Assignment
Data Structure

Q1.Discuss string slicing and provide examples.
Ans. Slicing in python refers to extracting specific portions (substring) of a string using indices. Slicing string Python involves obtaining a sub-string from a given string by slicing it from start to end.
Using Index positions, slicing allows you to capture substrings or characters with the slice syntax.

Syntax -: String[start : end : step]
Step by default is 1

Example ➖
```
str = " My self vaishnavi gupta and from rewari "
str[1:10]
```

Q2.Explain the key features of lists in Python.
Ans. Features of List in python -:
- Lists can contain items of different types at the same time (including integer,floating pointing numbers, strings and boolean values) .
- Lists are mutable and dynamic ; list items can be added ,removedor changed after the list is defined.
- Lists are ordered; newly added items will be placed at the end of the list.
- Lists use zero-based indexing; every list item has an associated index and the first item's index is 0.
- Duplicate list items are allowed.
- List can be nested within other lists indefinitely.
  Example -:

```
list = [2.3,2.3,4,"sohan", True,"gunnu"]
print(list)
```

Q3.Describe how to access, modify n and delete elements in a list with examples.
Ans. List are used to store multiple items in a single variable. List are one of 4 built-in data types in Python used to store collections of data , the other 3 are Tuple ,Set and Dictionary, all with different quality and usage.
1.Access item:- list items are indexed and you can access them by referring to the index number.
Example:
```
l = ["red","blue","orange","yellow","green"]
l[0]
```

Negative Indexing:- It means start from the end -1 refers to the last item, -2 refers to the second last element etc.
Example :
```
l = ["red","blue","orange","yellow","green"]
l[-4]
```
Range of Indexes:- you can specify a range of indexes by specifying where to start and where to end the range.
Example :-
```
l = ["red","blue","orange","yellow","green"]
l[1:3]
```
Range of Negative Indexes:- Specify negative indexes if you want to start the search from the end of the list.
Example :
```
l = ["red","blue","orange","yellow","green"]
l[-5:-2]
```
2.Modify element :- To change the value of a specific item,refer to the index number.
Example:
```
 l = ["red","blue","orange","yellow","green"]
l[2]="pink"
print(l)
```
Modify a Range of item Values:- To change the value of items within a specific range, define a list with the new value, and refer to the range of index numbers where you want to insert the new value.
Example:
```
l = ["red","blue","orange","yellow","green"]
l[0:2]=["pink","white"]
print(l)
```
Insert item:- To insert a new list item, without replacing any of the existing values ,we can use the insert() method. The insert() method Insert an item at the specified index.
Example:
```
l = ["red","blue","orange","yellow","green"]
l.insert(3,"black")
print(l)
```
3.**Delete element in list :-** pop() method remove the specified index.
Example :-
```
  l=["red","pink","orange","white","black"]
  l.pop()
```
del keyword can also delete the list completely.
Example:
```
   l=["red","pink","orange","white","black"]
   del l
```
clear() method empties the list.
Example:

```
l=["red","pink","orange","white","black"]
l.clear()
print(l)
```

Q4.Compare and contrast tuples and list with examples.

Ans. Tuples:-

- Tuples are immutable , it means elements can not be change after once created.
- The implication of iteration is comparatively Faster
- A Tuple data type is appropriate for accessing the elements
- Tuple consumes less memory as compared to the list
- Tuple does not have many built-in methods
- Because tuples do not change they are far less error-prone.

Example:-

```
tuple=("ram",1.2,"sita",4,5,True,3+5j)
print(tuple)
tuple[0]="gita"
```

```
 TypeError                                 Traceback (most recent
call last)
<ipython-input-5-b1b30ba97475> in <cell line: 3>()
      1 tuple=("ram",1.2,"sita",4,5,True,3+5j)
      2 print(tuple)
----> 3 tuple[0]="gita"

TypeError: 'tuple' object does not support item assignment
```

List:-

- List are mutable,it means elements can be changed after once created.
- The implication of iteration is time -consuming.
- The list is better for performing operation,such as insertion and deletion
- List consume more memory.
- List have several built-in methods
- Unexpected changes and error are more likely to occur.

Example:

```
list=["orange","Apple","Pear","Fig","Grapes",23.4,2,False]
print(list)
list[3]="Mango"
print(list)
```

Q5.Describe the key features of sets and provide examples of their use.

Ans. A set in Python programming is an unordered collection data type that is iterable , mutable, and has no duplicate elements

Set are represented by {}

How to Create s Set in python:

To create a python set ,place all elements or item separated by a comma inside curly braces{}.

```
set
={"priyanka","ravi","raju","sohan","yash","kanishka",34,234.4,True}
print(set)
```

How to Modify a Set in Python :-

Sets, through mutable are unordered.Thus,there is no scope for indexing,Indexing or slicing can not change or access an item of a set as a python set does not support it.
We use the **add()** method to add a single element and **update()** method when multiple elements are to be added.
Example:

```
 set  = {"ram"}
set.add("mohan")
set.add("syam")
print(set)
set.update("sita")
print(set)
```

How to Get the length of a Set:

len() method is used to determine the number of items a set has.

```
 set  =
{"priyanka","ravi","raju","sohan","yash","kanishka",34,234.4,True}
len(set)
```

Various Set Method :
copy()- used to return a copy of the set.
clear()- used to remove all items to the set
union()- used to return a new set as a union of sets.
difference()- used to return a new set as the differences of two or more sets
intersection()- used to return a new set as intersections of two sets.
pop()-used to return and remove an arbitrary set item, KeyError is raised if the set is empty.
issubset()- if another set is contained-in this set ,return true.

