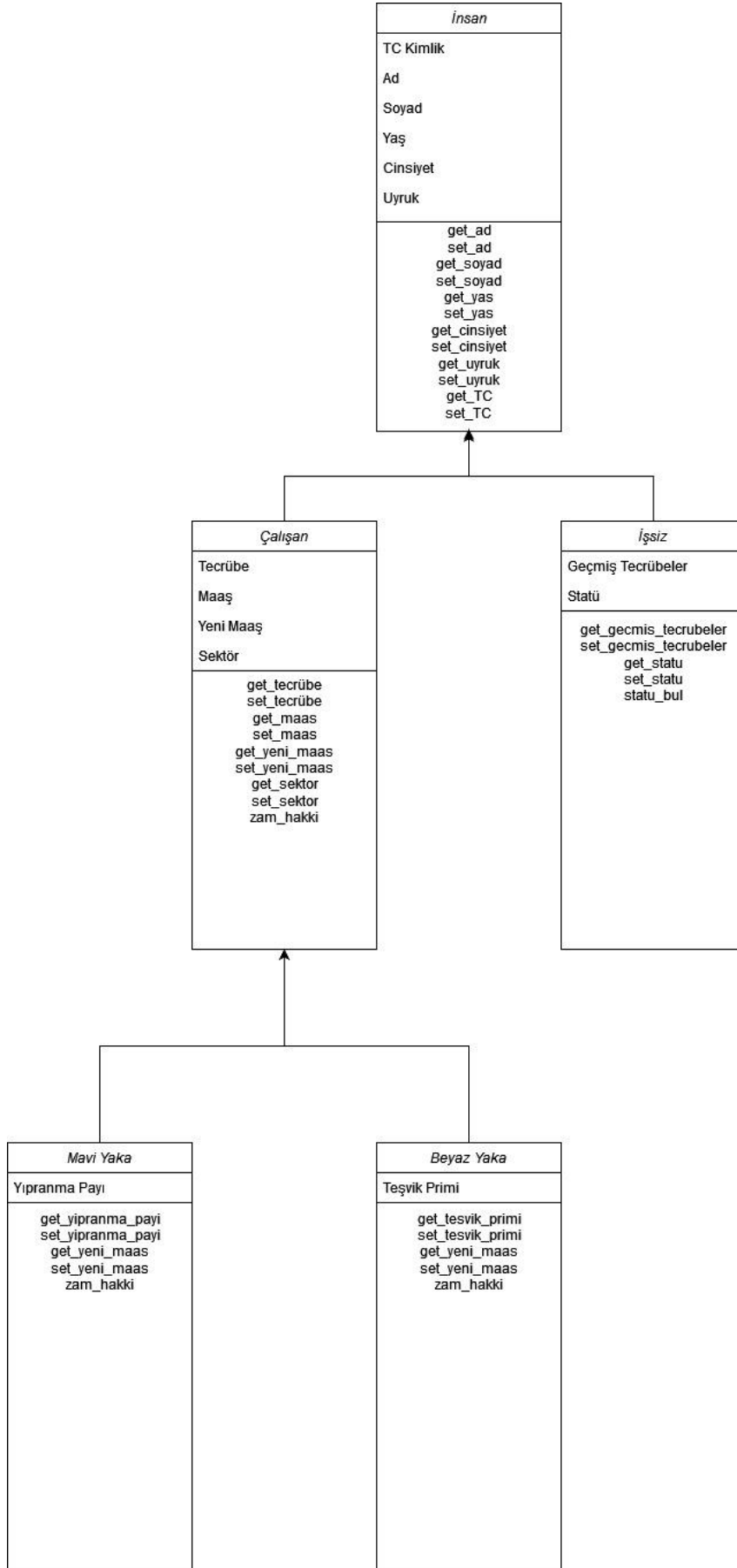


Aksel SAATCI  
220601122

## İçindekiler

- UML Diyagramı
- Sözde Kod
- İnsan Classı Kod Açıklaması ve Çıktıları
- Issiz Classı Kod Açıklaması ve Çıktıları
- Calisan Classı Kod Açıklaması ve Çıktıları
- MaviYaka Classı Kod Açıklaması ve Çıktıları
- BeyazYaka Classı Kod Açıklaması ve Çıktıları
- GitHub Geçmişi ve Notları

