Lecture 3

May 17, 2023

1 Lecture 3

1.1 Dictionary

[1]:
```
my_dict= {"Name": "Akash", "Age":20}
my_dict
```

[1]:
```
{'Name': 'Akash', 'Age': 20}
```

[2]:
```
my_dict["Age"] = 21
my_dict
```

[2]:
```
{'Name': 'Akash', 'Age': 21}
```

[3]:
```
my_dict.items()
```

[3]:
```
dict_items([('Name', 'Akash'), ('Age', 21)])
```

[4]:
```
my_dict.keys()
```

[4]:
```
dict_keys(['Name', 'Age'])
```

[5]:
```
my_dict.values()
```

[5]:
```
dict_values(['Akash', 21])
```

1.2 Conditional statements

1.2.1 If else statements

[6]:
```
#if

a=10
b=5

if a==10:
    print(b)
```

```
5
```

[7]:

```
#if...else

a=7
b=5

if a==10:
    print(b)
else:
    print(a)
```

```
7
```

[8]:

```
#elif

a=7
b=5

if a==10:
    print(b)
elif a==7:
    print(a+b)
else:
    print(a)
```

```
12
```

1.3 Iterative statements

1.3.1 for loop

[9]:

```
n=10
for i in range(n):          #range(start, stop, stepsize)
    print(i, end=" ")
```

```
0 1 2 3 4 5 6 7 8 9
```

[10]:

```
#Iterate through list
my_list = ["Akash", "Vishal", "Raju", "Ram"]
for i in my_list:
    print(i, end=" ")
```

```
Akash Vishal Raju Ram
```

1.3.2 while loop

[11]:

```
n=10
i=0
while i<n:
    print(i, end=" ")
    i+=1
```

```
0 1 2 3 4 5 6 7 8 9
```

[12]:

```
#while loop with continue

n=10
i=0
while i<n:
    if i==6:
        i+=1
        continue
    print(i, end=" ")
    i+=1
```

```
0 1 2 3 4 5 7 8 9
```

[13]:

```
#while loop with break

n=10
i=0
while i<n:
    if i==6:
        i+=1
        break
    print(i, end=" ")
    i+=1
```

```
0 1 2 3 4 5
```

[14]:

```
#while loop with pass

n=10
i=0
while i<n:
    if i==6:
        i+=1
        pass
    print(i, end=" ")
    i+=1
```

```
0 1 2 3 4 5 7 8 9
```

1.4 Functions

1.4.1 User defined functions

[15]:

```python
#Prime number check function

def isprime(n):
    for i in range(2,n//2):
        if(n%i==0):
            return False
    return True

n=7
print(isprime(7))
```

```
True
```

[16]:

```python
#Circle area function

import math
def circ_area(r):
    return math.pi*r*r

r = float(input("Enter the radius of your circle: "))
print("Area of your circle is ", circ_area(r))
```

```
Enter the radius of your circle: 3
Area of your circle is  28.274333882308138
```

[17]:

```python
#Rectange area function

def rect_area(l, b):
    return l*b

l = float(input("Enter the length of your rectangle: "))
b = float(input("Enter the breadth of your rectangle: "))
print("Area of your rectange is ", rect_area(l, b))
```

```
Enter the length of your rectangle: 3
Enter the breadth of your rectangle: 4
Area of your rectange is  12.0
```

[18]:

```python
#BMI calculation function

def bmi(w, h):
    return w/(h*h)

w = float(input("Enter your weight(in kg): "))
h = float(input("Enter your height(in m): "))
print("Your BMI is ", bmi(w, h))
```

```
Enter your weight(in kg): 60
Enter your height(in m): 1.75
Your BMI is  19.591836734693878
```

1.4.2 Builtin functions

[19]:

```
a= range(1, 11)
print(len(a))
```

```
10
```

1.4.3 Lambda functions

[20]:

```
#One argument

x = lambda a:a+10
print(x(10))
```

```
20
```

[21]:

```
#Two argument

x = lambda a,b:a+b
print(x(10, 20))

x = lambda a,b:a*b
print(x(10, 20))
```

```
30
200
```

