Programlama Dilleri Laboratuvar Föyü
Python Programlama Dili 1
Python Tanıtımı ve Dil Yapıları

Şevket Umut ÇAKIR

1 Giriş

Bu deneyde Python programlama dilinin özellikleri ve basit dil yapılarının öğretilmesi amaçlanmaktadır.

Python öğrenmesi kolay, güçlü bir programlama dilidir. Verimli, yüksek-seviyeli veri yapılarına ve basit ama güçlü bir nesneye yönelik programlama yaklaşımına sahiptir.

Python yorumlayıcısı ve geniş standart kütüphanesi `https://www.python.org` adresinde ücretsiz biçimde kaynak kod ve çalıştırılabilir uygulama olarak mevcuttur[2].

Python programların kısa ve öz biçimde yazılmasına olanak tanır. Python ile yazılan programlar aşağıdaki sebeplerden dolayı C, C++ ve Java gibi dillerde yazılan programlara göre daha kısa olacaktır:[3]

- Yüksek seviyeli veri tipleri karmaşık işlemleri tek satırda ifade etmeye olanak verir.
- Blokların gruplanması süslü parantezler yerine girintiler ile gerçekleştirilir.
- Değişken veya argüman bildirimi(declaration) yapmaya gerek yoktur.

Python programlama dili "Monty Python's Flying Circus" isimli bir televizyon programından sonra bu ismi almıştır ve bunun yılanlarla bir ilgisi yoktur.

1.1 Sürüm Seçimi

Python programlama dilinin *Python 2.x* ve *Python 3.x* olmak üzere bilinen iki büyük sürümü mevcuttur ve eskiden beri hangi sürümün seçilmesi gerektiği ile ilgili tartışmalar bulunmaktadır. An itibariyle aktif olarak geliştirilen sürümü Python 3.7'dir. Bazı durumlarda Python 2'nin kullanılması avantajlı olabilecekken yeni başlayan birinin Python 3 öğrenmesi daha iyi olacaktır. Python 2 ve 3 arasındaki belli başlı farklar Tablo 1'de görülebilir[4].

Tablo 1: Python 2-3 Karşılaştırması

	Python 2	**Python 3**
Kullanım	Eskiden kalan: Hala belirli firmalarda kullanılmaktadır	Gelecek: 2020 yılında Python 2'nin yerini tamamen alacak
Kütüphane	Python 2 için geliştirilen bir çok eski kütüphane ileriye yönelik uyumlu değildir	Günümüzdeki geliştiricilerin bir çoğu kütüphaneleri özellikle Python 3 ile kullanılacak şekilde geliştirmektedir.
Kodlama	Metinler varsayılan olarak ASCII biçiminde kodlanır.	Metinler varsayılan olarak Unicode biçiminde saklanır.
Bölme	Bölme işlemi sonucu aşağı yuvarlanır.(5/2=2)	Bölme işlemi sonucu yuvarlama uygulanmaz.(5/2=2.5)
Yazdırma	print "hello"	print("hello")

1.2 Kurulum ve IDE Seçimi

Python programlama dili günümüzde kullanılan başlıca işletim sistemlerinin hepsinde kullanılabilmektedir. Python yorumlayıcısının ham hali `https://www.python.org` adresinden indirilebilir. En temel hali ile basit bir metin düzenleyicisi Python programlarını yazmak için yeterlidir. Entegre geliştirme ortamları programcıya kod renklendirme, otomatik tamamlama, yazım hatalarını kolay görme, inşa etme, çalıştırma ve hata ayıklama araçları sunma ve kod sürümleme gibi araçları sağlamaktadır. Python diline özgü aşağıdaki geliştirme ortamlarından bir tanesinin kullanılması tavsiye edilmektedir.

- PyCharm : Kolay kurulumu ve üstün hata ayıklama özellikleri bulunmaktadır.
- Spyder : Entegre iPython ortamı ve kısmi kod çalıştırması bulunmaktadır.
- Thonny : Kolay kullanımı ve gömülü gelen Python dili ile zahmetsiz kurulumu bulunmaktadır.
- Anaconda : İçinde gelen Spyder ortamı ve kurulu gelen kütühaneleri ile makine öğrenmesi ve yapay zeka uygulamaları geliştirmeye hazırdır.

2 Konu Anlatımı ve Deney Hazırlığı

Bu bölümde Python etkileşimli kabuğu, değişkenler, temel dil işleçleri, karar ve kontrol yapıları anlatılmaktadır.

