# MDM_AI-ML_Python_Lec2

December 12, 2024

Data Analytics with Python

Lecture 2 (MDM)

By Ajit Kumar (ICT Mumbai)

Dec. 12, 2024

[ ]:
```

```

## 0.1 Decision making using if-else

[13]:
```
a = -6.96
if(a>=0):
    print(f'{a} is non negative')
```

[16]:
```
a = -6.96
if(a>=0):
    print(f'{a} is non negative')
    print('You win')
else:
    print(f'{a} is negative')
    print('Sorry, you have lost')
```

```
-6.96 is negative
Sorry, you have lost
```

[20]:
```
x,y,z = 40,7,30
if(x>y):
    if(x>=z):
        Max = x
    else:
        Max = z
else:
    if(y>=z):
        Max = y
    else:
        Max = z
print(f'Maximum of {x}, {y},{z} is {Max}.')
```

```
Maximum of 40, 7,30 is 40.
```

[22]:
```
def max3(x,y,z):
    if(x>y):
        if(x>=z):
            Max = x
        else:
            Max = z
    else:
        if(y>=z):
            Max = y
        else:
            Max = z
    print(f'Maximum of {x}, {y},{z} is {Max}.')
```

[24]:
```
max3(1,3,7)**2
```

```
Maximum of 1, 3,7 is 7.
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[24], line 1
----> 1 max3(1,3,7)**2

TypeError: unsupported operand type(s) for ** or pow(): 'NoneType' and 'int'
```

[33]:
```
def max3(x,y,z):
    if(x>y):
        if(x>=z):
            Max = x
        else:
            Max = z
    else:
        if(y>=z):
            Max = y
        else:
            Max = z
    return Max
```

[34]:
```
max3(1,3,7)**4
```

[34]:
```
2401
```

[27]:
```
def maxmin(x,y,z):
    if(x>y):
        if(x>=z):
            Max = x
        else:
            Max = z
        if(y<=z):
            Min = y
        else:
            Min = z
    else:
        if(y>=z):
            Max = y
        else:
            Max = z
        if(x<=z):
            Min = x
        else:
            Min = z
    #print(f'Maximum of {x}, {y},{z} is {Max}.')
    return Min, Max
```

[35]:
```
(a, b) = maxmin(4,2,3)
print(a)
print(b)
```

```
2
4
```

[ ]:
```
from math import sqrt
from cmath import sqrt as csqrt
a,b,c = 1,-6,3
disc = b**2-4*a*c
if(disc>=0):
    print('Roots are real')
    x1 = (-b+sqrt(disc))/(2*a)
    x2 = (-b-sqrt(disc))/(2*a)
else:
    print('Roots are imaginary')
    x1 = (-b+csqrt(disc))/(2*a)
    x2 = (-b-csqrt(disc))/(2*a)
print(f'Roots are {x1},{x2}')
```

[ ]:
```

```

## 0.2 Range function

[37]:
```
list(range(10))
```

[37]:
```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

[39]:
```
list(range(3,101,7))
```

[39]:
```
[3, 10, 17, 24, 31, 38, 45, 52, 59, 66, 73, 80, 87, 94]
```

[40]:
```
list(range(100,1,-7))
```

[40]:
```
[100, 93, 86, 79, 72, 65, 58, 51, 44, 37, 30, 23, 16, 9, 2]
```

[ ]:
```

```

## 0.3 For loops

[42]:
```
a = 3
d = 4
n = 20
s = 0
for i in range(n+1):
    s = s+(a+i*d)
print(s)
```

```
903
```

[ ]:
```

```

[44]:
```
sum([a+i*d for i in range(0,n+1)])
```

[44]:
```
903
```

## 0.4 Counting Pythagorian triplets

[49]:
```
n = 100
count = 0
for a in range(1,n+1):
    for b in range(a+1,n+1):
        for c in range(b+1,n+1):
            if(a**2+b**2==c**2):
                count += 1
                #print(a,b,c)
print(f'The number of Pythagorian triplets between 1 and {n} is {count}')
```

```
The number of Pythagorian triplets between 1 and 100 is 52
```

## 0.5 Newton-Raphson method

Find a appproximate root of $f(x) = x^3 - 5x + 21 = 0$ using Newton-Raphson method starting with $x_0 = 2.5$. Perform 10 iterations of

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, n = 0, 1, 2, ...$$

[52]:
```
def f(x):
    return x**3-5*x+21
def df(x):
    return 3*x**2-5
x0 = -2.5
for i in range(20):
    x1 = x0-f(x0)/df(x0)
    print(x1)
    x0=x1
```