## UML Diyagramı



## Sözde Kod

1. Başla
2. Boş bir obje\_listesi oluştur
3. İnsan1, İnsan2, İssiz1, İssiz2, İssiz3, Calisan1, Calisan2, Calisan3, MaviYaka1, MaviYaka2, MaviYaka3, BeyazYaka1, BeyazYaka2, BeyazYaka3 objelerini oluştur
4. Oluşturulan objeleri ekrana yazdır
5. Oluşturulan objeleri obje\_listesi ne ekle
6. Dictionary tipinde bir data değişkeni oluştur
7. Data değişkeninin nesne key'ine obje\_listesindeki her bir objeyi dönerek o objenin tipini value olarak ata
8. Data değişkeninin TC No key'ine obje\_listesindeki her bir objeyi dönerek o objenin TC No'sunu value olarak ata
9. Data değişkeninin Ad key'ine obje\_listesindeki her bir objeyi dönerek o objenin adını value olarak ata
10. Data değişkeninin Soyad key'ine obje\_listesindeki her bir objeyi dönerek o objenin soyadını value olarak ata
11. Data değişkeninin Yaş key'ine obje\_listesindeki her bir objeyi dönerek o objenin yaşını value olarak ata
12. Data değişkeninin Cinsiyet key'ine obje\_listesindeki her bir objeyi dönerek o objenin cinsiyetini value olarak ata
13. Data değişkeninin Uyruk key'ine obje\_listesindeki her bir objeyi dönerek o objenin uyruğunu value olarak ata
14. Data değişkeninin Sektör key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_sektor fonksiyonu varsa bu fonksiyonun değerini value olarak ata, yok ise valueyi 0 olarak ata
15. Data değişkeninin Tecrübe key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_tecrube fonksiyonu varsa bu fonksiyonun değerini 12 ye bölüp value olarak ata, yok ise valueyi 0 olarak ata
16. Data değişkeninin Maaş key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_maas fonksiyonu varsa bu fonksiyonun değerini value olarak ata, yok ise valueyi 0 olarak ata
17. Data değişkeninin Yıpranma Payı key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_yipranma\_payi fonksiyonu varsa bu fonksiyonun değerini value olarak ata, yok ise valueyi 0 olarak ata
18. Data değişkeninin Teşvik Primi key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_tesvik\_primi fonksiyonu varsa bu fonksiyonun değerini value olarak ata, yok ise valueyi 0 olarak ata
19. Data değişkeninin Yeni Maaş key'ine obje\_listesindeki her bir objeyi dönerek eğer o objenin get\_yeni\_maas fonksiyonu varsa bu fonksiyonun değerini value olarak ata, yok ise valueyi 0 olarak ata
20. Yeni bir data frame oluştur Kolonları "Nesne Değeri", "TC No", "Ad", "Soyad", "Yaş", "Cinsiyet", "Uyruk", "Sektör", "Tecrübe(Yıl)", "Maaş", "Yıpranma Payı", "Teşvik Primi", "Yeni Maaş" ve datası Data değişkeni olsun ve ekrana yazdır

21. mavi\_yakalilar , beyaz\_yakalilar , calisanlari oluştan dataframeden ayrı bir dataframe oluştur oluşturulan bu datframelerden maaş kolonunun ortalamasını mean fonksiyonu ile hesapla ve ekrana yazdır
22. Maaşı 15000 den fazla olanları ana dataframeden sorgulayarak bul ve bunları yeni bir dataframe olarak yazdır
23. Ana dataframeyi .sort\_values fonksiyonu ile by parametresi Yeni Maaş , ascending parametresi False olacak şekilde oluştur ve ekrana yazdır.
24. Tecrübesi 3 seneden fazla olanları ve beyaz yakalı olanları ana dataframeden sorgula ve yeni bir dataframe oluştur. Bu dataframeyi ekrana yazdır.
25. Yeni maaşı 10000'in üzerinde olanları ve dataframede 2 ile 4. Satır arasında olanları yeni bir dataframe olarak oluştur ve ekrana yazdır.
26. Ana dataframe'den sadece Ad,Soyad,Sektör ve Yeni Maaş kolonlarını alarak yeni bir dataframe oluştur ve oluşan dataframeyi ekrana yazdır.

