Mo Tu We Th Fr Sa Su

Date: 8/09/2021

Python

Syntax :-

we use print() to display the desired code.

```
print ("Hello World!")
```

Python Indentation :-

Indentation refers to the space at beginning of a code line. Indentation in python is very important.

Examples :-

①
```
if 5>3:
    print("Five is greater")
```

② اگر دو مختلف conditions آجائیں تو مختلف Space رکھی جا سکتی ہے۔

Mo Tu We Th Fr Sa Su

Date:

```
if 10>5:
    print("Ten is greater")
if 15>10:
    print("Fifteen is greater")
```

③ اگر ایک ہی Condition میں دو مختلف یا ایک جیسے ہی print() آجائیں تو دونوں میں Space ایک جیسی ہی رکھی جاتی ہے۔

```
if 10>8:
    print("Ten is greater")
    print("Eight is Less")
```

They produce Error if write like that:

```
if 10>8:
    print("Ten is greater")
        print("Eight is Less")
```

output :- Indent Error

Mo Tu We Th Fr Sa Su

Date:

''' This block of code is very important in this program.'''

→ Variables :-

Variables are containers for storing data values. Python has no command for declaring a variable. (No need any data type).

Variables are temporary storage.

```
x = 4
x = 'Moni'
```

Case - Sensitive :-

```
a = 4
A = 4
```

Both are different A will not overwrite a.

Variable name don't start with number.

```
2var-name = 4
```

Error

Mo Tu We Th Fr Sa Su

Date: _____

Many Values to Multiple Variables :-

Python allows to assign multiple values to multiple variables in single line.

```
x, y, z = 4, 5, 6
a, b, c, d = "Ali", "Moni", "Ch", "Kamal"
```

One Value to Multiple variables :-

You can also assign one value to multiple variables.

```
x = y = z = 4
a = b = c = d = 'Moni'
```

Global Variables :-

Variables that are created outside the function is called global variable.

Mo Tu We Th Fr Sa Su

Date:

```
x = " awesome"
def myfunc():
    print("Python is " + x)
myfunc()
```

Local variable :-

Variables that are created inside the function.

```
def fun():
    age = 15
    print("Age is" + age)
fun()
```

Important :-

To create a global variable inside the function, you can use the global keyword.

```
def fun():
    global x
    x = "fantastic"
fun()
```

Mo Tu We Th Fr Sa Su

Date:

print("Python is" + x)

Data Types :-

Str, int, float, complex, list, tuple, dict, set, range, frozenset, bool, bytes, bytearray, memoryview.

Python Numbers :-

① int x = 4 (4, -4, 456789...)

② Float x = 4.25 (4.2, -4.25, 4.258) (4e10)

③ complex x = 1j (3+5j, 5j, -5j, 3-5j)

Imaginary part

Conversion :-

int ⇌ float

Float ⇌ int

int → complex

Float → complex

Complex ✗→ int

Complex ✗→ float

Mo Tu We Th Fr Sa Su

Date:

Random Number :-

Python does not have a random() function to make a random number, but python has built-in module called random that can be used to make random number.

Exple :-

```
import random

Print(random.randrange(0,10))
```

⇒ Type Casting :-

If we want to specify a data type on to a variable. This can be done with Casting. Casting in python is therefore done using constructor functions.

int() , str()

float()

Mo Tu We Th Fr Sa Su

Date: 09-09-2021

Examples :-

```
x = int(1)    # Integer
x = float(1)  # output 1.0
x = str("1")  # x will be 1
```

⇒ Python Strings :-

Strings in python are surrounded by either single quotation marks, or double quotation marks.

"Moni" is same as 'Moni'.

Assigning Value to a Variable :-

```
a = 'Moni'
```

OR

```
a = "Moni"
```

Multi-line String :-

```
a = """This is very interesting to know about the python."""
```

OR

Mo Tu We Th Fr Sa Su

Date:

a = """ This is very interesting fact about python."""

Checking String:

If we want to check if a certain phrase or character is present in a string, we use the keyword "in".