2.1 Python Etkileşimli Kabuk

Python etkileşimli kabuğu basit işlemlerin yapılabileceği gibi karmaşık yapıların da oluşturulabileceği bir sanal makinedir. Başarılı bir Python kurulumundan

sonra terminal penceresinden **python** komutu verilerek kabuk çalıştırılabilir. Kurulum türüne bağlı olarak kabuk çalıştırıldığında Şekil 1'dekine benzer bir görünüme sahip olacaktır. Kabuk içinde Python kodları etkileşimli olarak çalıştırılabilir ve oluşturulan nesnelerin çıktıları ekranda görülebilir.

```
Python 3.6.5 |Anaconda, Inc.| (default, Apr 26 2018, 08:42:37)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Şekil 1: Anaconda platformu ile gelen Python kabuğu

2.2 Açıklama Satırları

Python dilinde açıklamalar tek satırlı veya çok satırdan oluşabilir. Aşağıda kullanımları gösterilmiştir.

```
# Açıklama satırı
x = 5 #'den sonrası açıklama
"""Bu da
çok satırlı
bir açıklama
"""
```

2.3 Değişkenler

Python dilinde değişkenleri ilk defa tanımlamak için atama işlemi yapmak gereklidir. Python dinamik tip sistemine sahip bir dil olduğu için bildirim gerekmemektedir. Bunun anlamı değişkenlerin türleri yapılan atama işlemindeki ifadenin türü ile belirlenir. Örneğin `x=5` işlemi x değişkeninin türünün tam sayı yapar.

Python temel olarak mantıksal değerler(`True` ve `False`), tam, ondalıklı ve kompleks sayılardan oluşan sayılar, listeler, çok ögeliler(tuple), metinler ve kümeler veri türlerini içerir. Bir değişkenin türünü öğrenmek için `type` fonksiyonu kullanılabilir.

2.4 İşleçler

Python aritmetik işlem notasyonu olarak infix notasyonunu(işlem ortada) kullanır. Tablo 2'de Python dilinde kullanılan işleçler verilmiştir.

2.5 Karar ve Kontrol Yapıları

Karar ve kontrol yapıları kodun akışını değiştirmemizi sağlayan yapılardır ve genellikle bir blok oluşturmamızı sağlarlar. Python dilinde bloklar diğer dillerde olduğu gibi süslü parantezlerle değil girintilerle tanımlanırlar. Girinti miktarı

Tablo 2: Python dilinde kullanılan işleçler

Tür	**İşleçler**	**Örnek**
Aritmetik	+, -, *, /, %, **, //	x+y
Karşılaştırma	==, !=, <>, >, <, >=, <=	a>b
Atama	=, +=, -=, *=, /=, %=, **=, //=	x += 2
Bit tabanlı	&, |, ^, ~, <<, >>	a&b
Mantıksal	and, or, not	True and False
Üyelik	in, not in	x in liste
Kimlik	is, is not	a is None

isteğe göre ayarlanabileceği gibi PEP 8 Python Stil Klavuzu'na göre bloklar için önerilen girinti miktarı 4 boşluk karakteri kadardır[1].

2.5.1 if yapısı

Verilen koşulun doğruluğuna göre işlem veya işlemler gerçekleştirmemizi sağlayan karar yapısıdır. Else if yapısı için **elif** anahtar kelimesi kullanılır. if, elif ve else yapılarından sonra blok tanımlamak için **:** sembolü kullanılır. Aşağıdaki örnekte kullanıcıdan alınan sayının değerine göre işlem yapan **if** yapısı verilmiştir.

```
sayi = int(input('Bir sayı girin: '))
if sayi < 0:
    print('Negatif bir sayı girdiniz.')
elif sayi > 0:
    print('Pozitif bir sayı girdiniz.')
else:
    print('Sıfır girdiniz.')
    print('Tebrikler!') # else bloğundaki başka bir satır
```

`input` fonksiyonu kullanıcıdan bir değer alınmasını sağlar. Alınan değer string(metin) türündedir. `int` fonksiyonu metni sayıya dönüştürmek için kullanılır. `print` fonksiyonu ekrana yazdırmak için kullanılmaktadır.

2.5.2 for döngüsü

Python dilindeki for döngüsü C ve türevi dillerden biraz farklıdır. C'deki for döngüsünün kısımlarının aksine Python dilindeki for döngüsü ardışık değerler içeren veri yapılarının(liste, string vd.) elemanları üzerinde tekrarlı işlemler yapmak için kullanılır. for döngüsü genellikle aralık belirten `range` fonksiyonu ile birlikte kullanılır. Aşağıda bu kullanıma örnek asal sayı hesaplayan bir kod bulunmaktadır. `range` fonksiyonu üst sınırda belirtilen değeri dahil etmez. Örneğin `range(4)` ifadesi 0,1,2,3 değerlerini içerir.