## Insan Classı

Bu classımda insan objesine ait olabilecek özellikleri tanımladım. Tüm alanları private olarak tanımladıktan sonra bu alanlara get ve set fonksiyonlarını yazdım. Daha sonra bir \_\_init\_\_ fonksiyonu tanımladım ve bu fonksiyonumda obje oluşturulurken gerekli alanları doldurdum. Son olarak \_\_str\_\_ fonksiyonu oluşturdum. Burada string.format kullanarak gereken tüm bilgileri bir string halinde return ediyorum.

```
class Insan:
    def __init__(self, tc_no, ad, soyad, yas, cinsiyet, uyruk_bilgileri):
        self._tc_no = tc_no
        self._ad = ad
        self._soyad = soyad
        self._yas = yas
        self._cinsiyet = cinsiyet
        self._uyruk_bilgileri = uyruk_bilgileri
```

```
C:\Users\aksel> python3
Tc No: 12345678910
Ad: Aksel
Soyad: Saatcı
Yaş: 20
Cinsiyet: Erkek
Uyruk Bilgileri: Türk
PS C:\Users\aksel\Desktop\final_odev> []
```

Yanda görüldüğü gibi classımı stringe çevirdiğimde \_\_str\_\_ fonksiyonu çalışıyor ve bana düzenli halde objemin özelliklerini bir string olarak dönüyor.

## Issiz Classı

Bu classımda insan objesini import ediyorum ve oluşturdum Issiz clasına miras olarak Insan classını alıyorum. Statu değerini public olarak tanımlıyorum ve başlangıç değerini Bulunamadi olarak atıyorum. Daha sonra bu classıma \_\_init\_\_ fonksiyonunu tanımlıyorum. Burada yine parametre olarak insan objesinin parametrelerinin yanında bir geçmiş\_tecrubeler kitaplığı alıyorum. Daha sonra private alanlarımı doldurduktan sonra daha sonra tanımladığım statu\_bul fonksiyonunu da burada çağırıyorum. Daha sonra ana Insan

classımın `__init__` fonksiyonunu kullanmak için `super` keywordü ile `__init__` fonksiyonunu çağırıyorum ve gerekli parametreleri içine atıyorum. Daha sonra `private` alanlarımı `get` ve `set` ettiğim fonksiyonlar mevcut. Bu fonksiyonlar arasında `get_statu` fonksiyonumda eğer `statu` alanım `Bulunamadi` ise tekrardan `statu_bul` fonksiyonumu çağırıyorum. Daha sonra oranlarını `mavi_yaka %20`si , `beyaz_yaka %35`'i ve yönetici `%45`'i olacak şekilde hesapladıktan sonra en büyük olanı statü olarak atıyorum. Daha sonra `__str__` fonksiyonum var bu fonksiyonda `string.format`'ı kullanarak gerekli parametreleri string olarak döndürüyorum.

```
aksel = Issiz("12345678910", "Aksel", "Saatci", 20, "Erkek", "Turk", {
    "mavi_yaka": 10, "beyaz_yaka": 20, "yonetici": 30})
print(str(aksel))
```

```
Aksel Saatci
Statu: Yonetici
PS C:\Users\aksel\Desktop\final_odev>
```

Yukarıda oluşturduğum aksel nesnesini string olarak yazdırdığımda aldığım sonucu görebilirsiniz.