Examples:

① a = "This is beautiful world!"
print("is" in a)

output True

② a = "This is beautiful world!"
if "is" in a:
print("Yes")

Mo Tu We Th Fr Sa Su

Date:

Slicing Strings :-

we can return a range of characters by using the slice syntax.

```
a = "Hello, World!"
print(a[2:5])
print(a[:5]) # print from start
print(a[1:]) # print till End
```

Negative Indexing :-

```
a = "Moni"
print(a[-4]) # output => M
```

Modifying Strings :-

Upper Case :- -ے کو بڑا کر دیتا Letters کی

```
a = "Hello"
print(a.upper())
# output => HELLO
```

Mo Tu We Th Fr Sa Su

Date:

Lower Case :-

تمام Letters کو چھوٹا کر دیتا ہے۔

```
a = "HELLO"
print(a.lower())
# output => hello
```

Remove white space :- (strip())

یہ Space کو ختم کرتا ہے۔

```
a = "    Hello "
print(a.strip())
# output => "Hello"
```

Replace Strings :- (replace())

اگر ہم ایک Character کو یا مکمل لفظ کو تبدیل کرنا چاہتے ہوں تو ہم replace() فنکشن استعمال کرتے ہیں۔

Examples :-

Mo Tu We Th Fr Sa Su

Date:

(1)

```
a = "Hello"
print(a.replace("H", "M"))
```

output => "Mello"

(2)

```
a = "Hello world!"
print(a.replace("Hello", "Hi"))
```

output => Hi world!

String Concatenation :- → Like this.

```
a = "Hello"
b = "World"
c = a+b
print(c)
```

OR

```
print(a+b)
```

=> out put => Hello world

Mo Tu We Th Fr Sa Su

Date:

String Format :-

ہم Variables کو تو Add کر سکتے ہیں۔ لیکن ہم String اور Number کو Add نہیں کر سکتے۔

Example:

```
age = 36
txt = "I'm Mudasir " + age
print (txt)
```

output Error.

اگر ہم String اور Number کو Combine/Add کرنا چاہتے ہوں تو format() فنکشن استعمال کریں گے۔

Example:-

```
age = 36
txt = "My name is moni and
      I am {}"
print (txt.format(age))
```

Mo Tu We Th Fr Sa Su

Date:

Output:

My name is Moni and I'm 36.

② quantity = 3
item no = 567
price = 50.5
myorder = "I want to pay {2} dollars for {0} pieces of item no {1}."

print(myorder.format(quantity, itemno, price))

نوٹ:-
یہاں ہم Index کا خیال رکھنا بہت ضروری ہے۔

③ a = "I'm moni, and {age}"
print(a.format(age=36))

④ a = "I want to pay {price:2f}"
print(a.format(price=49))

Sapphire PEOPLE | RELATIONSHIP | INTEGRITY | DIVERSITY | ENVIRONMENT

Mo | Tu | We | Th | Fr ✓ | Sa | Su

Date: 10-09-2021

⇒ Built-In-Functions or Methods :-

1 Capitalize()

انگلش میں، ہر sentence کے پہلے capitalization کے rule کو follow کرتا ہے۔ یعنی کہ پہلا character کو بڑا کر دے گا اور باقی تمام کو چھوٹا رہنے دے گا اور اگر بڑے ہوئے تو انہیں چھوٹا کر دے گا۔

Example:

(i)
```
a = hi! world
print(a.capitalize())
```

output
Hi! world!

(ii)
```
a = python is fun!
print(a.capitalize())
```

output
Python is fun!

Mo Tu We Th Fr Sa Su

Date:

② Casefold():

یہ بڑے فونٹ کی lowercase کی casefold() لیکن یہ محدود ہے اور زیادہ Characters کو تبدیل کر دیتا ہے۔

Example:

```
a = "Hello World!"
print(a.casefold())
# output => hello world!
```

③ Center():

یہ فنکشن center() String کو دی گئی parameter کے مطابق space دے کر print کرتا ہے۔

Example:

```
a = "Banana"
print(a.center(5))
output =>      banana
```

④ Count():

یہ کسی String میں کسی مخصوص Value یا character کی کتنی بار موجود ہے اسے count کر کے دیتا ہے۔