Q6.Discuss the use cases of tuples and sets in python programming.
Ans. Tuple:- Tuples are used to store multiple items in a single variable.
A tuple is a collection which is ordered and unchangeable.
Tuples are written with round brackets.
Tuple items:
Tuple items are ordered .unchangeable and allow duplicate values.
Tuple items are indexed, the first item has index[0], the second item has index[1] etc.
Ordered :
When we say that tuples are ordered ,it means that the items have is defined order, and that order will not change
Unchangeable:
We can not change, add or remove items after the tuple has been created.

Allow Duplicate :
They can have items with the same value.
Example:

```
t=("rishi","guddan","pari","vedika","ram",34,"sohan","sohan")
print(t)
t[0]="ram"
```

```
TypeError                          Traceback (most recent call last)
<ipython-input-5-ad010477dc46> in <cell line: 3>()
      1 t=("rishi","guddan","pari","vedika","ram",34,"sohan","sohan")
      2 print(t)
----> 3 t[0]="ram"

TypeError: 'tuple' object does not support item assignment
```

Set:-
A common use of sets in Python is computing standard math operations such as union,intersection ,difference and symmetric difference.
Union- combines element from two sets excluding duplicates
Example:

```
A={"orange","apple","pear","fig"}
B={"Mango","apple","grapes","orange"}
A|B
```

Intersection- Only common element between sets.
Example:

```
A={"orange","apple","pear","fig"}
B={"Mango","apple","grapes","orange"}
A&B
```

Differences- return the element that is present in first set and not in second set
Example:

```
A={"orange","apple","pear","fig"}
B={"Mango","apple","grapes","orange"}
A-B
```

Symmetric differences- return element that are present in either of set but nt in both set
Example:

```
A={"orange","apple","pear","fig"}
B={"Mango","apple","grapes","orange"}
A^B
```

Q7.Describe how to add, modify, and delete items in a dictionary with examples.
Ans. Dictionary:- These are used to store data value in key:value pairs.

A dictionary is a collection which is ordered*, changeable and do not allow duplicate.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
print(A)
```

Dictionary-item: Dictionary item are ordered,changeable,and do not allow duplicate
Dictionary items are presented in key:value pairs , and can be referred to by using the key name.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
print(A["Math"])
```

Adding items : Adding an item to the dictionary is done by using a new index key and assigning a value to it.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A["Drawing"]=99
print(A)
```

Change Dictionary Items:
Change Value: you can change the value of a specific item by referring to its key name
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A["Math"]=75
print(A)
```

Update Dictionary:
The update() method will update the dictionary with the items from the given argument.
The argument must be a dictionary ,or an iterable object with key:value pairs.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A.update({"Science":57})
print(A)
```

Get Keys: - The keys() method will return a list of all the keys in the dictionary.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
print(A.keys())
```

Get Values:
The values() method will return a list of all the values in the dictionary.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
print(A.values())
```

Get items: items() method will return each item in a dictionary , as tuples in a list.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A.items()
```

Remove items:
pop() method to remove items from a dictionary.pop() takes a key as an input and deletes the corresponding item from the python dictionary. It returns the values associated with input key.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A.pop("Social Science")
print(A)
```

popItem() : removes and returns a random item key-value from the dictionary.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A.popitem()
print(A)
```

clear():- This method drops all the items from the dictionary.clear() is referred to as the flush method because it flushes everything from the dictionary.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
A.clear()
print(A)
```

del:- Another way to remove an element from a dictionary is to use the del keyword del deletes individual elements and eventually the entire dictionary object.
Example:

```
A={"Math":99,"English":99,"Hindi":98,"Science":98,"Social Science":97}
del A
```

Q8.Discuss the importance of dictionary keys being immutable and provide examples.
Ans. Dictionaries store key-value pairs: Keys are analogous to indexes of a list.When using list you access the element via the index .With dictionaries you access value via the Keys.The keys can be of any datatype.
 Why must dictionaries keys be immutable:
The hash table **implementation of dictionaries uses a hash value calculated from the ley value to find the key.**
If the key were a mutable object , its value could change,and thus its hash could also change.
 The Python dictionary keys() function can be used to access dictionary elements in the same way that we can access list elements,however ,without the usages of keys(), no other mechanism provides a way to access dictionary keys as a list by index.

Retrieving dictionary keys as a list can be useful in many scenarios. It allows you to perform operations on the keys , such as iterating over them, checking for the presence of a specific key or performing operation based on the key's values.By converting the keys into a list, you can easily manipulate and analyse the data within the dictionary.

Example:

```
A={"Math":[99,45,44]}
A["Math"][1]
```

Example:

```
a={["Math","Eng","hindi"]:95}
print(a)
```

```
TypeError                                 Traceback (most recent call last)
<ipython-input-61-f3be4be06d91> in <cell line: 1>()
----> 1 a={["Math","Eng","hindi"]:95}
      2 print(a)
      3
      4

TypeError: unhashable type: 'list'
```