Python

Page :
Date : / /

C

1. Procedural
   - functions
- .exe ✗

C++

- OOP
  - class + obj
  - close to hardware
- OS → EX
- .exe ✗

Java

- OOPs
  - syntax
  - secure lang
- .exe ✓

Python

- OOPs
  - Syntax
  - Web
  - AI
  - ML
  - Libraries
  - Ex → Django

~~fastest~~

Compile

No operator overloading

pointers

---

fastest

Compile

operator loading

pointer

Yes Multiple inheritance

---

less than C++ faster

Compile + interpreter

No Operator loading

No Pointer

No multiple inheritance

Dynamic

compiler + interpreter

platform independent

.exe ✓

slower execution

Compile + interpreter

Yes operator overloading

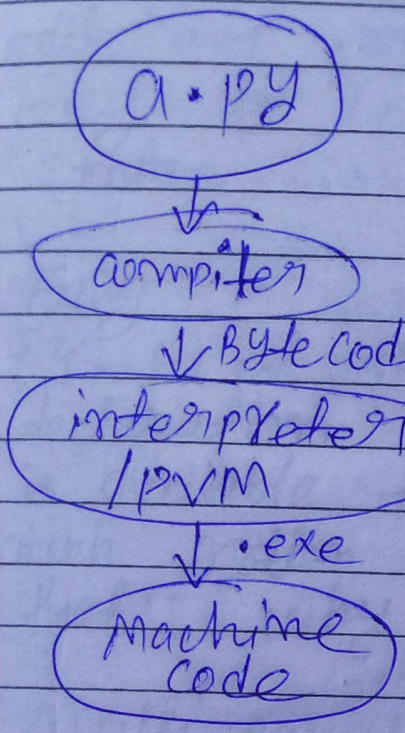
No pointer

Yes Multipl inheritance

Page :
Date / /

* How python achieves plateform independent ?

ans ⇒



→ Just because of byte code python is plateform independent.

Byte code
↓
PVM

but one condition is you have to install PVM (Python virtual Machine. then you will able to achieve independent plateform.

① what is plateform independent :
plateform independent means executive file of one operating system will be run on another operating system that is called plateform independent.

→ How it is plateform independent : python is plateform independent because it provides Byte code and pvm (python virtual machine) it is condition, using both thing python we will able to run python exe file from one system to another system

Note : pvm is also plateform dependent.

pyS

why Python is becoming so popular day by day.

Python features

Page :
Date : / /

* plateform independent: write once run anywhere.

(1) Free and open source: Python is freely available on python.org and it's code is openly available on the internet you can edit and takes it into your work.

(2) Easy to code: Easy to code because it's syntax is very easy to write

(3) Easy to read: There is no any concept of {}, (), <>, ;, and headfile, datatype declaration but in the python indentation is available that helps to identify how much portion of a block of code. that's why it is easy to read.

(4) Multi paradigm language: It supports both procedural programming and object oriented programming language that's why it is Multi paradigm language.

(5) Library support: Python have much and more librarys like Django → website, Tensorflow → AI, pandas → Data scientist.

(6) Portable language / plateform independent: Write ones run anywhere

(7) Interpreted language: Python is inter language that compile code line by line and easy to debugging it is a good

Integrated/

(8) Extensible: Python can by used with C++ Java and It canbe used with any other languages as well.
Ex⇒ Google, Netflix.

(9) Dynamic memory allocation: In the pyth memory allocation happen at run time that's why it is Dynamic memory allocation.

a=15; ~~int a; at runtime~~

* Limitations of Python.

(1) slow language: ~~Interpreter~~ Compile co one by one ~~language~~ that's why it is slow language.

(2) High memory consumption: Because of dynamic memory allocation a huge librar are used and all the memories allocated a the runtime that's why it consume more memo

(3) Mobile App development is not convinent: In the python App development is not easy, App development is easy in Java.

Data type in python

Page :
Date : / /

↳ Numeric → int = 56 ..... No limit (Comp memory)
↳ int
↳ float → only float No L/CM
↳ complex → 2+3i

↳ Sequence / ordered datatype → List, tuple, String

List ⇒ L = [] → string, int, float = [1, 'Ri', 1.6]
(indexes: 0, 1, 2 ; -3, -2)

print(L[0]) = o/p = 1.6

print(L[-3]) = o/p = 1

↳ mutable

Tuple ⇒ T = () ⇒ (1, 2.5, 'Rau')

T[1] = 3.5 X immutable

Cannot be change.