Example :-

```
a = """I love apples. apples are
my favorite fruit """

print( a. count("a"))
```

output => 4

(5) Ends with () :-

endswith() فنکشن بتاتا ہے کہ String مخصوص character پر ختم ہو رہی ہے یا نہیں

Example :

```
a = "This is beautiful world"
print( a. endswith ("world"))
```

output => True

(6) find () :-

find() فنکشن اور Index فنکشن Same کام کرتے ہیں لیکن find() فنکشن return -1 کرتا ہے اگر کوئی Value نہ ملے تو، جبکہ Index Error دیتا ہے۔

Mo Tu We Th Fr Sa Su Date:

دونوں یہ کام کرتے ہیں کہ ایک String میں ایک Character یا ایک مکمل لفظ کس index پر موجود ہے، اسے تلاش کرتے ہیں۔

Exp:

```
a = "Hello"
print(a.find("o"))

# output => 4
```

(7) Swapcase():

swapcase() بڑے Characters کو چھوٹا اور چھوٹے کو بڑا کر دیتا ہے۔

Exp:

```
a = "Hello worLD!"
print(a.swapcase())

# output  hELLO WORld!
```

Mo | Tu | We | Th | Fr | Sa | Su

Date:

⇒ Boolean Values :-

Boolean represents one of two values : True or False

If condition is true then it represents true and false when condition false.

```
a = 10
b = 8
print ( a>b)
print (b>a)
```

output True
False

Most Values are True :-

(i) Any value is evaluated to true if it has some sort of content.

(ii) Any string is true, except empty string.

Mo Tu We Th Fr Sa Su

Date:

(iii) Any number is true, except zero.

(iv) Any list, tuple, set, dict are true except empty ones.

اگر کسی کلاس کے اندر __len__ فنکشن ہم بنا آئیں تو وہ false دے گا۔

Example:

```
class car():
    def __len__(self):
        return 0
car1 = car()
print(bool(car1))
```

Isinstance():

یہ determine Data type میں فنکشن isinstance() help کرتا ہے۔

Exp:

```
x = 200
print(isinstance(x, int))
# output True
```

Mo	Tu	We	Th	Fr	Sa	Su

Date:

⇒ Operators :-

Operators are used to perform operations on variables and values.

Operators in Python :-

1. (i) Arithmetic operators (+, *, −, /, %, **, //)
2. (ii) Assignment " (Assign value, =, +=, −=, *=, /=, etc)
3. (iii) Comparison " (==, !=, >, <, >=, <=)
4. (iv) Logical " (And, OR, Not, x<5 and x>5, x>5 or x<5, not(x>5 and x<5))
5. (v) Identity " (is, is not, works at memory level)
6. (vi) Membership " (in, not in — اگر کوئی چیز ہو تو True دیتا Sequence میں)
7. (vii) Bitwise " ("&", "|" OR, "^" XOR, "~" NOT, << zero fill shift, >> signed right shift)

↓

They work at bit level.

a = 10
b = 8

a & b = ?

Mo Tu We Th Fr Sa Su

Date:

Most 8-bits

0 0 0 0 | 0 | 0
0 0 0 0 | 1 | 0 0 0
---
0 0 0 0 | 1 | 0 0 0

2	10
2	5 - 0
2	2 - 1
	1 - 0

Convert into Decimal

2	8
2	4 - 0
2	2 - 0
	1 - 0

~~0 x 1 + 0 + 0~~

$= 0\times2^7 + 0\times2^6 + 0\times2^5 + 0\times2^4 + 1\times2^3 + 0\times2^2 + 0\times2^1 + 0\times2^0$

$= 0+0+0+0+8+0+0+0$

$= 8$ Ans

AND

X	Y	X.Y
0	0	0
0	1	0
1	0	0
1	1	1

OR works as Same like AND.

OR

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT

X	$\bar{X}$
0	1
1	0

Mo | Tu | We | Th | Fr | Sa | Su

Date:

Python Lists :-

Lists are used to store multiple values or items in a single variable.

Lists are created using square brackets.

```
itemno = [5, 4, 8]
```

(1) ordered, Changeable, allow duplicate values.