## Calisan Classı

Bu classımda öncelikle insan classını import ediyorum daha sonra oluşturduğum `Calisan` classında `Insan` classını miras olarak alıyorum. Öncelikle sektörlerimi classımdan önce bir liste şeklinde tanımladım. Daha sonra sektörleri buradan kontrol edeceğim. Classımda öncelikle `__init__` fonksiyonumu tanımladım. Bu fonksiyonum insan parametrelerinin yanında sektör yeni maaş ve maaş da alıyor. Bunları `private` olarak set ediyorum. Maaşı ve tecrübeyi set ettikten sonra `zam_hakki` fonksiyonumu çağırıyorum. Sektörü set ederken bir `if` bloğu ile alınan sektörü sektörler listesine mevcut mu diye check ediyorum. Eğer mevcut değil ise bir hata ayağa kaldırıyorum. Daha sonra insan classımda olan özellikleri de doldurmak için `süper keyword`ünü kullanıp insan classının `__init__` fonksiyonunu çağırıyorum. Daha sonra oluşturduğum `private` alanların `get` ve `set` metodlarını oluşturuyorum. `get_yeni_maas` fonksiyonumda eğer yeni\_maas 0 ise `zam_hakki` fonksiyonunu çağırıyorum ve yeni maası geri dönüyorum. Daha sonra `zam_hakki` fonksiyonum var. Bu fonksiyonda eğer tecrübe yılı 2 den küçük ise yeni maaşı şuanki maaşına eşitliyorum ve zam oranı olarak 0 dönüyorum. 2 ila 4 yıl arasında ve maaşı 15000 tlden az ise zam oranı olarak maaşını tecrübe yılına mod olarak alıp zam oranını buluyorum. Daha sonra yeni maaşı buluyorum ve zam oranını fonksiyonumdan geriye dönüyorum. Eğer tecrübe yılı 4 den büyük ve maaşı 25000 den küçük ise zam oranını maaşın tecrübe yılına göre modunu 2 ye bölerek buluyorum. Daha sonra yeni maaşı buluyorum ve zam oranını geri dönüyorum. Eğer bu koşullarımdan hiçbiri sağlanmaz ise yeni maaşı eski maaşa eşitliyor ve zam oranını 0 olarak geriye dönüyorum. Daha sonra `__str__` fonksiyonumda gerekli parametrelerimi string olarak dönüyorum.

```
aksel = Calisan(12345678910, "Aksel", "Saatci", 20,
    "Erkek", "Türk", "teknoloji", 24, 10000)
print(str(aksel))
```

```
Aksel Saatci 10000.0 24
```

## Mavi Yaka Classı

Bu classımda Calisan classını import ediyorum. Oluşturduğum MaviYaka classına miras olarak Calisan classını alıyorum. Calisan classına ek olarak yıpranma payı alanımı private olarak tanımlıyorum. \_\_init\_\_ fonksiyonumda Calisan classının init fonksiyonuna gerekli parametreleri yolluyorum ve yıpranma\_payi'ni set ediyorum. Daha sonra yıpranma payına gerekli get ve set fonksiyonlarımı yazıyorum. Daha sonra override ettiğim zam\_hakki fonksiyonu var. Bu fonksiyonda öncelikle (tecrübeyi ay olarak aldığım için) tecrübeyi yıla çeviriyorum. Daha sonra eğer tecrübe yılı 2' den küçük ise zam oranını yıpranma payını 10 ile çarparak hesaplıyorum ve bu zam oranına göre yeni maaşı bulduktan sonra geriye zam oranını döndürüyorum. Eğer tecrübe 2 ila 4 yıl arasında ve maaşı 15000'den küçük ise zam oranını maaşın tecrübe yılına göre modunu alarak buluyorum. Daha sonra yeni maaşı hesaplıyorum ve zam oranını geriye dönüyorum. Eğer tecrübe\_yılı 4'den büyük ve maaşı 25000'den küçük ise zam oranını maaşın tecrübe yılına göre modunu 2 ye bölerek hesaplıyorum. Daha sonra yeni maaşı hesaplıyorum ve zam oranını geriye döndürüyorum. Daha sonra \_\_str\_\_ fonksiyonumda gerekli değerleri geriye döndürüyorum.

```
aksel = MaviYaka(12345678910, "Aksel", "Saatçi", 21, "Erkek", "Türk", "muhassebe", 24, 10005, 1)
print(str(aksel))
```

```
Aksel Saatçi 20010.0 24
```

