Yaş Sınırı Tanımayan Beceri: Programlama

Niyazi Doralp, PMP CISA CFE
https://doralp.ca
nick@doralp.ca

Sevgili okurlar, teknoloji çağında yaşıyoruz ve dijital dünya her geçen gün hayatımızın daha merkezi bir parçası haline geliyor. Belki de çoğumuz "programlama" veya "kodlama" kelimelerini duyduğumuzda aklımıza hemen gençler, üniversite öğrencileri ya da erken yaşta bilgisayarla tanışmış kişiler geliyor. Ancak gerçek bundan çok daha farklı. Programlama, yaş sınırı tanımayan, herkesin öğrenebileceği harika bir beceridir.

Programlama Nedir?

Programlama, bilgisayara ne yapması gerektiğini söylemenin bir yoludur. Tıpkı insanlara talimatlar vermek gibi, bilgisayarlara da belirli görevleri nasıl yapacaklarını öğretirsiniz. Bu iletişimi sağlayan araçlar ise programlama dilleridir. Bu dillerden biri olan Python, özellikle yeni başlayanlar için son derece uygun ve öğrenmesi kolay bir dildir.

Yaş Asla Engel Değildir

İlham verici bir örnek olarak, Japonya'nın Kanagawa kentinden 81 yaşındaki Masako Wakamiya'nın hikâyesinden bahsetmek istiyorum. Masako, 58 yaşına kadar hiç bilgisayar kullanmamıştı! Teknoloji konusunda hiçbir geçmiş deneyimi yoktu. Ancak çevresinde bir ihtiyaç fark etti: kendi yaş grubundaki insanlar için uygun mobil uygulamalar yoktu.

Başlangıçta genç uygulama geliştiricilerinden yardım istedi, fakat onlar yaşlıların bir uygulamada ne istediklerini bilmediklerini söyleyince, Masako işi kendi eline almaya karar verdi. "Boşver, kendim yaparım!" diyerek kodlama öğrenmeye başladı ve Hindan adlı oyun uygulamasını geliştirdi. Masako'nun bu girişimi, yaşlı vatandaşların teknolojiyle etkileşime girmelerine, bağlantıda kalmalarına ve yaşlanmayla gelen izolasyonu önlemelerine yardımcı olma misyonuyla başladı.

Python: Başlangıç İçin İdeal Bir Programlama Dili

Python, Guido van Rossum tarafından yaratılmış ve 1991'de piyasaya sürülmüş popüler bir programlama dilidir. Neden özellikle yeni başlayanlar ve ileri yaştakiler için uygundur?

1. **Basit Sözdizimi**: Python, İngilizce diline benzer basit bir sözdizimi kullanır, bu da kodu okumayı ve anlamayı kolaylaştırır.
2. **Çok Platformlu**: Windows, Mac, Linux gibi farklı işletim sistemlerinde çalışabilir.
3. **Az Kodla Çok İş**: Python, diğer programlama dillerine kıyasla daha az satır kodla daha fazla iş yapmanıza olanak tanır.
4. **Okunabilirlik**: Python, okunabilirlik göz önünde bulundurularak tasarlanmıştır.

Python'da Temel Kavramlar

Değişkenler

Değişkenler, programlarınızda veri depolamak için kullanılır. Bir değişkene kısa bir isim (x, y gibi) verebileceğiniz gibi daha açıklayıcı isimler de (isim, yaş, okul) verebilirsiniz. Python'da değişken adları için bazı kurallar vardır:

- Bir değişken adı bir harfle veya alt çizgi karakteriyle başlamalıdır
- Sayıyla başlayamaz
- Sadece alfa-sayısal karakterler ve alt çizgiler içerebilir
- Büyük/küçük harfe duyarlıdır (isim, Isim ve ISIM üç farklı değişkendir)

İşlem Operatörleri

Programlama yaparken çeşitli operatörler kullanırız:

Matematiksel operatörler:

- `+` Toplama
- `-` Çıkarma
- `/` Bölme
- `*` Çarpma
- `=` Değer atama

Karşılaştırma operatörleri:

- `==` Eşit
- `!=` Eşit değil
- `>` Büyük
- `>=` Büyük veya eşit
- `<` Küçük
- `<=` Küçük veya eşit

Koşullu İfadeler ve Döngüler

if İfadeleri:

```python
if karşılaştırma:
    # Doğru ise bunu yap
else:
    # Değilse bunu yap
```

while Döngüleri:

```python
while karşılaştırma:
    # Yapılacak işlemler
```

Neden İleri Yaşta Programlama Öğrenmeli?