(2) List is also created using list() constructor.

```
itemno = ((5, 4, 8))
```

(3) Empty list

```
intemno = [ ]
```

Mo Tu We Th Fr Sa Su

Date:

Accessing Item of List :-

Lists is are ordered and index so they can be accessed by Theis index nomber.

```
a = [7,9,10]
print(a[1])
# output -> 9
```

Also accessed by negative index

```
a = [7,9,10]
print(a[-2])
# output -> 9
```

Range The Indexes :-

```
a = [5,7,9,11,13]
print(a[2:4]) # Specific print
```

Mo Tu We Th Fr Sa Su

Date:

```
print(a[2:])  # print index 2 to End
print(a[:5])  # print start from specific index (5)
```

Changing Items :-

```
a = ["orange", "banana", "Kiwi", "mango", "cherry"]
a[1] = "watermelon"
print(a)
a[1:3] = "blackcurrant", "Grapes"
print(a)

# output
[orange, watermelon, Kiwi, mango, Cherry]
[orange, blackcurrant, Grapes, mango, cherry]
```

Insert Items :-

اگر کوئی ایک Add item کرنا ہو تو بغیر کسی item کو نکالے، اس وقت insert() فنکشن استعمال کرتے

Mo Tu We Th Fr Sa Su

Date:

ہیں۔

```
a = [5,6,7]
print(a.insert(4,8))
```

4 ↓ index no, 8 → No to be added

Add List items :-

Add item کرنے کے لیے ہم تین function استعمال کرتے ہیں

i. Append()
ii. insert()
iii. Extended()

Append() اور insert() same کام کرتے ہیں لیکن append() میں جو ہم Add item کرتے ہیں وہ last میں add ہوتا ہے اور insert() میں ہم اپنی مرضی کی جگہ Add item کر سکتے ہیں۔

Ex:

```
a = [5,7,8]
a.append(9)
print(a)
```

output [5,7,8,9]

Mo Tu We Th Fr Sa Su

Date:

```
a = [5,7,8]
a.insert(1,9)
print(a)
```

output = [5,9,7,8]

اور اگر ایک سے زیادہ Add items کرنے ہوں
تو ہم Extend استعمال کرتے ہیں۔ List میں

```
a = [5,7,9,10]
b = [11,12,13]
a.extend(b)
print(a)
# out [5,7,9,10,11,12,13]
```

Mo Tu We Th Fr Sa Su

Date:

Remove List Item:

چار طریقوں سے List کو ہم Item کو Remove کر سکتے ہیں۔

i) remove() فنکشن

ii) pop() فنکشن

iii) Del() فنکشن

iv) clear() فنکشن

Example:

i) میں ہم جس item کو نکالنا چاہتے ہیں remove() اس کا نام سے نکالا جاتا ہے۔

Exp.

```
a = ["orange", "mango", "banana"]
a.remove("banana")
print(a)
```

Argument دینا لازمی ہے۔

output ⇒ [orange, mango]

ii) اگر ہم item کو index کی مدد سے نکالنا چاہتے ہوں تو ہمیں pop() فنکشن استعمال کرنا ہوگا۔

Mo Tu We Th Fr Sa Su

Date:

```
a = [5,6,7,8]
a.pop(2)
print(a)
# output → [5,6,8]
```

اگر ہم pop() فنکشن میں argument کو pass نہیں کریں گے تو وہ item, Last کو ختم کر دے گا۔

```
a = [5,6,7,8]
a.pop()
print(a)
# output ⇒ [5,6,7]
```

فنکشن میں pop() اور remove() میں فرق یہ ہے کہ remove میں ہم item کو اس کے نام کے ساتھ remove کرتے ہیں اور pop() میں item کو اشٹقال کرتے ہوئے Index No کو remove کرتے ہیں۔

Mo Tu We Th Fr Sa Su

Date:

Del ( ) :-

Del() فنکشن مکمل list کو بھی ختم کر دیتا ہے اور ایک خاص item کو بھی remove کر سکتا ہے۔

```
a = [5,7,9]
del a[1]
print (a)
# output => [5,9]
```

اگر مکمل Delete کرنی ہو تو

```
a = [5,7,9]
del a

# output  Error
```

Clear ( ) :-

clear() فنکشن صرف list کو content ختم کر دیتا ہے لیکن Error نہیں دیتا۔

```
a = [5,7,9]
a.clear()
print(a)   # output []
```

Mo | Tu | We | Th | Fr | Sa | Su

Date:

List Comprehension:-

List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list.

