Sequences are containers where certain items are stored.

Ex sequence of cars, many different cars present combine to form a sequence

Sequence operations- concatenation, repetition, membership testing, slicing, indexing

Combine multiple sequences together to get one single sequence. Repetition to repeat multiple times

**Slicing** takes three things if we write index 1 to 4

**difference between sequence and array**

**array is a type of sequence**

**we dont have arrays we have lists in python**

**IN ARRAY we can store only one data type**

**in other sequences like lists and other sequences we can store multiple data types**

**in python indexing starts from 0 just like many other languages**

**VARIOUS TYPES OF SEQUENCES IN PYTHON**

* Lists
* Strings
* Dictionaries
* Tuples
* Sets

LISTS

these are the most versatile datatypes available in python. Are sequences where items are separated by commas and enclosed within square brackets.the items in the list **need not be** of the **same data type**.

Fruits=[‘mango’, ‘apple’,’grapes’]

In arrays there are only same data types(i.e if integers then only integers)

Operations in list

---------#append

**Listname.append**(‘history’)

Print(subjects)

**Stores and adds data after the last index by default**

--------#insert

We need to specify where exactly

**Listname.insert**(1, ‘history’)

Print(subjects)

**Stores and adds at a particular index where we want**

-------#extend

Means the list is extended and **both** the lists are **displayed** **together** now

**Listname.**extend(games)

Print(subjects)

-----------#remove

**Removes** the element from the list

Listname.remove(‘chemistry’)

Print(subjects)

---------#reverse

**Reverses** the **order** of the list

**Listname**.reverse()

Print(subjects)

---------#concatenation

**Combines** two strings but we use the plus **+ operator.** Similar to extend

**print(subjects + games)**

------#repitition

**Repeats** the statement list **twice** or **n** **times**

Print(listname \*2)

[‘physics’, ‘chemistry’, ‘maths’, ‘physics’, ‘chemistry’, ‘maths’]

TUPLES

Are basically **lists itself** but **once** a tuple is **defined** then **no changes** can be made to it. If we want to make changes in tuple we have to define another tuple.

We cant add or remove elements in a tuple

Hence it is basically a ***sequence of immutable python objects***.(immutables cos we cant make changes to it)

MAIN DIFFERENCE IS in tuples we use **parenthesis** but in LISTS we use **square braces**

**Ex- fruits = (‘Mango’, ‘apple’, ’grapes’)**

1. QUESTION--- when we have lists y do we use tuples?

**Iterating**(looping) through tuple is **faster** since tuples are immutable hence slight **performance boost**

* If we have **data which wont change**,implementing it as tuple guarantees that it will remain **write protected**

We cannot append a tuple so we cant use

Football = (‘persie’, ‘rooney’, ‘ronaldo’)

Cricket= (‘sehwag’,’virat’,’de villiers’)

***Football.append(‘messi’) is not possible in tuple***

SETS

A sequence that contains only unique objects (no repeated objects, all are immutable)it is represented in flower/curly braces.**”A LIST OF ITEMS INSIDE CURLY BRACES”.** similar to mathematics sets

Integers={1,2,3,4,5,6} real={7,6,8,9}

Print(integers)

Even if 6 here is entered **twice** it displays it **only once**

--------integers.discard(4) z #4 is discarded

------**union** print(integers|real) ans:- {1,2,3,4,5,6,7,8,9} ‘|’ symbol is used for union

------**intersection** print(integers & real) ans:- {6} ‘&’ symbol is used for intersection

-----**difference** difference is  **set1 – (set1 intersection set2)** print(integers-real)

ans:- {1,2,3,4,5,6} – {6}

------**symmetric difference** (basically union of uncommon elements)

Print(integers ^ real) #we use **xor** operator ans:- {1,2,3,4,5,7,8,9}

LIST can be converted to a set and list to tuple

***LIST TO SET***

Integers = [1,2,3,4,5,6,4,5,6,7,1]

Q=set(integers)

Print(q) ans:- {1,2,3,4,5,6,7} no elements are repeated

***LIST TO TUPLE***

Integers = [1,2,3,4,5,6,4,5,6,7,1]

Q=tuple(integers)

Print(q) ans:- (1,2,3,4,5,6,4,5,6,7,1)

DICTIONARIES

Dictionary is sequence which contains **key-valued** pairs. used when we have huge amount of data.

Every key has a value attached (ENCLOSED IN CURLY BRACES)

Example

D={1:’value’, ‘key’:2}

Print(type(d))

Print(“d[1]=”, d[1])

Print(“d[‘key’] = “, d[‘key’])

#generates error

Print(“d[2] = “, d[2])

EX1:-

Student= {‘Name’: ’dave’, ‘age’:32 }

Print(student[‘name’]) ans:-dave

#add a key value pair

**Student[‘gender’]= ‘male’**

Print(student) ans:- {‘name’: ‘dave’, ‘age’:32, ‘gender’:’male’}

#to **delete** a particular key value pair

**Student.pop**(‘name’) ans:- {‘age’:32}

#clear the dictionary

**Student.clear**()

#to update a particular value in a key

**Student[‘name’] = ‘robin’**

Print(student) ans:- {‘Name’: robin, ‘age’:32 }