[22]:

```
#Three argument

x = lambda a,b,c:a+b+c
print(x(10, 20, 30))
```

```
60
```

1.5 Numpy

[23]:

```
#loading packages
import numpy as np
```

[24]:

```
#creating an array

arr = np.array([1,2,3,4,5])
print(arr)
```

```
[1 2 3 4 5]
```

[25]:

```
#checking dimension of array
print(arr.ndim)
```

```
1
```

[26]:

```
#creating an 2D array

arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print(arr)
```

```
[[ 1  2  3  4  5]
 [ 6  7  8  9 10]]
```

[27]:

```
#checking dimension of array
print(arr.ndim)
```

```
2
```

[28]:

```
#creating an 3D array

arr = np.array([[[1,2,3,4,5], [6,7,8,9,10], [11,12,13,14,15]]])
print(arr)
```

```
[[[ 1  2  3  4  5]
  [ 6  7  8  9 10]
  [11 12 13 14 15]]]
```

[29]:

```
#checking dimension of array
print(arr.ndim)
```

```
3
```

1.5.1 Zero and one array

[30]:

```
#zero array
a= np.zeros(10)
print(a)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

[31]:

```
#one array
a= np.ones(10)
print(a)
```

```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

[32]:

```
#increasing dimensions

a= np.zeros([3, 3, 4])      #([kD, ..., 3D, 2D, 1D])
print(a)
```

```
[[[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]

 [[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]

 [[0. 0. 0. 0.]
  [0. 0. 0. 0.]
  [0. 0. 0. 0.]]]
```

[33]:
```python
type(arr)
```

[33]:
```
numpy.ndarray
```

1.5.2 Functions of numpy

[34]:
```python
arr = np.array([10,20,30,40,50])

print(arr.size)
```

```
5
```

[35]:
```python
arr[4].dtype
```

[35]:
```
dtype('int64')
```

[36]:
```python
#Find index position of maximum element in array by using argmax()

print(arr)
print(arr.argmax())
```

```
[10 20 30 40 50]
4
```

[37]:
```python
#Find index position of minimum element in array by using argmin()

print(arr)
print(arr.argmin())
```

```
[10 20 30 40 50]
0
```

[38]:
```python
# arange function

a = np.arange(15) #end
b = np.arange(0, 15) #start, end
c = np.arange(0, 15, 2) #start, end, step
```

```
print(a)
print(b)
print(c)
```

```
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14]
[ 0  2  4  6  8 10 12 14]
```

[39]:

```
# Reshape function

a = np.arange(12)
a = a.reshape(3,4)
print(a)    #Multiples of 12 ex: 1,12  3,4  4,3  etc.

# Create 3D array using reshape
a = a.reshape(3,2,2)   #Multiples of 12 ex: 1,12  3,4  4,3  etc.
print(a)
```

```
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
[[[ 0  1]
  [ 2  3]]

 [[ 4  5]
  [ 6  7]]

 [[ 8  9]
  [10 11]]]
```

[40]:

```
# shape function
print(a.shape)
```

```
(3, 2, 2)
```

[41]:

```
# max function
print(a.max())
```

```
11
```

[42]:

```
# min function
print(a.min())
```

```
0
```

1.5.3 Slicing

[43]:

```
arr = np.array([10,20,30,40,50])

print(arr[1:3])
```

```
[20 30]
```

[44]:

```
arr = [10,20,30,40,50]

print(arr[2:5])
```

```
[30, 40, 50]
```

[45]:

```
arr[::-1]
```

[45]:

```
[50, 40, 30, 20, 10]
```

1.5.4 Exercise

[46]:

```
arr = np.array([[1,2],[2,3],[3,4],[4,5]])
print(arr)
```

```
[[1 2]
 [2 3]
 [3 4]
 [4 5]]
```

[47]:

```
print(arr.size)
print(arr.ndim)
```

```
8
2
```