1. **Zihinsel Aktivite**: Programlama, beyninizi aktif tutar ve problem çözme yeteneklerinizi geliştirir.
2. **Yeni Dünyalar Keşfetme**: Teknoloji dünyasına daha aktif katılmanızı sağlar.
3. **İhtiyaçlara Cevap Verme**: Masako gibi, kendi yaş grubunuz için çözümler üretebilirsiniz.
4. **Sosyal Bağlantı**: Programlama toplulukları aracılığıyla yeni insanlarla tanışabilirsiniz.
5. **Yaratıcılığı Teşvik Etme**: Kendi projelerinizi hayata geçirerek yaratıcılığınızı kullanabilirsiniz.

Nasıl Başlanır?

1. **Online Kaynaklar**: http://online-python.com gibi web sitelerinden Python kodlamayı deneyebilirsiniz.
2. **Kurslar**: Çeşitli online platformlarda ücretsiz veya ücretli programlama kursları bulabilirsiniz.
3. **Topluluklar**: Programlama öğrenen diğer kişilerle bağlantı kurarak motivasyonunuzu yüksek tutabilirsiniz.
4. **Basit Projeler**: Hesap makinesi, basit oyunlar gibi küçük projelerle başlayabilirsiniz.

Sonuç

Programlama öğrenmek için asla geç değildir. Yaşınız, geçmiş deneyimleriniz veya eğitim seviyeniz ne olursa olsun, tek gereken şey öğrenme isteğidir. Masako Wakamiya'nın da

gösterdiği gibi, 81 yaşında bile yeni bir beceri edinmek ve bu beceriyi topluma fayda sağlamak için kullanmak mümkün.

Teknoloji dünyasına adım atmak ve kendi dijital çözümlerinizi üretmeye başlamak için bugün harekete geçin. Programlama, size yeni kapılar açacak ve dünyayı farklı bir perspektiften görmenizi sağlayacak harika bir yolculuktur.

Retry

Claude does not have the ability to run the code it generates yet. Claude does not have internet access. Links provided may not be accurate or up to date.

N

daha uzun ve detaylı yazabilirmisin

Edit

Yaş Sınırı Tanımayan Beceri: Programlama ve Python Dünyasına Giriş

Giriş: Programlama ve Yaş İlişkisi

Dijital çağın hızla ilerleyen dünyasında, teknoloji artık hayatımızın vazgeçilmez bir parçası haline geldi. Akıllı telefonlar, bilgisayarlar, tabletler ve hatta akıllı ev aletleri günlük yaşantımızın merkezinde yer alıyor. Bu teknolojik araçların arkasında ise "programlama" veya diğer adıyla "kodlama" yatıyor. Birçok kişi programlamayı sadece gençlerin, bilgisayar mühendislerinin veya erken yaşlardan itibaren teknolojiye maruz kalmış kişilerin yapabileceği bir iş olarak görüyor. Ancak, gerçek bundan çok daha farklı.

Programlama, yaş sınırı tanımayan, her yaştan insanın öğrenebileceği ve ustalaşabileceği bir beceridir. Elimizdeki bilgiler ve örnekler gösteriyor ki, yaşamlarının ilerleyen dönemlerinde programlamayı öğrenen birçok kişi, başarılı teknoloji kariyerlerine sahip olabiliyorlar ve dijital dünyada kendi izlerini bırakabiliyorlar.

Programlama Nedir? Temel Anlayış

Programlama, özünde bilgisayara ne yapması gerektiğini söylemenin sistematik bir yoludur. Tıpkı insanlara bir tarif verirken adım adım talimatlar sunduğumuz gibi, bilgisayarlara da belirli görevleri nasıl yerine getireceğini "öğretiriz". Bu iletişimi sağlayan araçlar ise programlama dilleridir.