Exp:-

```
a = [5,6,7,8]
b=[]
for i in a :
    b.append(a)
print (b)
# output [5,6,7,8]
```

With List Comprehesion

```
b = [ i for i in a]
print (b)
```

Same output

Mo Tu We Th Fr Sa Su

Date:

Sorts Lists :-

List is sorted by sort() alphanomericaly, ascending by default.

```
a = [b, a, c, d]
a.sort()
print(a)
# output  [a, b, c, d]
```

If we want to sort decendingly then we pass the argument "reverse = true"

```
a = [b, a, d, c]
a.sort(reverse = True)
print(a)
# output  [d, c, b, a]
```

Same as for numbers

Mo Tu We Th Fr Sa Su

Date:

نوٹ:

اگر کسی items کے نام capital آجائے
ہیں تو ان کو پہلے sort out کیا جاتا ہے باقیوں
کو بعد میں sort out کیا جاتا ہے۔

Reverse():

فنکشن reverse() order کو items کے list تبدیل
کر دیتا ہے۔

```
a = [5, 9, 10, 12]
a.reverse()
print(a)
# output [12, 10, 9, 5]
```

Mo Tu We Th Fr Sa Su

Date:

Copy lists :-

- کاپی کرنے کے لیے copy کو lists سے دو طریقے ہیں:

(i) فنکشن copy()

(ii) Constructor ← copy()

Example:

(i)

```
a = [1,2,3]
b = a.copy()
print(b)
# output [1,2,3]
```

(ii)

```
a = [1,2,3]
b = list(a)   => Constructor
print(b)
```

Mo Tu We Th Fr Sa Su

Date:

↗ یا Add

Join Lists :: ⟹ Same work as adding Lists

ہم Lists کو تین طریقوں سے Join کر سکتے ہیں۔

i) operator کو استعمال کرتے ہوئے Variables کو جوڑنا

ii) فنکشن append سے

iii) فنکشن extend سے

Examples ::

i

```
list1 = ['a','b','c']
list2 = [1,2,3]
li3 = list1 + list2
print (li3)

# output
[a,b,c,1,2,3]
```

(ii)

```
a = ['a','b','c']
b = [1, 2, 3]
a.append(b)
print(b)
```

output
[a, b, c, [1, 2, 3]]

(iii)

```
a = [1, 2, 3]
b = ['a', 'b', 'c']
a.extend(b)
print(a)
```

output
[1, 2, 3, a, b, c]

List Index () :

Index() فنکشن ہم کو List میں موجود items کا index no بتاتا ہے ۔

Mo Tu We Th Fr Sa Su

Date:

Example:-

(i)

```
a = [1, 2, 3, 4]
b = a.index(3)
print(b)
# output => 2
```

(ii)

```
a = ["a", "bc", "d", "E"]
x = a.index("bc")
print(x)
# output => 1
```

Mo Tu We Th Fr Sa Su

Date: 14-09-2021

Python Tuples:

Tuples are used to store multiple items in a single variable.

Tuple is written with round brackets "()".

A tuple is collection which is ordered and unchangeable, also allow duplicate values.

```
a = (1, 2, 3)
# 1,2,3 Tuple
```

Creating Tuple with One item:

اگر ایک item کے ساتھ Tuple بنانا ہو تو item کے بعد Comma لگانا ضروری ہوتا ہے۔

```
a = ("Moni",) # Tuple
a = ("Moni") # not a tuple it is String
```

Mo Tu We Th Fr Sa Su

Date:

Accessing Tuples:

The procedure of accessing the tuples is same like the list of items accessed (i By index number, (ii Use range (print from one index to other index)

Update Tuples:

(i

Tuples میں items کو تبدیل نہیں کر سکتے لیکن workarounds ایسے موجود ہیں، جن کی بدولت items تبدیل کیے جا سکتے ہیں۔

Tuple کو تبدیل کرنے کے لیے ہمیں Tuple کو items میں تبدیل کرنا ہوگا۔ پھر دوبارہ List کو tuple میں تبدیل کریں گے۔

Examples:

Mo Tu We Th Fr Sa Su

Date:

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)
print(x)
output # ('apple', 'kiwi', 'cherry')
```

(ii) tuple میں tuple کو Add کرنے کے لیے:

