Toppersnotes®
Unleash the topper in you

बिहार

कम्प्यूटर शिक्षक

बिहार लोक सेवा आयोग

कम्प्यूटर अध्ययन

भाग – 3

SIERRA INNOVATIONS

BIHAR COMPUTER TEACHER

कम्प्यूटर अध्ययन – 3

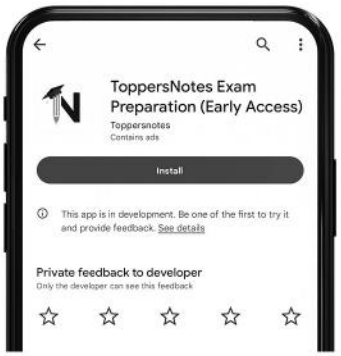
प्रिय विद्यार्थी, टॉपर्सनोट्स चुनने के लिए धन्यवाद।

नोट्स में दिए गए QR कोड्स को स्कैन करने लिए टॉपर्स नोट्स ऐप डाउनलोड करे ।

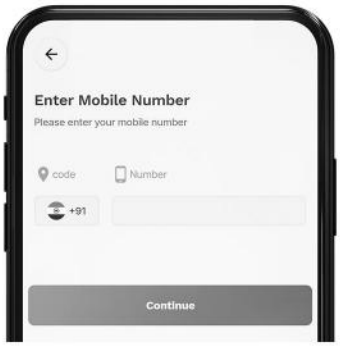
ऐप डाउनलोड करने के लिए दिशा निर्देश देखे :-

ऐप इनस्टॉल करने के लिए आप अपने मोबाइल फ़ोन के कैमरा से या गूगल लेंस से QR स्कैन करें ।

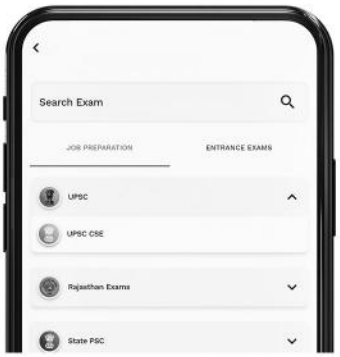
टॉपर्सनोट्स एग्जाम प्रिपरेशन ऐप



टॉपर्सनोट्स ऐप डाउनलोड करें गूगल प्ले स्टोर से ।

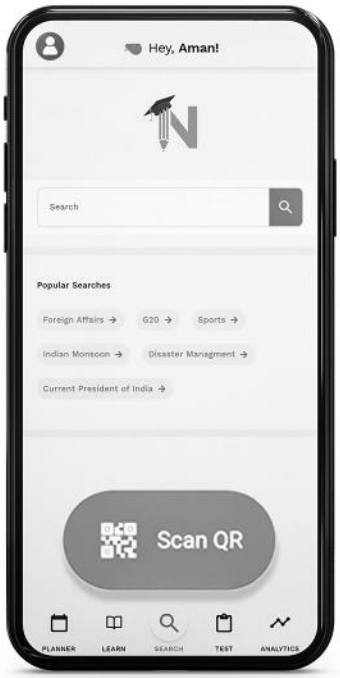


लॉग इन करने के लिए अपना **मोबाइल नंबर** दर्ज करें ।



अपनी **परीक्षा श्रेणी** चुनें ।

सर्च बटन पर क्लिक करें ।



SCAN QR पर क्लिक करें ।

किताब के **QR कोड को स्कैन** करें ।

किसी भी तकनीकी सहायता के लिए
hello@toppersnotes.com पर मेल करें
या **766 56 41 122** पर whatsapp करें ।

Python

- Python एक Interpreted, High Level और ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग लैंग्वेज (OOPS) है ।
- Python को Guido Van Rossum ने 1990 के दशक में विकसित किया था । Guido Van Rossum एक Dutch प्रोग्रामर थे । यह बहुत ही अच्छी प्रोग्रामिंग लैंग्वेज है क्योंकि इसके द्वारा बहुत तेजी से एप्लीकेशन को विकसित किया जा सकता है और एप्लीकेशन का तेजी से निर्माण किया जा सकता है क्योंकि यह Dynamic Typing तथा Dynamic Binding के Options देता है ।
- बहुत सारी बडी कंपनियाँ भी Python का प्रयोग करती है ।
  जैसे – Youtube, Quora, Instagram तथा Google आदि ।
- पायथन का प्रयोग ज्यादातर वीडियो गेम की प्रोग्रामिंग करने, Artificial Intelligence, Neural Networks, Natural Language Generation आदि में किया जाता है ।
- Python में अनेक प्रकार के एप्लीकेशन विकसित कर सकते है–
  1. वेब एप्लीकेशन
  2. गेमिंग एप्लीकेशन
  3. RP एप्लीकेशन
  4. ग्राफिकल एप्लीकेशन
- पायथन को **C,C++** तथा **Java** प्रोग्रामिंग लैंग्वेज में आसानी से **Integrate** कर सकते है ।

पायथन की विशेषताएँ (**Features of Python**)