Programlama dilleri, insan dili ile bilgisayarın anlayabileceği makine dili arasında bir köprü görevi görür. Her biri farklı amaçlar için tasarlanmış onlarca programlama dili bulunmaktadır, ancak bunların arasında Python, özellikle yeni başlayanlar ve ileri yaştaki öğrenenler için öne çıkan bir dildir.

İlham Verici Bir Örnek: Masako Wakamiya'nın Hikâyesi

Programlamanın her yaşta öğrenilebilecek bir beceri olduğunun en çarpıcı örneklerinden biri, Japonya'nın Kanagawa kentinden gelen Masako Wakamiya'dır. Masako'nun hikâyesi, yaşın sadece bir sayıdan ibaret olduğunu ve öğrenmenin ömür boyu devam eden bir süreç olduğunu kanıtlar niteliktedir.

Masako Wakamiya, 58 yaşına kadar hiç bilgisayar kullanmamıştı. Teknoloji konusunda hiçbir geçmiş deneyimi yoktu. Emeklilik döneminde, toplumda önemli bir boşluk fark etti: kendi yaş grubundaki insanlar için tasarlanmış, kullanımı kolay ve anlamlı mobil uygulamalar eksikti.

Başlangıçta, ihtiyaç duyduğu uygulamaları geliştirmek için genç programcılardan yardım istedi. Ancak onlar, yaşlı kullanıcıların ihtiyaçlarını ve tercihlerini tam olarak anlayamadıklarını itiraf ettiler. İşte bu noktada Masako, "Boşver, kendim yaparım!" diyerek kendi eline almaya karar verdi.

81 yaşında, kendi yaş grubu için Hindan adlı bir oyun uygulaması geliştirdi. Bu uygulama, Japon kültüründeki geleneksel bir oyunun dijital versiyonuydu ve yaşlı kullanıcıların kognitif becerilerini geliştirmeyi amaçlıyordu. Masako'nun bu girişimi, sadece bir uygulama geliştirmekten öte, yaşlı vatandaşların teknolojiyle etkileşime girmelerine, sosyal bağlantılarını sürdürmelerine ve yaşlanmayla gelen izolasyonu önlemelerine yardımcı olma misyonuna dönüştü.

Masako'nun hikâyesi bize gösteriyor ki, programlama öğrenmek için asla geç değildir. Yeni beceriler edinmek, yeni zorluklarla yüzleşmek ve teknolojinin sunduğu fırsatlardan yararlanmak için yaşın hiçbir önemi yoktur. Tek gereken şey, öğrenme isteği ve azmidir.

Python: İleri Yaş İçin İdeal Programlama Dili

Python, Guido van Rossum tarafından 1991 yılında geliştirilmiş bir programlama dilidir. İleri yaştaki öğrenenler için neden özellikle uygun olduğunu anlamak için Python'un temel özelliklerine bakalım:

1. Okunabilirlik ve Anlaşılabilirlik

Python, okunabilirlik ve anlaşılabilirlik ön planda tutularak tasarlanmıştır. Sözdizimi (syntax), İngilizce diline benzer bir yapıya sahiptir ve matematiksel ifadelerden etkilenmiştir. Bu, programlama diliyle yeni tanışan birinin kodu okuyup anlamasını kolaylaştırır. Örneğin, Python'da bir "Merhaba Dünya" programı şu şekilde yazılabilir:

python
Copy
```
print("Merhaba Dünya!")
```

Bu kod, ekrana "Merhaba Dünya!" yazısını yazdıracaktır. Gördüğünüz gibi, kod oldukça sade ve anlaşılırdır.

2. Çok Platformlu Çalışma

Python, Windows, Mac, Linux gibi farklı işletim sistemlerinde çalışabilir. Bu, hangi bilgisayarı kullanırsanız kullanın Python kodlarınızın çalışabileceği anlamına gelir. Bu esneklik, özellikle farklı cihazlarla çalışan kullanıcılar için büyük bir avantaj sağlar.