```
sayi = int(input('Bir sayı girin: '))
for i in range(2, sayi):
    asal = True
    for k in range(2, i):
        if i % k == 0:
            asal = False
            break
    if asal:
        print(i)
```

`input` fonksiyonundan geriye metin(string) döner. Bu değerleri tam sayıya dönüştürmek için `int` fonksiyonu kullanılır.

for döngüsü metnin karakterlerini ya da listenin elemanlarını da aşağıdaki şekilde dolaşabilir. Aşağıdaki kod kullanıcının girmiş olduğu metnin tüm karakterlerini alt alta yazdırır.

```
metin = input('Bir metin girin: ')
for harf in metin:
    print(harf)
```

2.5.3 while döngüsü

`while` döngüsü ilgili koşul doğru olduğu sürece sürekli olarak çalışacaktır. `while` döngüsünde de blok oluşturmak için `:` sembolü ve girintiler kullanılır. Aşağıda kullanıcı -1 değeri girene kadar kullanıcıdan sayı alan ve bu sayların toplamını ekrana yazdıran kod bulunmaktadır.

```
sayi = int(input('Toplanacak sayıları girin(-1 çıkış): '))
toplam=0
while sayi != -1:
    toplam += sayi
    sayi = int(input('Toplanacak sayıları girin(-1 çıkış): '))
print('Toplam: '+str(toplam))
```

Python metinleri birleştirmek için kullanılan + işlecinin parametrelerini otomatik olarak string tipine dönüştürmez. Bu nedenle son satırda bu dönüşüm `str` fonksiyonu ile gerçekleştirilir.

3 Deneyin Uygulanması

Bu deneyde taş-kağıt-makas oyunu, mükemmel sayı ve Collatz sanısı uygulamaları bulunmaktadır. Bu uygulamaların Python dilinde `http://bilmoodle.pau.edu.tr/` bulunan aktiviteler ile çözülmesi amaçlanmaktadır.

3.1 Taş, Kağıt ve Makas

Bu uygulamada iki kişi ile oynanan klasik taş, kağıt, makas oyununu gerçekleştirmek istenmektedir. Kullanıcıdan içerikleri {"taş", "kağıt", "makas"} olan iki adet metin alınacaktır. İlk girilen metin birinci oyuncunun seçimini, ikinci girilen metin de diğer oyuncunun seçimini göstermektedir. Girilen metinlere göre ekrana "birinci oyuncu kazandı", "ikinci oyuncu kazandı" veya "berabere" mesajlarını yazdıran Python kodunu yazınız.

3.2 Mükemmel Sayı

Tam bölenlerinin toplamı kendisine eşit olan sayılara mükemmel sayı denir. 28 ve 96 mükemmel sayılara verilebilecek iki örnektir. Kullanıcıdan alına sayının mükemmel sayı olup olmadığını bulan, mükemmelse ekrana "mükemmel" aksi takdirde "mükemmel değil" yazan Python kodunu yazınız.

3.3 Collatz Sanısı

Henüz doğruluğu kanıtlanmamış olan Collatz sanısı yapılan iki işlem sonunda 1'e indirgenebileceğini söyler. Bu iki işlem aşağıdaki gibidir:

$$f(n) = \left\{ \begin{array}{ll} \frac{n}{2}, & \text{n çift ise} \\ 3n+1, & \text{n tek ise} \end{array} \right\}$$

$$a_i = \left\{ \begin{array}{ll} n, & \text{i=0} \\ f(a_{i-1}), & \text{i>0} \end{array} \right\}$$

Collatz sanısında bir seri sayının kendisi ile başlar ve 1 ile sonlanır. Örneğin 13 sayısı için Collatz serisi 13, 40, 20, 10, 5, 16, 8, 4, 2,1 sayılarından oluşan 10 sayılık bir seridir.

Kullanıcının girdiği herhangi bir sayının Collatz serisinde kaç adımda 1'e ulaştığını bulan Python programını yazınız.

Kaynaklar

[1] *PEP 8 – Style Guide for Python Code.* URL: `https://www.python.org/dev/peps/pep-0008/` (son erişim: 19.2.2019).

[2] *Python web sitesi.* URL: `https://www.python.org` (son erişim: 18.2.2019).

[3] *The Python Tutorial.* URL: `https://docs.python.org/3/tutorial/index.html` (son erişim: 18.2.2019).

[4] *What Should I Learn as a Beginner: Python 2 or Python 3?* URL: `https://learntocodewith.me/programming/python/python-2-vs-python-3/` (son erişim: 18.2.2019).