```
-3.8
-3.4118997912317326
-3.356470126693511
-3.355383985322108
-3.355383572557528
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
-3.355383572557468
```

## 0.6 While loops

[54]:
```
N = 5734097569
ndigits = 0
while(N!=0):
    N = N//10
    ndigits = ndigits+1
print(ndigits)
```

```
10
```

[57]:
```
N = 5734097569
ndigits = 0
S = 0
while(N!=0):
    R = N%10
    N = N//10
    S = S+R
    ndigits = ndigits+1
print(ndigits, S)
```

```
10 55
```

[62]:
```
s= 0
n = 1
while(s<=1000):
    s=s+n
    n = n+1
print(n-1)
```

```
45
```

[61]:
```
k = 46
k*(k+1)/2
```

[61]:
```
1081.0
```

[69]:
```
def f(x):
    return x**3-5*x+7
def df(x):
    return 3*x**2-5
x0 = -2.5
maxitr = 100
tol = 1e-15
err = 1
i = 1
while(err>tol and i<=maxitr):
    x1 = x0-f(x0)/df(x0)
    err = abs(x1-x0)
    x0=x1
    i = i+1
print(x1,i)
```

```
-2.747346540307211 7
```

[70]:
```
def Newton_Raphson(f,df,x0,maxitr=300,tol=1e-8):
    err = 1
    i = 1
    while(err>tol and i<=maxitr):
        x1 = x0-f(x0)/df(x0)
        err = abs(x1-x0)
        x0=x1
        i = i+1
    return x1
```

[72]:
```
Newton_Raphson(f,df,x0,maxitr=30,tol=1e-8)
```

[72]:
```
-2.747346540307211
```

## 0.7 Use of break and continue

[74]:
```
password = ''
count = 1
while password != 'Python':
    password = input("Enter your password: ")
    count += 1
    if count > 3:
        print("No more tries.")
        break
```

```
Enter your password: Python
```

[ ]:
```
```

[75]:
```
while True:
    name = input('Enter your username: ')
    if name != 'Alice':
        continue
    pwd = input('Enter your password: ')
    if pwd == 'Star_Gold!':
        break
    else:
        print('That password is incorrect')
```

```
Enter your username: AJit
Enter your username: XYZ
Enter your username: Alice
Enter your password: StarGold
That password is incorrect
Enter your username: Star_Gold
Enter your username: Alice
Enter your password: Star_Gold
That password is incorrect
Enter your username: Alice
Enter your password: Star_Gold!
```

[ ]:
```
```

## 0.8 Pass statement

[76]:
```
word = 'Marathon'
count = 0
while count < len(word):
    if count < 3:
        print(word[count])
    elif count == 3 or count==5:
        pass
    else:
        print(word[count])
    count += 1
```

```
M
a
r
t
o
n
```

### 0.9 List Comprehensions

[9]:
```
word = 'ict mumbai'
letters = [letter.upper() for letter in word]
letters
```

[9]:
```
['I', 'C', 'T', ' ', 'M', 'U', 'M', 'B', 'A', 'I']
```

[11]:
```
word = 'Ict Mumbai'
cap = [letter.upper() for letter in word if letter.isupper()]
cap
```

[11]:
```
['I', 'M']
```

### 0.10 Excercises

1. Explore ‘try’ and ‘except’ keywords with two examples each.
2. Illustrate with examples how to iterate over a dictionary data.
3. Write a python programme (UDF) to input three real numbers $a, b$ and $c$ and check if it forms a triangle. If so find the nature of the triangle and hence print its area using the Heron’s formula.
4. Write a Python programme (UDF) to find root of an equation $f(x) = 0$ using the bisection method and hence use your defined function to find a root of a cubic $ax^3 + bx^2 + cx + d = 0$ in an an appropriate domain.
5. Write a python programme to find the appximate value of $\pi$ using the Monte Carlo method.
6. Write a python programme to estimate the value of $\log(a)$ for any positive real number $a$ using the Maclaurin series expansion

$$\log(1+x) = \sum_{i=1}^{\infty}(-1)^{i-1}\frac{x^i}{i}, \text{ for } |x| < 1.$$

7. Write a Python programme to find roots of a depressed cubic $x^3 + px + q = 0$. by considering all possible cases. (Follow the wiki page: https://en.wikipedia.org/wiki/Cubic_equation)

[ ]:
```

```