3. Az Kod, Çok İş

Python, diğer birçok programlama diline kıyasla daha az kod yazarak daha fazla iş yapmanıza olanak tanır. Bu, özellikle yazma hızı veya el-göz koordinasyonu gibi fiziksel faktörlerin önem kazanabileceği ileri yaştaki öğrenenler için büyük bir avantajdır. Aynı işi yapmak için Python'da belki 10 satır kod yazarken, başka bir dilde 30-40 satır yazmanız gerekebilir.

4. Geniş Kütüphane Desteği

Python, çok geniş bir standart kütüphane setiyle birlikte gelir. Bu kütüphaneler, web geliştirmeden veri analizine, yapay zekâdan oyun geliştirmeye kadar birçok alanda hazır araçlar sunar. Bu, sıfırdan her şeyi kendiniz yazmanız gerekmediği anlamına gelir. Mevcut araçları kullanarak hızla ilerleyebilir ve projelerinizi geliştirebilirsiniz.

5. Güçlü Topluluk Desteği

Python, dünya çapında milyonlarca geliştirici tarafından kullanılan popüler bir dildir. Bu, çevrimiçi forumlarda, sosyal medyada ve çeşitli platformlarda sorularınıza cevap bulabileceğiniz, zorluklarla karşılaştığınızda yardım alabileceğiniz geniş bir topluluk olduğu anlamına gelir. Bu destek ağı, özellikle yeni başlayanlar için paha biçilmezdir.

Python'da Temel Programlama Kavramları

Programlama öğrenmeye başlarken, bazı temel kavramları anlamak önemlidir. Python'da en temel kavramlardan bazıları şunlardır:

1. Değişkenler ve Veri Tipleri

Değişkenler, programlarınızda veri depolamak için kullanılan "kutu" veya "kap" gibi düşünülebilir. Her değişkenin bir adı ve içinde saklanan bir değeri vardır. Python'da değişken tanımlamak oldukça basittir:

python
Copy
```
isim = "Ahmet"
yas = 65
boy = 1.75
emekli_mi = True
```

Yukarıdaki örnekte, dört farklı değişken tanımladık:

- `isim`: Metin (string) veri tipinde ve değeri "Ahmet"
- `yas`: Tam sayı (integer) veri tipinde ve değeri 65
- `boy`: Ondalıklı sayı (float) veri tipinde ve değeri 1.75
- `emekli_mi`: Boolean (True/False) veri tipinde ve değeri True

Python'da değişken adlandırma kuralları şunlardır:

- Bir değişken adı harf veya alt çizgi (_) ile başlamalıdır
- Bir değişken adı sayı ile başlayamaz
- Bir değişken adı sadece alfa-sayısal karakterler (A-Z, a-z, 0-9) ve alt çizgi içerebilir
- Değişken adları büyük/küçük harfe duyarlıdır (isim, Isim ve ISIM üç farklı değişkendir)

2. İşlem Operatörleri

Programlama yaparken, değişkenler üzerinde çeşitli işlemler gerçekleştirmek için operatörler kullanırız. Python'da yaygın olarak kullanılan operatörler şunlardır:

Matematiksel operatörler:

- `+` Toplama: `a + b`
- `-` Çıkarma: `a - b`
- `*` Çarpma: `a * b`
- `/` Bölme: `a / b`
- `%` Mod (kalan): `a % b`
- `**` Üs alma: `a ** b`
- `//` Taban bölme: `a // b`

Atama operatörleri:

- `=` Değer atama: `a = 5`
- `+=` Toplama ve atama: `a += 3` (a = a + 3 ile aynıdır)
- `-=` Çıkarma ve atama: `a -= 3` (a = a - 3 ile aynıdır)
- `*=` Çarpma ve atama: `a *= 3` (a = a * 3 ile aynıdır)
- `/=` Bölme ve atama: `a /= 3` (a = a / 3 ile aynıdır)

Karşılaştırma operatörleri:

- `==` Eşit: `a == b`
- `!=` Eşit değil: `a != b`
- `>` Büyük: `a > b`
- `<` Küçük: `a < b`
- `>=` Büyük veya eşit: `a >= b`
- `<=` Küçük veya eşit: `a <= b`

