Python Programlama Dili Ders Notları

BÖLÜM 1: PROGRAMLAMAYA VE PYTHON'A GİRİŞ

1. Temel Kavramlar

- **Programlama:** Günlük hayattaki problemlerin ve bu problemlerin çözümlerinin bilgisayarlara (makinelere) öğretilmesi işidir. Programlama, **programlama dilleri** aracılığıyla yapılır.
- **Program:** Problemlerin çözümünü makinelere öğreten, birbirini izleyen komut ve yönergeler bütünüdür. *(Örnek: Word, Excel, Chrome, Photoshop vb.)*
- **Kullanım Alanları:** Masaüstü yazılımları, web uygulamaları, mobil uygulamalar, oyun geliştirme, işletim sistemleri, gömülü sistemler, yapay zekâ, makine öğrenmesi ve veri madenciliği.

2. Python Programlama Dili Nedir?

1991 yılında Hollandalı programcı **Guido van Rossum** tarafından geliştirilmiş, popüler ve yüksek seviyeli bir programlama dilidir. Google, YouTube, NASA, Netflix, Instagram ve Dropbox gibi kuruluşlar tarafından aktif olarak kullanılmaktadır.

- **Platform Bağımsızdır:** Windows, Linux, MacOS, Raspberry Pi gibi farklı işletim sistemlerinde çalışabilir.
- **Öğrenmesi ve Yazması Kolaydır:** Söz dizimi (syntax) İngilizceye benzerdir ve diğer dillere göre çok daha az kod yazılır.
- **Yorumlayıcı (Interpreter) Kullanır:** Kodlar derlenmeye gerek kalmadan yazıldığı gibi anında çalıştırılabilir.
- **Zengin Kütüphanelidir:** Yapay zekâ, bilimsel hesaplama, ağ ve web programlama için hazır modüllere sahiptir.

3. Kod Yazma Araçları: Editör ve IDE

- **Metin Editörü (Kod Düzenleyici):** Sadece kaynak kodu yazmak ve düzenlemek için kullanılan sade ve hızlı araçlardır. Kod renklendirme ve tamamlama sunar; birçok dil ile kullanılabilir. *(Örnekler: Notepad++, Sublime Text, Atom, VS Code)*
- **IDE (Entegre Geliştirme Ortamı):** Kod yazmanın yanında derleyici, yorumlayıcı ve hata ayıklayıcı (debugger) gibi araçları tek çatıda birleştiren komple yazılımlardır. *(Örnekler: **IDLE** — Python kurulumuyla gelen kendi IDE'sidir, PyCharm, Spyder, Jupyter Notebook, Anaconda)*

4. Programların Çalıştırılma Yöntemleri

İnsanların yazdığı kaynak kodların, bilgisayarın anlayacağı makine diline (0 ve 1'lere) çevrilmesi iki şekilde yapılır:

Özellik	Derleyici (Compiler)	Yorumlayıcı (Interpreter)
Çalışma Şekli	Kodun **tamamını** tek seferde kontrol eder ve makine diline çevirir.	Kodu **satır satır** okur, makine diline çevirir ve anında çalıştırır.
Hata Kontrolü	Tüm koddaki hataları bulup topluca bildirir.	Hata olan satıra kadar kodu çalıştırır, hatayı görünce durur.
Çıktı Dosyası	Çalıştırılabilir bir dosya (örneğin `.exe`) üretir.	Ayrı bir çalıştırılabilir dosya üretmez.
Hız ve Örnekler	Daha hızlı çalışır. *(C, C++, C#, Pascal)*	Derlenen dillere göre biraz yavaştır. *(**Python**, JavaScript, PHP)*

5. Python Kodlarını Çalıştırma Yöntemleri

1. **Komut İstemi (CMD) ile:** `python.org` adresinden Python kurulduktan sonra komut satırına Windows ortamında `>>python dosya_adı` yazılarak, Pardus ortamında `>>python3 dosya_adı` yazılarak kod çalıştırılır.
2. **IDLE ile:** Python kurulumuyla beraber gelen kendi geliştirme ortamıdır. Hem etkileşimli kabukta (Shell) hem de `.py` uzantılı dosya oluşturularak kod yazılır.
3. **Çevrimiçi (Online) Editörler ile:** Bilgisayara kurulum gerektirmeden tarayıcı üzerinden kod çalıştırmayı sağlar. Örneğin **Replit** üzerinde *New Repl* ile proje açılıp `main.py` dosyasına kod yazılır ve *Run* ile çalıştırılır. *(Diğerleri: Programiz, Jupyter Notebook, W3Schools)*

```
# İlk Python Kodumuz:
print("Merhaba Dünya!")   # Ekran Çıktısı: Merhaba Dünya!
```

BÖLÜM 2: VERİ TİPLERİ, OPERATÖRLER VE DEĞİŞKENLER

1. Python'da Temel Veri Tipleri

Programlamada kullanılan veriler türlerine göre sınıflandırılır. Bir verinin veya değişkenin tipini öğrenmek için `type()` fonksiyonu kullanılır.

