Data Structure

- structuring of data
- Way of organizing the data in a particular format
- 4 Data structures

1.Tuple 2.List 3.Set 4.Dictionary

Tuple

- One of the data structures in python allows the user/programmer to store heterogeneous data items

```
    - It can store different type of data at a time
    - tuple() is the pre-defined function
    - It is immutable means cannot be modified further after
initialization
```

- 2 Methods
- Count
- Index

In [2]:
```python
tp=(1,2,"word",80.90,"python","a+b")
print(tp)
```

```
(1, 2, 'word', 80.9, 'python', 'a+b')
```

In [3]:
```python
for item in tp:
    print(item)
```

```
1
2
word
80.9
python
a+b
```

In [6]:
```python
#using index
tp[-2]
```

Out[6]:
```
'python'
```

In [11]:
```python
#slicing means extracting some part of iterabl
tp[:]
```

Out[11]:
```
(1, 2, 'word', 80.9, 'python', 'a+b')
```

In [12]:
```
tp[::2]
```
Out[12]:
```
(1, 'word', 'python')
```

In [13]:
```
tp[2:5]# upper bound is exclusive
```
Out[13]:
```
('word', 80.9, 'python')
```

In [14]:
```
tp[:4]# starts from first by default
```
Out[14]:
```
(1, 2, 'word', 80.9)
```

In [15]:
```
bin(9)# binary values
```
Out[15]:
```
'0b1001'
```

In [ ]:
```
count()# frequency of an item
# no.of occurence of data item

```

In [22]:
```
tp2=tuple(input().split())
tp2
```
```
Python Workshop 2345 @$$#@ 4343 4343
```
Out[22]:
```
('Python', 'Workshop', '2345', '@$$#@', '4343', '4343')
```

In [23]:
```
# print the integer values in tuple tp2
for item in tp2:
    if(item.isnumeric()):
        print(item,end=' ')
```
```
2345 4343 4343
```

In [29]:
```
t=(3,4,5,"word",90,34,"Workshop","SRKIT",9.3)
for item in t:
    if type(item)==float:
        print(item)

```
```
9.3
```

In [30]:
```
count=0
for val in t:
    if val==3:
        count+=1
print(count)
```
```
1
```

In [31]:
```
t.count(3)
```
Out[31]:
```
1
```

In [32]:
```
t.count(4)
```
Out[32]:
```
1
```

In [34]:
```
t.count("Workshop")
```
Out[34]:
```
1
```

In [35]:
```
t.index("Workshop")
```
Out[35]:
```
6
```

In [37]:
```
t.index(9.3)
```
Out[37]:
```
8
```

In [ ]:
```
#immutable
```

List

- It is also heterogeneous data structure
- Mutable in nature
- list() is the pre-defined function that represents the list
- [] square brackets
- list methods
  1. append
  2. count
  3. copy
  4. clear
  5. extend
  6. sort
  7.
  8. pop
  9. remove
  10.

In [38]:
```
# List initialization
nums=input().split()
print(nums)
```
```
9 45 tarun siddhardha 39 22
['9', '45', 'tarun', 'siddhardha', '39', '22']
```

In [44]:
```
li=[2,3,"python","Workshop",90.23,bin(int(input())),None,2,3,3,4,3,2,2]
li
```

```
10
```

Out[44]:
```
[2, 3, 'python', 'Workshop', 90.23, '0b1010', None, 2, 3, 3, 4, 3, 2, 2]
```

In [45]:
```
li.index(2)
```

Out[45]:
```
0
```

In [46]:
```
li.remove(90.23)
```

In [47]:
```
li
```

Out[47]:
```
[2, 3, 'python', 'Workshop', '0b1010', None, 2, 3, 3, 4, 3, 2, 2]
```

In [48]:
```
li.remove(li[3])
```

In [49]:
```
li
```

Out[49]:
```
[2, 3, 'python', '0b1010', None, 2, 3, 3, 4, 3, 2, 2]
```

In [51]:
```
li.append([1,2,3])
```

In [52]:
```
li
```

Out[52]:
```
[2, 3, 'python', '0b1010', None, 2, 3, 3, 4, 3, 2, 2, [1, 2, 3]]
```

In [53]:
```
li.extend([1,2,3])
```

In [54]:
```
print(li)
```

```
[2, 3, 'python', '0b1010', None, 2, 3, 3, 4, 3, 2, 2, [1, 2, 3], 1, 2, 3]
```

In [55]:
```
li.insert(4,"new")
```

In [56]:
```
li
```

Out[56]:
```
[2,
 3,
 'python',
 '0b1010',
 'new',
 None,
 2,
 3,
 3,
 4,
 3,
 2,
 2,
 [1, 2, 3],
 1,
 2,
 3]
```

In [58]:
```
li.pop(5)
```

Out[58]:
```
2
```

In [ ]:
```
#add,delete
# list allows duplicate value
```

In [59]:
```
unq=[]
for item in li:
    if item not in unq:
        unq.append(item)
    print(unq)
```

```
[2]
[2, 3]
[2, 3, 'python']
[2, 3, 'python', 'new']
[2, 3, 'python', 'new', None]
[2, 3, 'python', 'new', None]
[2, 3, 'python', 'new', None]
[2, 3, 'python', 'new', None, 4]
[2, 3, 'python', 'new', None, 4]
[2, 3, 'python', 'new', None, 4]
[2, 3, 'python', 'new', None, 4]
[2, 3, 'python', 'new', None, 4, [1, 2, 3]]
[2, 3, 'python', 'new', None, 4, [1, 2, 3], 1]
[2, 3, 'python', 'new', None, 4, [1, 2, 3], 1]
[2, 3, 'python', 'new', None, 4, [1, 2, 3], 1]
```

```
#### SET
- A well defined collection of objects
    - It is also heterogeneous data structure