Mantıksal operatörler:

- `and` Ve: `a and b` (her iki koşul da doğruysa True)
- `or` Veya: `a or b` (en az bir koşul doğruysa True)
- `not` Değil: `not a` (a'nın değilini alır)

Basit bir örnek üzerinden bu operatörleri görelim:

python
Copy

```
a = 10
b = 5

toplam = a + b          # 15
fark = a - b            # 5
carpim = a * b          # 50
bolum = a / b           # 2.0
kalan = a % b           # 0
us = a ** b             # 100000 (10^5)
taban_bolum = a // b    # 2

a_buyuk_mu = a > b      # True
esit_mi = a == b        # False
```

3. Koşullu İfadeler (if/else)

Koşullu ifadeler, programınızın belirli koşullara göre farklı işlemler yapmasını sağlar. Python'da koşullu ifadeler şu şekilde kullanılır:

python
Copy

```
yas = 65

if yas >= 65:
    print("Emeklilik yaşındasınız.")
```

```
else:
    print("Emeklilik yaşında değilsiniz.")
```

Yukarıdaki örnekte, `yas` değişkeni 65 veya daha büyükse "Emeklilik yaşındasınız." mesajı, değilse "Emeklilik yaşında değilsiniz." mesajı ekrana yazdırılacaktır.

Birden fazla koşul için `elif` (else if kısaltması) kullanabiliriz:

python
Copy
```
yas = 65

if yas < 18:
    print("Gençsiniz.")
elif yas < 65:
    print("Yetişkinsiniz.")
else:
    print("Emeklilik yaşındasınız.")
```

4. Döngüler (while ve for)

Döngüler, belirli bir işlemi tekrar tekrar gerçekleştirmek istediğimizde kullanılır. Python'da iki temel döngü türü vardır: `while` ve `for`.

While Döngüsü: While döngüsü, belirtilen bir koşul doğru olduğu sürece bir dizi işlemi tekrarlar.

python
Copy
```
sayac = 1
while sayac <= 5:
    print(f"Sayac: {sayac}")
    sayac += 1
```

Bu kod, sayaç 1'den 5'e kadar her değer için "Sayac: X" (X sayacın değeridir) yazdıracaktır.

For Döngüsü: For döngüsü, genellikle bir koleksiyon üzerinde (liste, dizi, vb.) dolaşmak için kullanılır.

python
Copy
```
meyveler = ["elma", "armut", "muz", "çilek"]
for meyve in meyveler:
    print(f"Sepetteki meyve: {meyve}")
```

Bu kod, `meyveler` listesindeki her meyve için "Sepetteki meyve: X" (X meyvenin adıdır) yazdıracaktır.

Ayrıca, `range()` fonksiyonu ile belirli bir aralıkta döngü oluşturabiliriz:

python
Copy
```
for i in range(1, 6):  # 1'den 5'e kadar
    print(f"Sayı: {i}")
```

Bu kod, 1'den 5'e kadar her sayı için "Sayı: X" (X sayının değeridir) yazdıracaktır.

5. Fonksiyonlar

Fonksiyonlar, belirli bir görevi yerine getiren ve gerektiğinde tekrar tekrar çağrılabilen kod bloklarıdır. Fonksiyonlar, kodunuzu daha organize, okunabilir ve yeniden kullanılabilir hale getirir.

Python'da fonksiyon tanımlamak için `def` anahtar kelimesi kullanılır:

python
Copy
```
def selamla(isim):
    """Bu fonksiyon, verilen isimle bir selamlama mesajı
oluşturur."""
    mesaj = f"Merhaba, {isim}! Python öğrenmeye hoş geldin!"
    return mesaj

# Fonksiyonu çağırma
sonuc = selamla("Ayşe")
print(sonuc)  # "Merhaba, Ayşe! Python öğrenmeye hoş geldin!"
yazdırır
```