* String : No char in python

s = 'a'

s = 'Hi'

s = "I am Son's of"

s = """Hi I am "Rayhan" """ → Multi line string
(indexes: 0, 1, 2, 3)

index

Imp

anything

unordered datatyp : → set
→ dictionary

* Set : • No duplicate value • It is mutable
• No concept of index
S = {1, 1, 2} → print(s) = 1, 2 or 2, 1
S = {1, 1.5, 'Rai'}

Imp/ S = {}, print(type(s)) O/P = dictionary
S = set() print O/P = set

using loop you can access set.

for in S: print(i)

* Dictionary : No index in this, access using key only, key can be anything, any value can be anything, dictionary is mutable.
D = { key : value, Name : "Rai", age : 15 }

print(key)

print("Rai") = error we can not access key using value

Assignment -

Q1. How to create array in python How to access element in array

Q2. What is the use of Negative index in list

Q3. What is the boolean datatype in python

Operators

division int / float division

↳ Arithmetic : + - * % / //→ floor

print (5/2) — 2.5
" (5//2) — 2
" (-5/2) — -2.5
" (-5//2) — -3

~~क्यों की~~ floor division low floor leta है
so -2.5 नहीं -3 आएगा

* Comparison operator :

< ≤ == a=5
> ≥ != b=4
a>b F

* logical operator : And a=5
OR b=4
NOR T T
a && b = 4 5 and 4 = 4
a || b = 5 T T
a NOR b = false 5 ! 4 = false

* Bitwise operator : & | ~ ^ << >>

a=5
b=4

a & b → 4
a | b → 5
~ a → -6
a ^ b → 1
a << 2 → 4 → 0101 → 0100 = 4
a >> 2 → 1 → 0101 → 0001 = 1

Page:
Date: / /

5
0101
1010
1c → 0101
+1 → +1
0110 = -6

a = 1100 = -4
b = 0100 = 4
a & b = 4
a | b = -4
~a = 3 → 1100 → ! → 0011 → 3
a ^ b = -8
a << = -16
a >> = -1

1100 << 0100 =
00001100 << 00000100 =

1100
0011
1100
1
1101 = -8+4+1 = -3
|-3| = 3

In case of right sift if the no is negative instead of putting 00 you have to put (11).

Q = a = -3
b = -2

a << 3
a >> 3
~a

* Identity operator: Identity operators are used to check if two values are located on the same part of the memory

identity → is
→ is not

a = 5
b = 5 print(a is b) o/p = T

a = [10, 20, 30]
b = [10, 20, 30] print(a is b) ⇒ false

* print(id(a)) integer
print(id(b)) float
memory address same

List
tuple
dictionary
memory address same

In simple data type variable share same memory address like → int, float but in advance python like list tuple dictionary variable share different memory address.
a = [1,2,3]
b = [1,2,3]
print(a==b) = T

* Membership → in
in
not in

x = 20
L = [10,20,30]
print(x in L) = T

* Ternary operator: a if a>b else b

* Operator precedence:

			Associativity
1	**		right to left
2	~	bitwise NOT	L-R
3	*/11%		11
4	+,-		11
5	<< >>		11

Assignment = (R to L)

Page :
Date : / /

(6) &
(7) ^ bitwise and | L - R
(8) | " xor
(9) " or
(10) identity + membership + Assignment operator
(11) Not
(12) and
(13) or] boolean Not, and, or operator

good question

(2) Name = "abc"
Age = 5

if name == "abc" or
name = "def" and
age > 10
print ("1")
else print ("2")

O/P => 1

(1) 5 + (100 / 10) - 3 * 2

5 + 10 - 3 * 2
5 + 10 - 6
15 - 6 = 9.0

पहले and & then OR

if -> (if name = "abc" or name = "def") and age > 10) then o/p = 2

(3) x = 6
y = 2
print (6 | 2) O/P: 3
print (6 * * 2) O/P: 36

(4) a = 4 =
b = 11 =
print (~a) = -5
print (a | b) = 15

print (a >> 2) = 1
print (-18 || 4)
= -5

4 | b

Page :
Date : / /

(5) x = 10 y = 20

print(x and y) = 20

print (x or y) = 10

(6) x = 10 y = 50

if(x ** 2 > 100 and y == 50)

print("yes") // No output

(7) x = 1

y = 2

x = y += 4 = x = y = y + 4

print(x) = x = y = 6

x = 6

// x = 6 X

Syntax error

(8) Assignment → what are non associative operators.