## Beyaz Yaka Classı

Bu classımda öncelikle Calisan classını import ediyorum ve Beyaz Yaka classını oluştururken Calisan classını miras olarak alıyorum. Daha sonra ekstra olarak teşvik primi parametremi private olarak ekliyorum. \_\_init\_\_ fonksiyonumda teşvik primimi atıyorum yeni maaşı 0 olarak atıyorum. Daha sonra gerekli get ve set fonksiyonlarımı tanımlıyorum. get\_yeni\_maas fonksiyonumda eğer yeni maaş 0 a eşit ise zam\_hakki fonksiyonunu çalıştırıyorum. Override ettiğim zam\_hakki fonksiyonumda öncelikle tecrübe\_yili nı hesaplıyorum. Eğer tecrübe yılı 2 den küçük ise zam oranını maaş ile teşvik primini çarparak buluyorum. Daha sonra yeni maaşı hesapladıktan sonra zam miktarını bu fonksiyonumdan geriye dönüyorum. Eğer tecrübe yılı 2 ila 4 yıl arasında ve maaş 15000 TL'nin altında ise zam oranını maaşın tecrübe yılına göre modu ile 5'i çarpıp buna teşvik miktarını ekleyerek buluyorum. Daha sonra zam miktarı ve yeni maaşı hesapladıktan sonra zam miktarını bu fonksiyonumdan geriye dönüyorum. Eğer tecrübe yılı 4 ten büyük ve maaş 25000'den küçük ise zam miktarını maaşın tecrübe yılına göre modunu 4 ile çarptıktan sonra teşvik primini ekleyerek buluyorum. Daha sonra yeni maaşı hesapladıktan sonra zam miktarını fonksiyonumdan geriye dönüyorum. Eğer bu koşullardan hiçbirine uymuyor ise yeni maaşı eski maaşıma eşitliyorum. Daha sonra \_\_str\_\_ fonksiyonum ile gerekli değerlerimi string olarak geriye dönüyorum.

```
BeyazYaka3 = BeyazYaka("3413546653454", "Ümit", "Kalem",
                        26, "Erkek", "Yabancı", "inşaat", 8, 28000, 500)
print(str(BeyazYaka3))
```

```
Ümit Kalem 28500 8
```

## Main Dosyam

Bu dosyamda öncelikle oluşturduğum tüm classlarımı import ediyorum. Pandas kütüphanemide daha sonra kullanacağım için import ediyorum. Öncelikle bir obje listesi oluşturuyorum. Bu obje listesine oluşturduğum ve dataframe için kullanacağım tüm objeleri atıyorum. Bunu yapmamın sebebi daha sonra yeni bir çalışan eklemek istediğimde sadece bu listeye eklememin yeterli olmasını sağlamak. Daha sonra classlarımdan objeler türetiyorum , bunları string yapıp ekrana yazdırıyorum ve obje listeme tüm bu objeleri ekliyorum. Daha sonra bir data kitaplığı oluşturuyorum. Burada keylerimi kolon isimleriyle aynı yapmaya dikkat ettim.

```
data = {
    "Nesne Değeri": [type(obje).__name__ for obje in obje_listesi],
    "TC No": [obje.get_tc_no() for obje in obje_listesi],
    "Ad": [obje.get_ad() for obje in obje_listesi],
    "Soyad": [obje.get_soyad() for obje in obje_listesi],
    "Yaş": [obje.get_yas() for obje in obje_listesi],
    "Cinsiyet": [obje.get_cinsiyet() for obje in obje_listesi],
    "Uyruk": [obje.get_uyruk_bilgileri() for obje in obje_listesi],
    "Sektör": [(obje.get_sektor() if hasattr(obje, "get_sektor") else 0) for obje in obje_listesi],
    "Tecrübe(Yıl)": [(obje.get_tecrube() / 12 if hasattr(obje, "get_tecrube") else 0) for obje in obje_listesi],
    "Maaş": [(obje.get_maas() if hasattr(obje, "get_maas") else 0) for obje in obje_listesi],
    "Yıpranma Payı": [(obje.get_yipranma_payi() if hasattr(obje, "get_yipranma_payi") else 0) for obje in obje_listesi],
    "Tevsik Primi": [(obje.get_tesvik_pirimi() if hasattr(obje, "get_tesvik_pirimi") else 0) for obje in obje_listesi],
    "Yeni Maaş": [(obje.get_yeni_maas() if hasattr(obje, "get_yeni_maas") else 0) for obje in obje_listesi]
}
```