```
x = (1,2,3)
y = (5,6)      # { All are tuples }
z = (7,)
m = x+y+z
# output (1,2,3,5,6,7)
```

Mo Tu We Th Fr Sa Su

Date:

Remove items ::

Tuple سے items کو remove نہیں کر سکتے
کیونکہ Tuples کو تبدیل نہیں کیا جا سکتا۔ البتہ
items کو workarounds کی جس سے
remove کیا جا سکتا ہے۔

اسی طرح remove کریں گے جس طرح List سے
items کو remove کرتے ہیں۔

تین) چار طریقے ہیں وہی۔

i) remove فنکشن
ii) pop فنکشن
iii) Del فنکشن
iv) clear فنکشن

Examples:-

i)
```
x = (1, 2, 3)
x.remove(1)
print(x)
```
output # (2, 3)

Mo Tu We Th Fr Sa Su

Date:

(2) x = (1,2,3,4)

x.pop(2)

print(x)

value ← item ; index ← no

output → (1,2,4)

Unpacking Tuple :

To extract the values of a tuple back into many variables. This is called unpacking.

```
fruits = ("apple", "banana")
(a,b) = fruits
print(a)
print(b)
```

output => apple
banana

Mo Tu We Th Fr Sa Su

Date:

Using Asterisk (*) :-

اگر Variable کی تعداد Values کے برابر نہ ہو تو
ہم (*) استعمال کرتے ہیں۔

Example:

(i) int = (1, 2, 3, 4, 5, 6, 7)

(a, b, *c) = int

print (a) # 1
print (b) # 2
print (*c) # 3, 4, 5, 6, 7

(ii) int = (1, 2, 3, 4, 5, 6, 7)

(a, *b, c) = int

print (a) # 1
print (*b) # 2, 3, 4, 5, 6
print (c) # 7

Mo Tu We Th Fr Sa Su

Date:

Join Tuples :-

To join tuples use "+" operator.

```
tuple1 = (1,2,3)
tuple2 = ("a","b")
tuple3 = tuple1 + tuple2
print (tuple3)

# output => (1,2,3,a,b)
```

Multiply Tuples :-

```
int = (1,2,3)
x = int * 2
print (x)

# output=> (1,2,3,1,2,3)
```

Mo | Tu | We | Th ✓ | Fr | Sa | Su

Date: 16-09-2021

Python Sets :-

Sets are used to store multiple items in a single variable.

A set is a collection which is both unordered and unindexed.

Sets are written with curly brackets.

Example:

```
a = {1,2,3}      # Set
a = {}           # Not Empty set
a = set()        # Empty set
```

Sets are also unchangeable and also do not allow duplicate values.

```
a = {1,2,3,4,3,2,1}   # Not a set
```

* set items can't be changed.

Mo Tu We Th Fr Sa Su

Date:

Access items of Sets :-

ہم Access کو items نہیں کر سکتے کیوں کہ یہ unordered ہوتے ہیں۔
البتہ for loop کے ذریعے print کروا سکتے ہیں۔

```
a = {1, 2, 3}
for x in a:
    print(a)
# output => 1, 2, 3
```

Add Set Items :-

Set میں items کو تبدیل نہیں کیا جا سکتا۔
لیکن مزید items کو Add کیا جا سکتا ہے۔

دو طریقے ہیں۔

(i) Add() فنکشن (ایک item کو Add کرنے کے لیے)
(ii) update() فنکشن (ایک سے زیادہ کو Add کرنے کے لیے)

Mo Tu We Th Fr Sa Su

Date:

Examples :-

(1)

```
a = {1, 2, 3}

b = {'a', 'b', 'c'}

a.update(b)
print(a)
# output {1, a, 2, b, c, 3}
```

(2)

```
a = {1, 2, 3}

a.add(4)
print(a)
# output {1, 4, 3, 2}
```

Mo Tu We Th Fr Sa Su

Date:

Remove Set Items:

Sets میں سے items کو remove کرنے کے لیے ہم تین طریقے استعمال کرتے ہیں۔

i) فنکشن remove()

ii) فنکشن discard()

iii) فنکشن pop() (یہ کسی بھی Last item کو remove کرتا ہے)

Examples:

i)
```
a = {1,2,3}
a.remove(3)
print(a)
# output → (1,2)
```

ii)
```
a = {1,2,3}
a.discard(2)
print(a)
# output (1,3)
```

فرق:

اگر جس item کو remove کر رہے ہیں وہ موجود نہ ہو تو remove() فنکشن Error دیتا ہے اور discard میں Error نہیں دیتا۔

Mo Tu We Th Fr Sa Su

Date:

(iii) a = {'a', 'b', 'c', 'd'}