* for in → for in ____ / L, T, D, S

while

L = [1, 2, 3, 4]

T = (1, 2, 3, 4)

S = {1, 2, 3, 4}

D = {1: "M", 2: "by"}

for i in T/L/S/D :

print(i)

but in dictionary

```
for in D:
    print( i, D[i] )
```

key value

Range :

* range (start, stop, step)

```
for in range( len(L) ):
    print( L[i] )
```

* while loop

```
while exp:
```

```
while (count<3)
    print("Hi")   #error
```

count = 0

```
while (count<9)
    print ("Hi")
    count ++;
```

पहली बार जो भी variable use होगा उसको define करना जरूरी है।

*

```
i=0
while( i<5 ):
    print(i)
    i=i+1;
    if i==3:
        break
else:
    print(0)
```

0
1

Page :
Date : / /

loop Control statement :

break, continue, pass

```
i=1
while True:
    if(i%7==0)
        break;
    print(i)
    i=i+1
```

O/P:
1, 2, 3, 4, 5, 6

```
i=0
while i<3
    print(i)
    i=i+1
else:
    print(0)
```

O/P:
0
1
2
0

```
x=['ab','cd']
for i in x
    i.upper()
    print(i)
```

['ab','cd']

```
a=i.upper()
print(x)
```

AB CD

```
x = 'abcdef'
while i in x:
    print(i)
```
i not defined

```
x = 'abcdef'
i = 'i'
while (i in x):
    print(i)
```
no output

```
x = 'abcd'
while i in x:
    print(i)
```
take while as condition infinite & print

```
i = 0
while (i<5):
    print(i)
    i = i+1
    if(i==3):
        continue
    else
        print(i)
```

0
1
2
3
4
5

Page :
Date : / /

Assign

* difference b/w continue and break

* Nesting loop :

```
i=1, j=i
for i in range (1,5):
    for j in range(i):
        print(j)
```

```
for i in range(1,5):
    for j in range(i):
        print(i, end=' ')
    print()
```

1
2 2
3 3 3
4 4 4 4

* condition statement
- if
- if else
- if elif else
- Nested if

(1)

```
if age > 10:
    ==
    ==
else:
    ==
    ==
```

```
if a>b:
    ==
    ==
elif:
    ==
    ==
else:
    ==
```

```
if a>b:
    if ==
       ==
       ==
```

Q. Accept three sides of a triangle and check whether it is equilateral, isosceles (2) or scalene (3) triangle.
(कोई MISS नहीं)

```
side1 = int(input("1st side"))
side2 = int(input("2nd side"))
side3 = int(input("3rd side"))

if (side1 == side2 == side3)
    print('equilateral triangle')
elif (side1 != side2 != side3)
    print('scalene triangle')
else
    print('isosceles triangle')
```

using if, elif, else

100 − 30 = (70)

Page :
Date : / /

Accept following from the user and calculate the % of the day attendance
1. total no of lecture
2. total no of days he/she absent
3. After calculating percentage show that if % is less than 75 then user will not able to sit in the exam.

```
lec = Input("Lecture No:")
days = input("total days absent:")
wow = (total lecture / 100) * 75
if (wow >= 75)
    print('you can sit')
else
    print('you cannot sit')
absent_percen = (days / lec) * 100
present = 100 - absent_percen
if (present >= 75)
    print('you can sit')
else
    print('you cannot sit')
```

Page:
Date: / /

Take an integer n and perfor following options:

① if n is odd → weard

② if n is ~~even~~ and in the inclusive range of 2 to 5 print Not weird.

③ if n is even and in the inclusive range of 6 to 20 print Weird.

④ if n is even and greater than 20 print not weird.

```
n = int(input("enter n"))
if(n%.2!=0)
    print('weard')

if(n%.2==0) if(n>20) print('not weird')
elif n in range(2,6)
    print('Not weird')

elif n in range(6,21)
    print('weird')
```

5

21

4

part-1

* String methods :

(i) lower() ; (ii) upper() (iii) ~~Capitalize~~ capitalize

(iv) title() → s = "second year"
हर word का first word upper होगा,
s = Second Year

limitation → "student's data"
→ Student'S Data
↑ limitation

(v) swapcase() → lower to upper, upper to lower
s = "Second YEAR"
op = ""

(vi) casefold() → similar to lower, but more agresive, ये भी lower करता है।
s = "ß"
s.lower() → ß
s.casefolder() → ss ये कोई भी को नहीं होगा,

(vii) center() → this method creates and return a new string that is paded with the specified character.