```

```
In [60]: dir(set)
```

...

```
In [62]: a={8,4,9,10,23,54,1,9,5,10,45,90,12,9,14}
         a

Out[62]: {1, 4, 5, 8, 9, 10, 12, 14, 23, 45, 54, 90}

In [63]: a.add(20)

In [64]: a

Out[64]: {1, 4, 5, 8, 9, 10, 12, 14, 20, 23, 45, 54, 90}

In [65]: b={4,5,7,10,9,12,15,20}
         b

Out[65]: {4, 5, 7, 9, 10, 12, 15, 20}

In [67]:  a.intersection_update(b)

In [68]: a

Out[68]: {4, 5, 9, 10, 12, 20}

In [69]: a.update(b)

In [70]: a

Out[70]: {4, 5, 7, 9, 10, 12, 15, 20}

In [71]: a.difference_update(b)

In [72]: a

Out[72]: set()

In [73]: a.discard(22)

In [74]: a

Out[74]: set()
```

In [75]:
```
new=[1,2,4,67,9,2,3,4,10]
print(new)
```

```
[1, 2, 4, 67, 9, 2, 3, 4, 10]
```

In [76]:
```
set(new)
```

Out[76]:
```
{1, 2, 3, 4, 9, 10, 67}
```

In [77]:
```
a
```

Out[77]:
```
set()
```

Dictionary

- It is a paired data structure
- Represented by {key:value}
- dict() is the pre-defined function
- Dynamic data structures/mutable
    - keys can be any data type
        1. Keys should be unique
        2. Key will act as index/reference
    - Values can be any other data structure
        1. values might be similar
- Key&value together called as item

In [79]:
```
dic={1:'hi','name':'student',
     'friends':('tarun,siddahrdha'),
     'subjects':'marks',90.45:'point'}
print(dic)
```

```
{1: 'hi', 'name': 'student', 'friends': 'tarun,siddahrdha', 'subjects': 'mark
s', 90.45: 'point'}
```

In [ ]:
```
#working with dictionary
#method
```

```
dic.values()
```

In [81]:
```
dic.items()
```

Out[81]:
```
dict_items([(1, 'hi'), ('name', 'student'), ('friends', 'tarun,siddahrdha'),
('subjects', 'marks'), (90.45, 'point')])
```

In [ ]:
```
# entire dict depends only on keys
```

In [87]:
```
st="SRKIT"
for ch in st:


```

```
  Input In [87]

      ^
IndentationError: expected an indented block
```

In [89]:
```
for each in dic:
    print(each)
```

```
1
name
friends
subjects
90.45
```

In [90]:
```
for key in dic:
    print(dic[key])
```

```
hi
student
tarun,siddahrdha
marks
point
```

In [91]:
```
for item in dic.items():
    print(item)
```

```
(1, 'hi')
('name', 'student')
('friends', 'tarun,siddahrdha')
('subjects', 'marks')
(90.45, 'point')
```

In [92]:
```
help(dic.fromkeys)
```

```
Help on built-in function fromkeys:

fromkeys(iterable, value=None, /) method of builtins.type instance
    Create a new dictionary with keys from iterable and values set to value.
```

In [ ]:
```
sqs={}
for num in range(int(input()),int(input())):
    sqs[num]=num**2
print(sqs)
```

```
1
```

In [ ]:
```
sqs={}
for num in range(int(input()),int(input())):
    sqs[num]=num**2
print(sqs)
```

```
2
```

In [1]:
```
name,location=input(),input()
print("myself {} and i am from {}".format(name,location))

```

```
janaki ram
vijayawada
myself janaki ram and i am from vijayawada
```

Modules in Python

```
set of statements written to perform task said to be function
group of functions called as module
group of modules called as a package
    eng--
```

In [7]:
```
import math
```

In [8]:
```
math.pi
```

Out[8]:
```
3.141592653589793
```

In [10]:
```
math.gcd(5,8)
```

Out[10]:
```
1
```

In [13]:
```
random.randint(2,40)
```

Out[13]:
```
19
```

In [12]:
```
import random
```

In [14]:
```
import package
```

In [15]:
```
from package import functions
```

In [17]:
```
dir(functions)
```

Out[17]:
```
['__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__spec__']
```

In [22]:
```
functions.is_prime(9)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [22], in <cell line: 1>()
----> 1 functions.is_prime(9)

AttributeError: module 'package.functions' has no attribute 'is_prime'
```

Data Science Modules

- numpy,pandas,seaborn,matplotlib,open cv,scikit learn etc..

Numpy

- one of the data science modules
- numpy stands for numerical python
- used for scientific purpose
- homogenous data structure
- cannot be modified
    - matrix--array

In [23]:
```
tp=(4,5,6,'hi','hello')
ar=n.array(tp)
print(ar)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [23], in <cell line: 2>()
      1 tp=(4,5,6,'hi','hello')
----> 2 ar=n.array(tp)
      3 print(ar)

NameError: name 'n' is not defined
```

In [25]:
```
ar=np.array([[1,2,4],[3,4,7]])
ptint(arl)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Input In [25], in <cell line: 1>()
----> 1 ar=np.array([[1,2,4],[3,4,7]])
      2 ptint(arl)

NameError: name 'np' is not defined
```