```
a.pop()
print(a)
```

output => (a, b, c)

remove کرتا ہے item Last کو set صرف pop()
ہے اور ہم کوئی argument نہیں دے سکتے۔

البتہ list میں popu فنکشن میں ہم
arguments استعمال کر سکتے ہیں۔

Join Sets:

Join کو sets کیا جا سکتا ہے دو طریقوں سے۔

(i) Union() فنکشن
(ii) update() فنکشن

Mo Tu We Th Fr Sa Su

Date:

Examples :-

(i)

```
a = {1,2,3}
b = {4,5}
a.union(b)
print(a)
# output → (1,2,3,4,5)
```

(ii)

```
a = {'a','b','c'}
b = {1,2}
a.update(b)
print(a)
# output → (a,b,c,1,2)
```

Mo Tu We Th Fr Sa Su

Date: 17-09-2021

⇒ Python Dictionaries :-

Dictionaries are used to store data values in key : value pairs.

A dictionary is a collection which is ordered, changeable and does not allow duplicate values.

Dictionaries are written with curly brackets, and have keys and values.

Example :-

```
a = { 1 : 'a', 2 : 'b', 3 : 'c' }
print (type(a))
# output (class dict)
```

2021

Mo Tu We Th Fr Sa Su

Date:

Empty Dictionary:

```
a = { }        # Empty dict
```

To create Empty set:

```
a = set()      # Empty set
```

Not a:

```
a = { }
or
a = set{}
```

} Not Empty sets

Accessing items:

Access کرنے کے items سے طریقوں دو ہم

ہیں

(i) key سے

(ii) فنکشن سے get()

Mo Tu We Th Fr Sa Su

Date:

Examples:-

(i)

```
a = { 1: 'a', 2: 'b', 3: 'c' }

print (a[1])

# output => a
```

(ii)

```
a = { "a": 1, "b": 2, "c": 3 }

print (a.get("a"))

# output => 1
```

Add / Change Items:-

Change کے طریقوں سے ہم items کو Dictionaries میں کیا جا سکتا ہے۔

(i)

```
a = { 1: 'a', 2: 'b' }

print (a[1] = 'c')

a[1] = "c"
```

Mo Tu We Th Fr Sa Su

Date:

print (a)

output => {1:'c', 2:'b', 3}

(ii) update () :-

a = {1: "a", 2: "b"}

a.update({3: "c"})

print (a)

output => {1:'a', 2:'b', 3:'c'}

Remove items :-

Dictionary سے items کو remove کرنے کے کئی طریقے ہیں۔ لیکن pop() فنکشن کا استعمال بہترین ہے۔

Examples :-

Mo Tu We Th Fr Sa Su

Date:

```
a = {1:'a', 2:'b', 3:'c'}

a.pop(2)

print (a)

# output => (1:'a', 3:'c')
```

Copy Dictionary :-

(i)

```
a = {1:'a', 2:'b', 3:'c'}

b = a.copy()

Print (b)

# output → (1:'a', 2:'b', 3:'c')
```

« طریقے »

(ii)

Mo Tu We Th Fr Sa Su

Date:

```
a = { 1: 'a' , 2 : 'b' }

b = dict (a)
print (b)

# output -> {1: 'a' , 2: 'b' }
```

Main Difference between List, Tuple, set and Dictionary.

- List is a collection which is ordered changeable and allow duplicate values.
- Tuple is ordered, unchangeable and also allow duplicate values.
- Set is unordered, unindexed, unchangeable and don't allow duplicate values.
- Dictionary is ordered, changeable and no duplicate values.

Mo Tu We Th Fr Sa Su

Date: 19-09-2021

⇒ Python If-Else :-

Python supports the usual logical conditions from mathematics.

i. a == b
ii. a != b
iii. a > b
iv. a < b
v. a >= b
vi. b <= a or a <= b

These conditions can be used in several ways, mostly in if-statements.

If-Else کا استعمال تب کیا جاتا ہے جب کافی سارے Conditions کو check کرنا ہو۔

Examples:-

Mo Tu We Th Fr Sa Su

Date: _____

(i)