* Assign → Diff b/w isdecimal(), isdigit(), isnumeric() string.

S = "This is D2IT"

S1 = S.center(16) || (16, #) o/p

print(S1)

O/P = ---This is D2IT---

actual len = 12 → 16-12 = 4

(viii) count() → Count function returns the no of occurances of a substring within a string and it is case sensitive.

Ex ⇒

S = "This is D2IT"

S.count(i) || o 2

S.count(i, start, end);

(ix) endswith() : This method returns true if a string ends with the given suffix otherwise returns false.

S = "This is GNDEC."

print(S.endwith("DEC") → f

" (S.endwith("GNDEC") → f

("GNDEC.") → T

("GNDEC_") → f

("is GNDEC" T

("This is GNDEC.") → T

(10) find().

(11) ~~index~~ index().

Page :
Date : / /

4 marks → write a program to distinguished b/w find and index method of a string

find()	index()
(i) find() methods return the lowest index or the first occurance of the substring.	index() method return the index of first occurance of the substring.
(ii) ~~if find() method~~ if the substring in not found by find() method then it return -1.	if the substring is not found by index() method the it throws exception in this case.
(iii) (a) Syntax: s = "find me if you can" print(s.find("me")) → 5 print(s.index("me")) → 5	print(s.index("me")) → 5
(b) s.find("me", 4, 10)	
(iv) It can take three parameters.	This also can take three parameters.

9. WAP to find the no of whitespaces in the given string.

part-2

Page:
Date: / /

* String methods:

① isprintable() : This method returns true, if all the characters of the string are printable or the string is empty.

```
S = "This is DSIT" → f
S.isprintable() → F
S1 = ""
S1.isprintable() → T
```

② isspace():

```
S = "This_is_DSIT"
count = 0
for i in S:
    if(i.isspace() == True):
        count ++
print(count)
```

0/p = 2

③ istitle() → if the given string is title case (UpperCase) it returns true otherwise false.

④ join() → It is used to join elements of the sequence seprated by a string seprator

#, 11 & $

s = '_'.join("D2IT")

o/p → D_2_I_T

s = {1, 2, 3, 4, 5, 6, 7}

s1 = ('#')

print(s1.join(s))

o/p → #1, #2, #3 - - - -

inbuilt of dictionary

s1 = ('#')

s1 = {1: 'A', 2: 'B'}

print(s1.join(s))

1#2#

(5) ljust() & rjust() → sister of center()

lj → padding on right side

rj → padding on left side

Ex ⇒ s = "D2IT"

s.rjust(6, 'H') → ##D2IT

s.ljust(7, #) → D2IT###

(6) partition() → This method splits the string at the first occurance of the separator and returns a tuple containing the part before the separator,

the separater and the part after the separater.

S = "I love GNDEC Ludhiana"

S.partition("love")

O/P → ('I', 'love', 'GNDEC Ludhiana')

S = "I love GL and I love coffee"

S.partition("love")

O/P → ('I', 'love', 'GNDEC L and I love coffee")

if separater is not present

('Thisis', '', '').

(ii) rpartition() :- from right side

S = "I love you and I love cricket"

S.rpartition("love")

('I love you and', 'love', 'cricket')

⑦ replace() →

S = "Hello world Hello"

S.replace("Hello", "Bye")

O/P → Bye world Bye

S.replace("Hello", "Bye", 2) → 2 times

O/P → Bye Bye world Bye Bye.

⑧ startswith() → true / false

S = "GeekS for GeekS"
0 1 2 3 4

S.startswith("Geek") → True

&

S.startswith("for", 6, 9) → True

S.startswith("for", 6, 8) → False.
(n-1)

⑨ find() & index() →

File handling

* How to open a file:

```
with open("file.txt", "r") as f:
    print(f.read())
```

file.txt: Hello world

you can pass the value (2) → He

- read() display whole content of file.

* Read line →

```
with open("file.txt", "r") as f:
    print(f.readline())
```

readline() displays the first line of file only.

* Readlines →

```
with open("file.txt", "r") as f:
    print(f.readlines())
```

file.txt: Hello world

o/p → ["Hello", "world"]

readlines displays whole file in list.