Burada her seferinde gerekli değerleri doldurmak için obje listesini dönüyorum. Burda aslında arka arkaya bir sürü for dönmek yerine tek bir for döngüsü ile datanın gerekli keylerine append ederek çok daha verimli bir şekilde çözebilmişim. Nesne değerini alırken typein içine objeyi aldıktan sonra bu type in name'ine bakıyorum. Bunun sebebi type <class İnsan.Insan gibi bir sonuç döndürüyor. Ben ise sadece classın ismini almak istediğim için bu yolu tercih ettim. Aynı şekilde for döngüleri ile gerekli alanları set ediyorum. Sektör keyine geldiğimde bazı classlarda sektör alanına sahip olmadıkları için bunu hasattr() fonksiyonu ile çek ettikten sonra eğer var ise get fonksiyonunun sonucunu yok ise 0 atıyorum. Bunu geri kalan tüm keyler içinde gerçekleştiriyorum. Datamı oluşturduktan sonra bir dataframe oluşturuyorum. Bunu pd.DataFrame ile yapıyorum.(pd pandas) Parametre olarak kolon isimlerimi data olarakda oluşturduğum kitaplığı yolluyorum.

Daha sonra mavi yakalıları bulmak için yeni bir tane dataframe oluşturuyorum ve sadece ana dataframemimde Nesne Değeri MaviYaka olanları alıyorum. Burada ben direk df[df["Nesne Değeri"] == "MaviYaka"] bu şekilde yapmayı tercih ettim. Fakat okunabilirliği ve anlaşılabilirliği arttırmak açısından SQL gibi .where gibi bir sorgu ile de yapabildim. Daha sonra bunları çalışan ve beyaz yaka ile de yaptıktan sonra maaş ortalamalarını maaş colonunu seçtikten sonra .mean() fonksiyonu ile hesaplıyorum.

```
-----
Mavi Yakalıların maaş ortalaması : 35866.666666666664
Çalışanların maaş ortalaması 14733.333333333334
Beyaz yakalıların maaş ortalaması 34533.333333333336
```



Daha sonra maaşı 15000'den yüksek olanları alan bir sorgu oluşturdum bunu yeni bir dataframeye atıyorum ve bunu ekrana yazdırıyorum.

```
-----\Maaşı 15000 den fazla olanların listesi
-----
```

	Nesne Değeri	TC No	Ad	Soyad	Yaş	Cinsiyet	Uyruk	Sektör	Tecrübe(Yıl)	Maaş	Yıpranma Payı	Teşvik Primi	Yeni Maaş
6	Calisan	23243354545	Yiğit	Meriç	56	Erkek	Türk	muhasabe	4.750000	15400	0.0	0	19250.000000
7	Calisan	23243567565	Serkan	Melih	47	Erkek	Türk	teknoloji	1.083333	23000	0.0	0	23000.000000
8	MaviYaka	23245564545	Melih	Fatma	38	Erkek	Yabancı	diğer	0.008333	16800	0.3	0	16803.000000
9	MaviYaka	34354545454	Şeyma	Aydın	36	Kadın	Türk	teknoloji	0.333333	43000	0.5	0	43005.000000
10	MaviYaka	23243564566	Merve	Kaya	32	Kadın	Yabancı	teknoloji	1.916667	47800	0.7	0	47807.000000
11	BeyazYaka	232456565435	Nihal	Kaya	30	Kadın	Türk	diğer	1.833333	60000	0.0	3000	63000.000000
12	BeyazYaka	343454657565	Ayşe	Erdoğan	27	Kadın	Türk	muhasabe	6.583333	15600	0.0	2500	18116.333333
13	BeyazYaka	3413546653454	Ümit	Kalem	26	Erkek	Yabancı	inşaat	0.666667	28000	0.0	500	28500.000000

Aynı şekilde küçükten büyüğe sıralamak için ana dataframeyi sort\_values fonksiyonuna by parametresi olarak yeni maaşı, ascending parametresi olarak False atıyorum.

```
-----
Sort edilmiş dataframe
-----
```