- **Tam Sayılar — `int` (Integer):** Pozitif veya negatif tam sayıları ifade eder. *(Örnek: `5`, `-12`, `0`, `100`)*
- **Ondalıklı Sayılar — `float`:** Kesirli sayıları ifade eder. Ondalık ayracı olarak **nokta (`.`)** kullanılır. *(Örnek: `2.5`, `-3.14`, `4.0`)*
- **Metinsel İfadeler — `str` (String):** Harf, kelime veya karakter dizilerinden oluşan sözel verilerdir. **Tek tırnak (`' '`)** veya **çift tırnak (`" "`)** içine alınır. *(Örnek: `'Ali'`, `"Bilişim 9A"`)*

```
print(type(5))       # Çıktı: <class 'int'>   (Tam sayı)
print(type(2.5))     # Çıktı: <class 'float'> (Ondalıklı sayı)
print(type("Ali"))   # Çıktı: <class 'str'>   (Metinsel ifade)
```

2. Matematiksel (Aritmetik) Operatörler

Operatör	İşlem Adı	Örnek	Sonuç	Açıklama
`+`	Toplama	`10 + 15`	`25`	İki sayıyı toplar.
`-`	Çıkarma	`15 - 10`	`5`	Birinci sayıdan ikinciyi çıkarır.
`*`	Çarpma	`2 * 5`	`10`	İki sayıyı çarpar.
`/`	Bölme	`8 / 2`	`4.0`	Bölme yapar. **Sonuç daima `float` çıkar.**
`//`	Kalansız (Tam) Bölme	`15 // 2`	`7`	Bölümün sadece tam kısmını alır, küsüratı atar.
`%`	Mod Alma (Kalan)	`15 % 2`	`1`	Bölme işleminden **kalan** sayıyı verir.
`**`	Üs Alma	`2 ** 3`	`8`	Üslü sayı hesaplar ($2^3 = 8$).

3. Değişken (Variable) Kavramı

Program çalışırken verileri bilgisayar belleğinde (RAM) geçici olarak saklamak için oluşturduğumuz alanlara **değişken** denir. Değişken tanımlamak için `=` **(atama operatörü)** kullanılır.

```
ad = "Ali"          # str (metin) türünde değişken
soyad = 'AY'        # str (metin) türünde değişken
x = 10              # int (tam sayı) türünde değişken
y = 20              # int (tam sayı) türünde değişken
toplam = x + y      # x ve y değerleri toplanıp 'toplam' değişkenine atanır
ortalama = 2.3      # float (ondalıklı sayı) türünde değişken

print(ad)           # Çıktı: Ali
print(toplam)       # Çıktı: 30
print(ortalama)     # Çıktı: 2.3
```

> **Uyarı:** Değer atanmamış (tanımlanmamış) bir değişken ekrana yazdırılmaya çalışılırsa Python `NameError: name '...' is not defined` (isim tanımlanmadı) hatası verir.

4. Değişken İsimlendirme Kuralları

1. **Rakam ile başlayamaz:** Harf veya alt çizgi (`_`) ile başlamalıdır. **1yas = 15 (Hatalı)** | **yas1 = 15 (Geçerli)**
2. **Özel karakter ve boşluk kullanılamaz:** Boşluk ve `. , / * - + ? ! @` gibi işaretler kullanılamaz. Kelimeleri ayırmak için sadece **alt çizgi (`_`)** kullanılır. **ogrenci-no (Hatalı)** | **ogrenci_no (Geçerli)**
3. **Python komut kelimeleri kullanılamaz:** `if`, `for`, `while`, `print`, `True`, `class` gibi özel anahtar kelimeler değişken adı yapılamaz.
4. **Büyük-küçük harf duyarlılığı (Case Sensitive) vardır:** `isim = "Ali"` ile `Isim = "Ay"` birbirinden tamamen farklı iki değişkendir.
5. **Türkçe karakter kullanılmaması önerilir:** Yazılım standartları gereği `ç, ğ, ı, ö, ş, ü` yerine İngilizce karakterler tercih edilmelidir. *(Örnek: `ogrenci_yasi`)*

BÖLÜM 3: KARŞILAŞTIRMA VE MANTIKSAL OPERATÖRLER

1. Mantıksal Veri Tipi: `bool` (Boolean)

Karşılaştırma ve mantıksal işlemlerin sonucu her zaman **`bool` (Mantıksal)** veri tipinde döner ve yalnızca iki değer alabilir: **`True`** (Doğru) veya **`False`** (Yanlış). Python'da ilk harfleri mutlaka büyük yazılır.