Fonksiyonlar, parametre almak zorunda değildir ve bir değer döndürmek zorunda da değildir:

python
Copy
```
def gunaydin():
    """Bu fonksiyon, basit bir günaydın mesajı yazdırır."""
    print("Günaydın! Bugün harika bir gün olacak!")

# Fonksiyonu çağırma
gunaydin()  # "Günaydın! Bugün harika bir gün olacak!" yazdırır
```

Python'u Uygulamada Kullanma: Basit Projeler

Teorik bilgiler önemlidir, ancak programlama öğrenmenin en etkili yolu, öğrendiklerinizi uygulayarak deneyim kazanmaktır. İşte ileri yaştaki öğrenenler için basit Python projeleri:

1. Hesap Makinesi

Toplama, çıkarma, çarpma ve bölme gibi temel matematiksel işlemleri yapabilen basit bir hesap makinesi:

python
Copy

```
def toplama(x, y):
    return x + y

def cikarma(x, y):
    return x - y

def carpma(x, y):
    return x * y

def bolme(x, y):
    if y == 0:
        return "Sıfıra bölme hatası!"
    return x / y

print("Basit Hesap Makinesi")
print("1. Toplama")
print("2. Çıkarma")
print("3. Çarpma")
print("4. Bölme")

secim = input("Yapmak istediğiniz işlemi seçin (1/2/3/4): ")
sayi1 = float(input("Birinci sayıyı girin: "))
sayi2 = float(input("İkinci sayıyı girin: "))

if secim == '1':
    print(f"{sayi1} + {sayi2} = {toplama(sayi1, sayi2)}")
elif secim == '2':
    print(f"{sayi1} - {sayi2} = {cikarma(sayi1, sayi2)}")
elif secim == '3':
    print(f"{sayi1} * {sayi2} = {carpma(sayi1, sayi2)}")
elif secim == '4':
    print(f"{sayi1} / {sayi2} = {bolme(sayi1, sayi2)}")
else:
    print("Geçersiz giriş!")
```

2. Sağlık Takip Uygulaması

Günlük adım sayısı, su tüketimi veya ilaç hatırlatıcısı gibi sağlık verilerini takip eden basit bir uygulama:

python
Copy

```
def adim_kaydet(gun, adim_sayisi):
    """Günlük adım sayısını kaydeder."""
    with open("adim_takip.txt", "a") as dosya:
        dosya.write(f"{gun}: {adim_sayisi} adım\n")
    print(f"{gun} günü için {adim_sayisi} adım kaydedildi.")

def su_kaydet(gun, bardak_sayisi):
    """Günlük su tüketimini bardak cinsinden kaydeder."""
    with open("su_takip.txt", "a") as dosya:
        dosya.write(f"{gun}: {bardak_sayisi} bardak su\n")
    print(f"{gun} günü için {bardak_sayisi} bardak su kaydedildi.")

def ilac_hatirlatici(ilac_adi, saat):
    """İlaç hatırlatıcısı ekler."""
    with open("ilac_hatirlatici.txt", "a") as dosya:
        dosya.write(f"{ilac_adi}: Her gün saat {saat}\n")
    print(f"{ilac_adi} ilacı için her gün saat {saat} hatırlatıcısı
eklendi.")

print("Sağlık Takip Uygulaması")
print("1. Adım Sayısı Kaydet")
print("2. Su Tüketimi Kaydet")
print("3. İlaç Hatırlatıcısı Ekle")

secim = input("Yapmak istediğiniz işlemi seçin (1/2/3): ")

if secim == '1':
    gun = input("Hangi gün için kayıt yapacaksınız? (örn.
01/03/2023): ")
    adim_sayisi = int(input("Adım sayısını girin: "))
    adim_kaydet(gun, adim_sayisi)
elif secim == '2':
    gun = input("Hangi gün için kayıt yapacaksınız? (örn.
01/03/2023): ")
    bardak_sayisi = int(input("Bardak sayısını girin: "))
    su_kaydet(gun, bardak_sayisi)
elif secim == '3':
    ilac_adi = input("İlaç adını girin: ")
```

```
    saat = input("Hangi saatte alınacak? (örn. 08:00): ")
    ilac_hatirlatici(ilac_adi, saat)
else:
    print("Geçersiz giriş!")
```