	Nesne Değeri	TC No	Ad	Soyad	Yaş	Cinsiyet	Uyruk	Sektör	Tecrübe(Yıl)	Maaş	Yıpranma Payı	Teşvik Primi	Yeni Maaş
11	BeyazYaka	232456565435	Nihal	Kaya	30	Kadın	Türk	diğer	1.833333	60000	0.0	3000	63000.000000
10	MaviYaka	23243564566	Merve	Kaya	32	Kadın	Yabancı	teknoloji	1.916667	47800	0.7	0	47807.000000
9	MaviYaka	34354545454	Şeyma	Aydın	36	Kadın	Türk	teknoloji	0.333333	43000	0.5	0	43005.000000
13	BeyazYaka	3413546653454	Ümit	Kalem	26	Erkek	Yabancı	inşaat	0.666667	28000	0.0	500	28500.000000
7	Calisan	23243567565	Serkan	Melih	47	Erkek	Türk	teknoloji	1.083333	23000	0.0	0	23000.000000
6	Calisan	23243354545	Yiğit	Meriç	56	Erkek	Türk	muhasabe	4.750000	15400	0.0	0	19250.000000
12	BeyazYaka	343454657565	Ayşe	Erdoğan	27	Kadın	Türk	muhasabe	6.583333	15600	0.0	2500	18116.333333
8	MaviYaka	23245564545	Melih	Fatma	38	Erkek	Yabancı	diğer	0.008333	16800	0.3	0	16803.000000
5	Calisan	34343434344	Hasan	Yiğit	42	Erkek	Türk	teknoloji	4.916667	5800	0.0	0	15225.000000
0	Insan	151441122	Ali	Mehmet	48	Erkek	Türk	0	0.000000	0	0.0	0	0.000000
1	Insan	4545154784	Ayşe	Fatma	59	Kadın	Türk	0	0.000000	0	0.0	0	0.000000
2	Issiz	23243546234	Aksel	Saatçı	32	Erkek	Türk	0	0.000000	0	0.0	0	0.000000
3	Issiz	345646544545	Zeynep	Fatma	43	Kadın	Türk	0	0.000000	0	0.0	0	0.000000
4	Issiz	231254354545	Batuhan	Doğa	26	Erkek	Türk	0	0.000000	0	0.0	0	0.000000

Daha sonra tecrübesi 3 senden fazla olan beyaz yakalıları bir sorgu ile alıp yeni bir dataframe oluşturuyorum ve bunu ekran yazdırıyorum.

```
-----
Tecrubesi 3 seneden fazla olan beyaz yakalılar
-----
```

	Nesne Değeri	TC No	Ad	Soyad	Yaş	Cinsiyet	Uyruk	Sektör	Tecrübe(Yıl)	Maaş	Yıpranma Payı	Teşvik Primi	Yeni Maaş
12	BeyazYaka	343454657565	Ayşe	Erdoğan	27	Kadın	Türk	muhasabe	6.583333	15600	0.0	2500	18116.333333

Daha sonra yeni maaşı 10000 den fazla olan ve satır olarak 2.ve 4.satır arasında kalanları bulmak için bir sorgu oluşturdum ve sonucunu yeni bir dataframe oluşturup ekrana yazdırıyorum.

```
Yeni maaşı 10000 TL üzerinde olanlar için; 2-5 satır arası olanlar
-----

Empty DataFrame
Columns: [TC No, Yeni Maaş]
Index: []
```

