Nümerik Analiz Uygulama

Hazırlayan:
Arş. Gör. Oğuz KIRAT

Python ile ilgili hatırlatmalar

- Bu dersimizde geçen sene öğrendiğimiz Python programlama dilini kısaca hatırlayacağız.
- Nümerik analiz dersi içerisinde öğreneceğimiz bazı metodları algoritmik olarak ifade etmede Python programlama dilinden faydalanacağız.

Python Kurulumu (Windows)

- Python Windows için aşağıdaki sayfadan indirilebilmektedir:

https://www.python.org/downloads/

Derslerimizde Python 3.11 sürümünü kullanacağız.

https://www.python.org/downloads/release/python-3117/

Windows Installer (64bit) sürümünü indirip kurabilirsiniz.

Python Kurulumu (macOS)

Homebrew paket yöneticisini kurunuz.

https://brew.sh/

Ardından

brew install python3

Komutuyla Python kurulumunu gerçekleştirebilirsiniz.

Python Kurulumu (Linux)

- Python, bir çok Linux dağıtımının depolarında bulunmaktadır:

sudo apt install python3 python3-pip python3-dev python3-testresources

Yukarıdaki komutla Debian/Ubuntu tabanlı dağıtımlarda Python kurulumu yapılabilir.

Geliştirme Ortamı (IDE)

- Bu ders kapsamında yazacağımız kodlar için

Visual Studio Code
ya da

Jetbrains PyCharm

geliştirme ortamlarını kullanabilirsiniz.

Python

- Dinamik yazılan bir dildir. Veri tipleri kesin olarak tanımlanmaz.
- Yorumlanan bir dildir.
- Nesneyi dayalı, esnek bir dildir.
- Satır sonlarına ; konmaz.
- {} işaretleri yerine içeri kaydırma kullanılır. Kod bloklarının hizası önemlidir.

Python Ekrandan Alma/Yazdırma

- Python’da ekrana yazdırmak için print fonksiyonunu kullanırız.
- Ekrandan almak için input() fonksiyonunu kullanırız.

```
isim=input("Adınızı giriniz: ")
yas=int(input("Yaşınızı giriniz: "))
dyili=2024-yas
print(isim+", doğum yılın:"+ str(dyili))
```

Koşullu İfadeler

```
alinan=input("Notunuzu giriniz: ")
alinan=int(alinan)
gecme_notu=60
kosullu_gecme_notu=50
if alinan>=gecme_notu:
    print("Tebrikler, dersi gectiniz!")
elif alinan>=kosullu_gecme_notu:
    print("Dersi kosullu gectiniz!")
else:
    print("Maalesef, dersi gecemediniz!")
```

Döngüler

```
for x in range(5):
    print(x,end=" ")
```

Çıktı: 0 1 2 3 4

```
for x in range(5,10):
    print(x,end=" ")
```

Çıktı: 5 6 7 8 9

```
for x in range(8,15,2):
    print(x,end=" ")
```

Çıktı: 8 10 12 14

Döngüler

```
x = 0
while x < 4:
    print(x, end=" ")
    x += 1
```

Çıktı: 0 1 2 3

Döngüler

Döngüyü kırmak için **break**, döngüde bir sonraki adıma geçmek için **continue** ifadesi kullanılır.

```
print("Girilen sayının 2 katını bulma programı")
while True:
    girdi=input("Bir sayı girin ya da çıkmak için q yazın:")
    if girdi.isdigit():
        girdi=int(girdi)
        print("Girilen sayının 2 katı:",girdi*2)
    elif girdi=="q":
        break
    else:
        print("Geçersiz giriş")
        continue
```

Listeler

```
listem=[3,"a"]
listem.append("b")
listem.append("c")
listem.append(1)
listem.pop(-2)
listem.append("d")
listem.insert(0,"e")
listem.remove("a")
b_indisi=listem.index("b")
print(listem)
print(b_indisi)
print(listem.count("b"))
listem.reverse()
print(listem)
```

Çıktı:

```
['e', 3, 'b', 1, 'd']
2
1
['d', 1, 'b', 3, 'e']
```

Tuple (Demet)

- Değiştirilemez (immutable) listelerdir

```
t="oguz","kirat"
print(t[1])
a=(1,2)
print(len(a))
```

Çıktı:

```
kirat
2
```

Set (Küme)

- Aynı elemanın yalnızca bir kez yer aldığı özel listelerdir.
- Matematikteki kümeler ile özdeşleştirilebilir.
- indis ile elemanlarına erişilemez.