```
a = 50
b = 100

if a > b:
    print("A is greater")
else
    print("B is greater")
```

(ii)

```
a = 10
b = 20
c = 05
if a > b:
    print("A is greater")
elif a > c:
    print("A is greater")
```

Mo Tu We Th Fr Sa Su

Date:

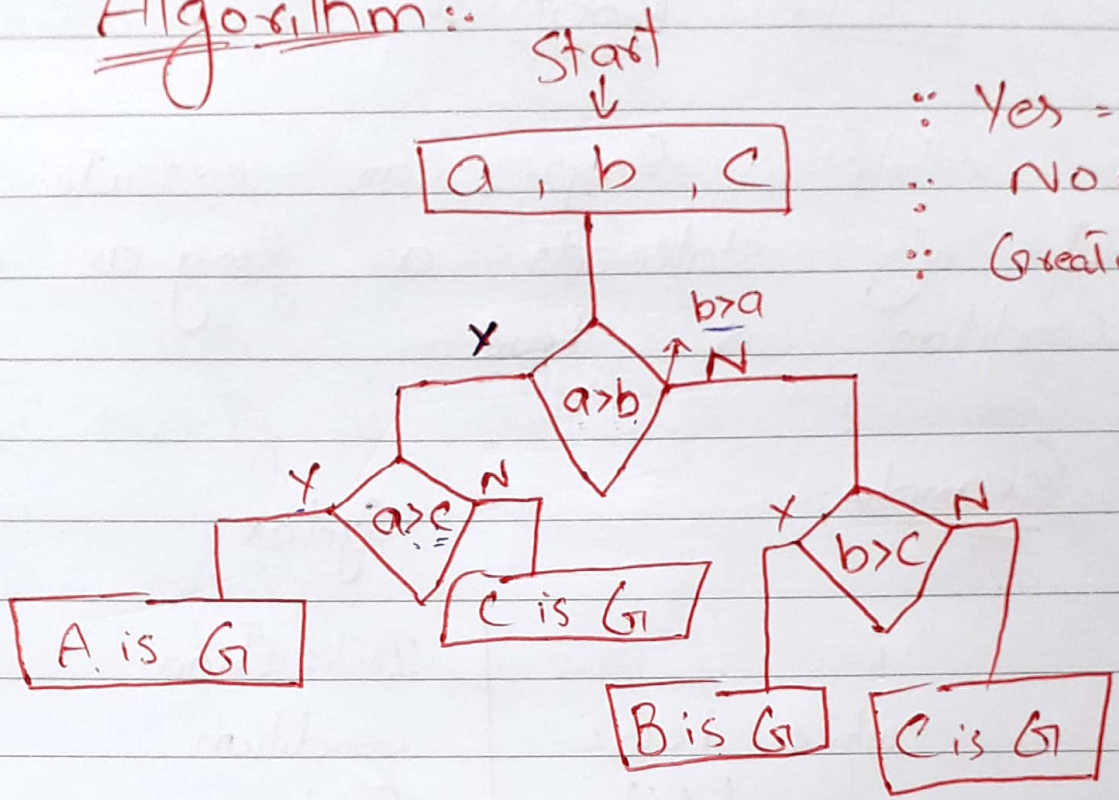
Nested If-Else:-

```
a = input("Enter the 1st No")
b = input("Enter The 2nd No")
c = input("Enter the 3rd No")

# To check which No is
# greater using If-Else

if a>b:
    if a>c:
        print("A is Greater")
    else:
        print("C is Greater")
elif b>a:
    if b>c:
        print("B is Greater")
    else:
        print("C is Greater")
```

Mo | Tu | We | Th | Fr | Sa | Su

Date: _____

Algorithm ::



:: Yes = Y

:: No = N

:: Greater = G

Mo Tu We Th Fr Sa Su

Date: 22-09-2021

⇒ While Loops:-

While loop can execute a set of statements as long as a condition is true.

Example:-

```
i = 0
while i<6:
    print(i)
    i = i+1
```

Syntax

Inilization
condition
Print
Increment / Decrement

—— // ——