* WAP that will display third line of file.

```
with open("file.txt", "r") as f:
    L = f.readlines()
    print(L[2])
```

* How to create a file:

```
with open("file.txt", "x") as f:
```

(A) capability of write fun अगर file exist नही करती है तो वह create कर देता है

```
with open("file.txt", "w") as f:
    f.write("Hellow\n world\n Bye")
```

write → It override the old content

In append mode

```
with open("file.txt", "a") as f:
    f.write("Hello\n world")
```

* Remove operation.

```
import os
os.remove("file1.txt")
```

* How to create CSV file in python.

* WAP that will count and display total no of words in a text file.

```
count = 0;
with open("file.txt", "r") as f: data = file.read()
for line in f:                    (each line)
    words = line.split(" ");
    count = count + len(words);
print("No of char:" + str(count))
```

WAP that will display the count of 'The' word in your text file.

① MDSL
② immutable → Insta FB

Pg: 1
Date: / /

(pyq) What are the immutable data types in python?
ans ⇒ Mutable data types can be modified after creation
Mutable data types
① List ② Dictionaries ③ Sets

Immutable data types
Immutable data types cannot be changed once they created
① strings ② Tuples ③ integers ④ floats ⑤ Booleans

(PYQ) difference b/w isdecimal(), isdigit(), ~~isdigit()~~ and ~~isn~~ isnumeric() string functions.

isdecimal()	isdigit()	isnumeric()
(i) It returns True if all characters in the string are decimal digit (0-9)	It returns True if all characters in the string are digits.	It returns True if all characters in the string are numeric.
(ii) It doesn't accept Superscript or subscript digits.	It accepts superscript and subscript digits.	It accepts superscript or subscript and of well as fractions.
(iii) Example: isdecimal("1234") O/P: True isdecimal("$1^2 3^4$") O/P: False	Example: isdigit("1234") and isdigit("$1^2 3^4$") O/P: True isdigit("1/2") O/P: false	Example: isnumeric("1234"), isnumeric("$1^2 3^4$"), and isnumeric("1/2") O/P: True

PY8 write a program to accept a number from a user and calculate the product of all the digits

n=56 o/p = 30

```
n = int(input("Enter a number:"))
sum = 1
while n > 0:
    digit = n % 10
    sum = sum * digit
    n = n // 10;
print("The product of the digits is:", sum)
```

PY9 write a program to accept a number from user and calculate the ~~product of all the digits of the given~~ the sum of all numbers from 1 to the given number.

ans ⇒

```
n = int(input("Enter a number:"))
total_sum = sum(range(1, n+1))
print("The sum is:", total_sum)
```

08 →

```
sum = 0
n = int(input("Enter a number:"))
for i in range(n+1):
    sum = sum + i
print("Sum is:", sum)
```

PYQ Write short Note on Operator precedence vs operator Associativity.

Operator precendence

It dictates the order in which operations are performed when multiple operators are present in an expression.

Exponential	**
Unary plus/Minus, bitwise Not	+, -, ~
	*, /, //, %
Addition, subtraction	+, -
bitwise shift left, right	<<, >>
	<, <=, >, >=
	==, !=
	and, or
Assignment operator	=, +=, -=, *=, /=, //=, %=
	**=, &=, ^=

Operator Associativity

It determines how to group operands when multiple operators have the same precendence.

Right to left
Right to left
Left to right
"
"
"
"
"
Not applicable
Not applicable

Python ascii

0 →	48	A → 65	a → 97
1 →	49	B → 66	b → 98
2 →	50	C → 67	c → 99
3 →	51	!	!
4 →	52	!	!
5 →	53	Y → 89	x → 121
6 →	54	Z → 90	z → 122
7 →	55		
8 →	56		
9 →	57		

```
O/P=? L=['a','b','c','d','D']
      L.sort(reverse = True)
      print(L)
O/P: ['d','c','b','a','D']
L.sort(reverse = False)
O/P: ['D','a','b','c','d']
```

Page :
Date : / /

(Q7) How to read and write into a Text files in python.

ans ⇒

To read from a text file in python

```
with open('filename.txt', 'r') as f:
    content = f.read()
    print(content)
```

To write to a ~~text~~ file

```
with open('filenam.txt', 'w') as f:
    f.write('Hello, I am raushem.')
```

(Q8) what is difference b/w count(), and length(), function in List ?

count()

(i) count() method is used with list to count the occurrences of a specified element, while len().

(ii)
```
list = [1,2,3,1,4,1,5]
target = 1
count = list.count(target)
print(count)
```
o/p → 3

length()

~~len~~ len() gives the total no of element in the list

(2)
```
list = [1,2,3,1,4,1,5]
target = 1
count = len(list)
print(count)
```
o/p → 7