Dictionary (Sözlük)

- Anahtar-değer (key-value) ikilileri tutmak için kullanılır.

```
d={"oguz":123, "ali":456, "veli":789}
d["selim"]=125
print(d["ali"])
#.get() metodu listede yoksa None dondurur
print(d.get("fatma"))
print(d.items())
print(d.keys())
print(d.values())
```

Çıktı:

456
None
dict_items([('oguz', 123), ('ali', 456), ('veli', 789), ('selim', 125)])
dict_keys(['oguz', 'ali', 'veli', 'selim'])
dict_values([123, 456, 789, 125])

Fonksiyonlar

```
def carp(liste):
    sonuc=1
    for i in liste:
        sonuc=i*sonuc
    return sonuc

print(carp([1,2,3,4,5]))
```

Sınıflar

```
class Kare:
    def __init__(self, kenar):
        self.kenar = kenar
    def alan(self):
        return self.kenar ** 2
    def cevre(self):
        return self.kenar * 4

kare = Kare(5)
print(kare.alan())
print(kare.cevre())
```

Çıktı:

25
20

Exception Handling (İstisna/Hata Denetimi)

```
l=["a","b","c"]
try:
    print(int(l[1]))
except IndexError:
    print("Bu indiste eleman bulunmamaktadır.")
except:
    print("Başka bir hata var.")
    #pass
else:
    #isteğe bağlı
    print("bir hata yok")
```

Modül Kullanımı

```
import time
from sys import platform
import random as rd
print(time.strftime("%A"))
print(platform)
rastgele = rd.randint(1, 100)
print(rastgele)
```

Çıktı:

Thursday
win32
86

Python Standart Kütüphaneleri

Python çok zengin bir standart kütüphaneye sahiptir.
İçeriğindeki modüllerle ilgili dokümantasyona:
https://docs.python.org/
adresinden ulaşabilirsiniz.

Modül Yükleme

Komut satırına

pip install numpy

Yazarak numpy kütüphanesini indirebilirsiniz.

Python’da kullanabileceğiniz 3. parti kütüphaneler için

https://pypi.org adresini ziyaret edin.

math modülü

```
>>> import math as m
>>> m.sqrt(4)
2.0
>>> m.pi
3.141592653589793
>>> m.cos(m.radians(180))
-1.0
>>> m.fabs(-4)
4.0
>>> m.floor(1.22)
1
>>> m.ceil(1.22)
2
>>> m.pow(2,3)
8.0
>>> round(1.1234,2)
1.12
```

Uygulama

- İkinci dereceden bir bilinmeyenli bir denklemi çözmek için Python'u kullanalın.
- Denklemi sıfıra eşitleyip katsayılarını paremetre gönderdiğiniz ve denklemin çözümünü döndüren bir fonksiyon yazın.
- Karmaşıklar sayılarda işlem yapmak için math yerine cmath modülünü kullanabilirsiniz.

Örneğin:$x^2-3x+2=0$

$\Delta=b^2-4ac$ ise kökler $\frac{-b+\sqrt{\Delta}}{2a}$ $\frac{-b-\sqrt{\Delta}}{2a}$

```python
import cmath

def kok_bul(a, b, c):
    delta = (b**2) - (4*a*c)
    root1 = (-b + cmath.sqrt(delta)) / (2*a)
    root2 = (-b - cmath.sqrt(delta)) / (2*a)
    return root1, root2


a = 1
b = -3
c = 2

root1, root2 = kok_bul(a, b, c)
print("Kök 1:", root1)
print("Kök 2:", root2)
```

```python
import cmath

class Denklem:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c

    def kok_bul(self):
        discriminant = (self.b**2) - (4*self.a*self.c)
        kok1 = (-self.b + cmath.sqrt(discriminant)) / (2*self.a)
        kok2 = (-self.b - cmath.sqrt(discriminant)) / (2*self.a)
        return kok1, kok2

a,b,c = 1,-3,2
denklem = Denklem(a, b, c)
kok1, kok2 = denklem.kok_bul()
print("Kök 1:", kok1)
print("Kök 2:", kok2)
```