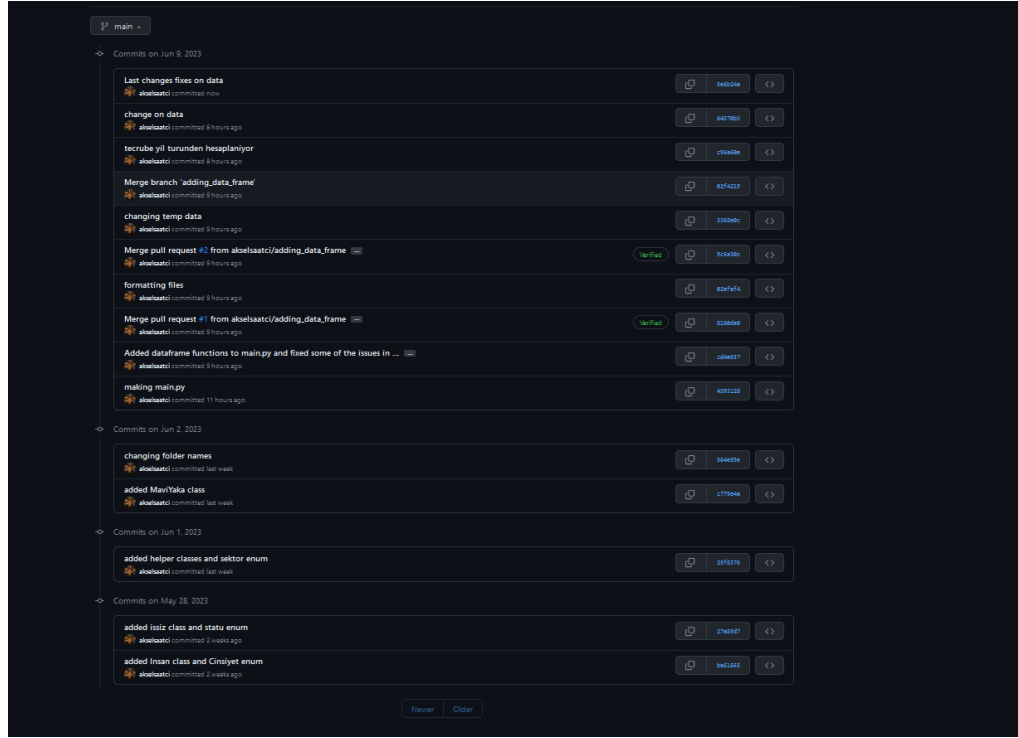
```
-----
Yeni DataFrame
-----
```

	Ad	Soyad	Sektör	Yeni Maaş
0	Ali	Mehmet	0	0.000000
1	Ayşe	Fatma	0	0.000000
2	Aksel	Saatçı	0	0.000000
3	Zeynep	Fatma	0	0.000000
4	Batuhan	Doğa	0	0.000000
5	Hasan	Yiğit	teknoloji	15225.000000
6	Yiğit	Meriç	muhasabe	19250.000000
7	Serkan	Melih	teknoloji	23000.000000
8	Melih	Fatma	diğer	16803.000000
9	Şeyma	Aydın	teknoloji	43005.000000
10	Merve	Kaya	teknoloji	47807.000000
11	Nihal	Kaya	diğer	63000.000000
12	Ayşe	Erdoğan	muhasabe	18116.333333
13	Ümit	Kalem	inşaat	28500.000000

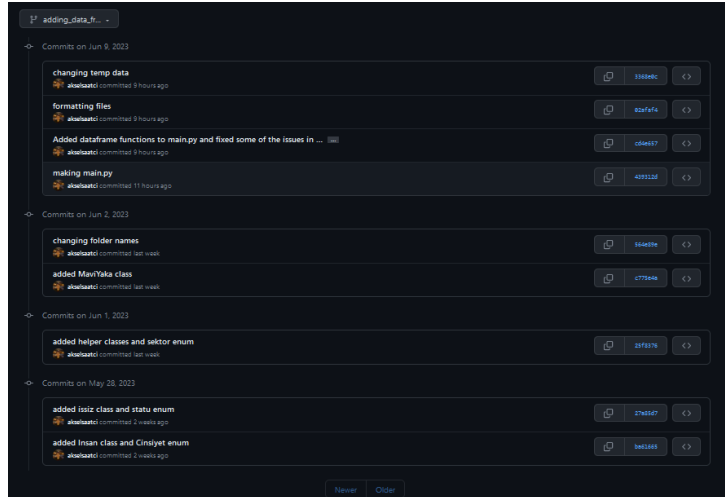
Son olarak sadece Ad, Soyad, Sektör, Yeni Maaş kolonları olan bir dataframe oluşturup bunu ekrana yazdırıyorum.

# GitHub Geçmişi ve Notları

## Main Branch



## adding\_data\_frame Branch



Bu projemi yaparken öncelikle daha farklı bir yapı düşünmüştüm. Fakat bu yapıyı tam olarak sağlayamadığım için making main.py adlı classımdan itibaren bu yapıyı değiştirmek zorunda kaldım. Python'da bir modül import ederken bir klasörden projemin olduğu dizindeki başka bir klasördeki bir modülü import edemediğimden ötürü bu sorunu yaşadım.

GitHub Profilim: <https://github.com/akselsaatci/>

GitHub Projeimin Adre : <https://github.com/akselsaatci/FinalProject>