3. Kişisel Finans Takip Uygulaması

Gelir ve giderleri takip eden, bütçe oluşturmaya yardımcı olan basit bir uygulama:

python
Copy

```
def gelir_ekle(tarih, kategori, miktar):
    """Gelir ekler."""
    with open("finansal_takip.txt", "a") as dosya:
        dosya.write(f"GELİR|{tarih}|{kategori}|{miktar}\n")
    print(f"{tarih} tarihinde {kategori} kategorisinde {miktar} TL
gelir eklendi.")

def gider_ekle(tarih, kategori, miktar):
    """Gider ekler."""
    with open("finansal_takip.txt", "a") as dosya:
        dosya.write(f"GİDER|{tarih}|{kategori}|{miktar}\n")
    print(f"{tarih} tarihinde {kategori} kategorisinde {miktar} TL
gider eklendi.")

def rapor_olustur():
    """Finansal rapor oluşturur."""
    toplam_gelir = 0
    toplam_gider = 0
    kategori_giderler = {}

    try:
        with open("finansal_takip.txt", "r") as dosya:
            for satir in dosya:
                parcalar = satir.strip().split("|")
                if len(parcalar) == 4:
                    tip, tarih, kategori, miktar = parcalar
                    miktar = float(miktar)

                    if tip == "GELİR":
                        toplam_gelir += miktar
                    elif tip == "GİDER":
                        toplam_gider += miktar
                        if kategori in kategori_giderler:
                            kategori_giderler[kategori] += miktar
```

```
                        else:
                            kategori_giderler[kategori] = miktar
    except FileNotFoundError:
        print("Finansal takip dosyası bulunamadı. Önce gelir/gider
ekleyin.")
        return

    print("\n--- FİNANSAL RAPOR ---")
    print(f"Toplam Gelir: {toplam_gelir} TL")
    print(f"Toplam Gider: {toplam_gider} TL")
    print(f"Net Durum: {toplam_gelir - toplam_gider} TL")

    print("\n--- KATEGORİYE GÖRE GİDERLER ---")
    for kategori, miktar in kategori_giderler.items():
        print(f"{kategori}: {miktar} TL")

print("Kişisel Finans Takip Uygulaması")
print("1. Gelir Ekle")
print("2. Gider Ekle")
print("3. Rapor Oluştur")

secim = input("Yapmak istediğiniz işlemi seçin (1/2/3): ")

if secim == '1':
    tarih = input("Tarih girin (örn. 01/03/2023): ")
    kategori = input("Gelir kategorisi girin (örn. Maaş, Kira
Geliri): ")
    miktar = float(input("Miktar girin (TL): "))
    gelir_ekle(tarih, kategori, miktar)
elif secim == '2':
    tarih = input("Tarih girin (örn. 01/03/2023): ")
    kategori = input("Gider kategorisi girin (örn. Market,
Faturalar): ")
    miktar = float(input("Miktar girin (TL): "))
    gider_ekle(tarih, kategori, miktar)
elif secim == '3':
    rapor_olustur()
else:
    print("Geçersiz giriş!")
```

İleri Yaşta Programlama Öğrenmenin Faydaları

Programlama öğrenmek, ileri yaştaki bireyler için birçok fayda sağlar. Bu faydaların bazıları şunlardır:

1. Zihinsel Aktivite ve Bilişsel Sağlık

Programlama, mantık, problem çözme ve analitik düşünme becerileri gerektirir. Bu tür zihinsel aktiviteler, beyni aktif tutar ve bilişsel yeteneklerin korunmasına yardımcı olabilir. Araştırmalar, beyin jimnastiği olarak değerlendirilebilecek aktivitelerin, yaşlanmayla ilişkili bilişsel gerilemeyi yavaşlatabileceğini göstermektedir.

2. Yeni Yetenekler ve Öz Güven

Yeni bir beceri edinmek, her yaşta insanın özgüvenini artırır. Programlama gibi teknik bir beceriyi öğrenmek ve projeler geliştirmek, "ben yapabilirim" duygusunu güçlendirir ve kişinin kendine olan inancını artırır.