2. Karşılaştırma Operatörleri

İki değeri veya değişkeni karşılaştırmak için kullanılır. *(Örneklerde `notum = 70`, `gecme_notu = 50` ve `sifre = "1234"` kabul edilmiştir.)*

Operatör	Anlamı	Kod Örneği	Çıktı	Açıklama
`==`	Eşittir	`print(sifre == "1234")` `print(notum == 100)`	`True` `False`	İki değer birbirine eşitse `True` döner.
`!=`	Eşit Değildir	`print(sifre != "0000")` `print(notum != 70)`	`True` `False`	İki değer birbirinden farklıysa `True` döner.
`>`	Büyüktür	`print(notum > gecme_notu)` `print(50 > 70)`	`True` `False`	Soldaki değer sağdakinden büyükse `True` döner.
`<`	Küçüktür	`print(notum < 50)` `print(45 < gecme_notu)`	`False` `True`	Soldaki değer sağdakinden küçükse `True` döner.

Operatör	Anlamı	Kod Örneği	Çıktı	Açıklama
`>=`	Büyük veya Eşittir	`print(notum >= 70)` `print(notum >= 85)`	`True` `False`	Soldaki değer büyük veya eşitse `True` döner.

Operatör	Anlamı	Kod Örneği	Çıktı	Açıklama
`<=`	Küçük veya Eşittir	`print(gecme_notu <= 50)` `print(notum <= 60)`	`True` `False`	Soldaki değer küçük veya eşitse `True` döner.

> **Önemli Fark:** Tek eşittir (**`=`**) **atama operatörüdür** (değer aktarır). Çift eşittir (**`==`**) ise **karşılaştırma operatörüdür** (eşitliği sorgular).

3. Mantıksal Operatörler

Birden fazla koşulu birbirine bağlamak veya sonucun tersini almak için kullanılır. *(Örneklerde `yas = 15`, `ortalama = 82`, `devamsizlik = 3` ve `yagmurlu = True` kabul edilmiştir.)*

Operatör	Çalışma Mantığı	Kod Örneği (Senaryo)	Çıktı

`and` (VE)	Bağlanan **tüm** koşullar `True` ise `True`; en az biri `False` ise `False` döner.	**Teşekkür Belgesi Şartı:**
		`print(ortalama >= 70 and devamsizlik < 5)`
		Ehliyet Alma Şartı:
		`print(yas >= 18 and ortalama >= 50)`
			`True`
			
			`False`

`or` (VEYA)	Koşullardan **en az biri** `True` ise `True`; hepsi `False` ise `False` döner.	**Başarı veya Yaş Şartı:**
		`print(yas < 10 or ortalama > 80)`
		Kantin İndirimi (10 yaş altı veya 65 üstü):
		`print(yas < 10 or yas > 65)`
			`True`
			
			`False`

`not` (DEĞİL)	Mevcut mantıksal sonucun **tersini** alır (`True` → `False`, `False` → `True`).	**Dışarı Çıkma Durumu:**
		`print(not yagmurlu)`
		Yaş Kontrolünün Tersi:
		`print(not (yas >= 18))`
			`False`
			
			`True`

4. Defter Uygulaması: Örnek Kod ve Ekran Çıktısı

```
kullanici_adi = "ogrenci9a"
sifre = "python123"
sinav_notu = 85
devamsizlik = 4

# 1. Giriş Kontrolü (Hem kullanıcı adı hem şifre doğru olmalı -> and)
print(kullanici_adi == "ogrenci9a" and sifre == "python123")  # Çıktı: True

# 2. Takdir Belgesi Kontrolü (Not en az 85 VE devamsızlık 5'ten az olmalı ->
and)
print(sinav_notu >= 85 and devamsizlik < 5)                   # Çıktı: True

# 3. Sorumluluk Sınavı Kontrolü (Not 50'den küçük VEYA devamsızlık 10'dan fazla
-> or)
print(sinav_notu < 50 or devamsizlik > 10)                    # Çıktı: False

# 4. Tersini Alma (not)
print(not (sinav_notu < 50))                                  # Çıktı: True
```