3. Teknoloji Dünyasına Daha İyi Entegrasyon

Günümüzde teknoloji, hayatımızın her alanında yer alıyor. Programlama öğrenmek, dijital dünyayı daha iyi anlamanıza ve onunla daha etkili bir şekilde etkileşime girmenize yardımcı olur. Bu, yaşam kalitenizi artırabilir ve teknolojiyle ilgili korkularınızı azaltabilir.

4. Sosyal Bağlantı ve Topluluk Hissi

Programlama öğrenen kişiler, çevrimiçi forumlar, sosyal medya grupları ve yerel topluluklar aracılığıyla benzer ilgi alanlarına sahip diğer kişilerle tanışabilirler. Bu sosyal bağlantılar, izolasyonu azaltabilir ve ortak bir amaca hizmet eden bir topluluğun parçası olma hissini güçlendirebilir.

5. Yaratıcılık ve Kişisel Tatmin

Kendi yazılım projelerinizi geliştirmek, yaratıcı düşünmeyi teşvik eder ve kişisel tatmin sağlar. Bir sorunu çözen veya hayatınızı kolaylaştıran bir uygulama veya program yazmak, büyük bir başarı duygusu ve memnuniyet getirebilir.

6. Potansiyel Ekonomik Faydalar

İleri yaşta programlama öğrenmek, freelance iş fırsatları, yarı zamanlı danışmanlık veya kendi dijital ürünlerinizi satma gibi gelir getirici faaliyetlere kapı açabilir. Bu, emeklilik gelirini destekleyebilir veya boş zamanları değerlendirmenin anlamlı bir yolu olabilir.

Programlamaya Başlamak İçin Pratik Adımlar

İleri yaşta programlama öğrenmeye karar verdiyseniz, işte başlamanıza yardımcı olacak bazı pratik adımlar:

1. Doğru Araçları Edinmek

Programlamaya başlamak için ihtiyacınız olan temel araçlar şunlardır:

- **Bilgisayar**: Temel bir laptop veya masaüstü bilgisayar yeterlidir. En son teknoloji ürünü olması gerekmez.
- **İnternet Bağlantısı**: Öğrenme kaynaklarına erişmek, sorunlarınıza çözüm aramak ve toplulukla etkileşime geçmek için.
- **Metin Editörü veya Entegre Geliştirme Ortamı (IDE)**: Python için Thonny, IDLE, VS Code gibi ücretsiz seçenekler mevcuttur.
- **Python Yükleyicisi**: Python'u bilgisayarınıza yüklemek için python.org adresinden indirin.

2. Çevrimiçi Kaynakları Kullanmak

Python öğrenmek için birçok ücretsiz ve ücretli çevrimiçi kaynak bulunmaktadır:

- **Online Python Editörleri**: http://online-python.com gibi siteler, herhangi bir şey yüklemeden Python kodlarını denemenize olanak tanır.
- **Online Kurslar**: Udemy, Coursera, Khan Academy gibi platformlarda çeşitli Python kursları bulunmaktadır.
- **YouTube

Niyazi (Nick) DORALP Kimdir?

1974 Ekim ayında ilk programımı yazdım. TED Ankara Koleji ve Boğaziçi Üniversitesinden mezunum. Hala aynı zevkle ve heyecan ile hem öğreniyorum hem kodluyorum. Yaklaşık 40 yıl Kanada'da yaşadım ve en büyük bankalarının Bilgi İşlem departmanlarında teknik görevlerde ve yöneticilik yaptım. Bu günlerde sivil toplum kuruluşlarına web siteleri hazırlamak ve iOS (iPhone) ve Android için app geliştiriyorum. 15 adet yazdığım app şu an hem Apple App Store hem Google Play Store da bulunuyor.
Bazı uzmanlık belgelerim:
PMP - Project Management Professional: *Proje Yönetimi Profesyoneli*
CISA - Certified Information Systems Auditor: *Sertifikalı Bilgi Sistemleri Denetçisi*
CFE - Certified Fraud Examiner: *Sertifikalı Dolandırıcılık İnceleme Uzmanı*
EDP - Electronic Document Professional: *Elektronik Belge Profesyoneli*