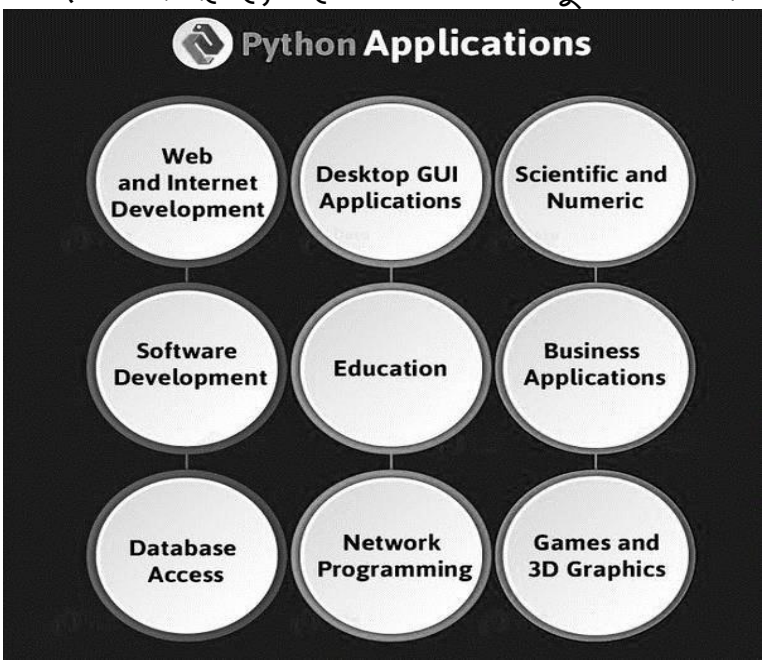
- पायथन सीखना और उपयोग करना आसान है । यह डेवलपर के अनुकूल और हाई लेवल लैंग्वेज है ।
- पायथन लैंग्वेज अधिक अभिव्यंजक (**Expressive**) है इसका मतलब यह है कि यह अधिक समझने योग्य और पठनीय है ।
- पायथन एक इंटरप्रेटेड लैंग्वेज है यानी इंटरप्रेटेड कोड को एक ही बार में लाइन बाय लाइन एक्जीक्यूट करता है । यह डिबगिंग को आसान बनाता है और इस प्रकार शुरुआत के लिए उपयुक्त है ।
- पायथन विभिन्न प्लेटफार्मों जैसे विंडोज, लिनक्स, यूनिक्स और मैकिंटोश आदि पर समान रूप से चल सकता है, इसलिए, हम कह सकते है कि पायथन एक पोर्टेबल लैंग्वेज है ।
- पायथन ऑफिसियल वेब एड्रेस पर फ्री उपलब्ध है । सोर्स–कोड भी उपलब्ध है । इसलिए यह ओपन सोर्स है ।
- पायथन ऑब्जेक्ट ओरिएंटेड लैंग्वेज का समर्थन करता है और **Classes** और **Objects** का प्रयोग करता है । तात्पर्य यह है कि अन्य लैंग्वेजों जैसे कि **C/C++** का उपयोग कोड को कम्पाइल करने के लिए किया जा सकता है और इस प्रकार इसका उपयोग पायथन कोड में किया जा सकता है ।
- पायथन में एक बडी लाइब्रेरी है और यह तेजी से एप्लीकेशन डेवलपमेंट के लिए मॉड्यूल और कार्यों के समृद्ध सेट को प्रस्तुत करता है ।
- पायथन का उपयोग करके ग्राफिकल यूजर इंटरफेस विकसित किया जा सकता है ।
- इसे **C,C++, JAVA** आदि लैंग्वेजों के साथ आसानी से एकीकृत किया जा सकता है ।

Versions of Python

Python Version	Released Date
Python 1.0	January 1994
Python 1.5	December 31, 1997
Python 1.6	September 5, 2000
Python 2.0 Python 2.1	October 16, 2000 April 17, 2001
Python 2.2	December 21, 2001
Python 2.3	July 29, 2003
Python 2.4 Python 2.5	November 30, 2004 September 19, 2006
Python 2.6	October 1, 2008
Python 2.7 Python 3.0	July 3, 2010 December 3, 2008
Python 3.1	June 27, 2009
Python 3.2	February 20, 2011
Python 3.3	September 29, 2012
Python 3.4	March 16, 2014
Python 3.5	September 13, 2015.
Python 3.6 Python 3.6.4	December 23, 2016 December 19, 2017
Python 3.7.0	June 27, 2018

पायथन एप्लीकेशन्स (Python Applications)

पायथन अपने सामान्य उद्देश्य प्रकृति के लिए जाना जाता है जो इसे सॉफ्टवेयर डेवलपमेंट के लगभग प्रत्येक क्षेत्र में लागू करता है। पायथन डेवलपमेंट के किसी भी क्षेत्र में इस्तेमाल किया जा सकता है। यहाँ, हम उन क्षेत्रों को निर्दिष्ट कर रहे हैं, जहाँ पायथन को लागू किया जा सकता है।



1. वेब एप्लीकेशन (Web Applications)

हम वेब एप्लीकेशन डेवलप करने के लिए पायथन का उपयोग कर सकते हैं । यह इंटरनेट प्रोटोकॉल जैसे कि **HTML** और **XML, JSON, Email Processing, Request, Beautifulsoup, Feedparser** आदि को संभालने के लिए लाइब्रेरी प्रदान करता है । यह वेब एप्लीकेशन को डिजाइन करने और हटाने के लिए **Django, Pyramid, Flask** आदि जैसे फ्रेमवर्क भी प्रदान करता है ।
कुछ महत्वपूर्ण डेवलपमेंट हैं – **PythonwikiEngines, Pocoo, PythonBlogsoftware** आदि ।

2. डेस्कटॉप जीयूआई एप्लीकेशन (Desktop GUI Applications)

पायथन आधारित एप्लीकेशन में यूजर इंटरफेस विकसित करने के लिए **Tk GUI Library** प्रदान करता है । कुछ अन्य उपयोगी टूलकिट्स **Wxwidgets, Kivy, Pyqt** जो कई प्लेटफार्मों पर उपयोग करने योग्य हैं । मल्टीवॉक एप्लीकेशन लिखने के लिए **Kivy** लोकप्रिय है ।

3. सॉफ्टवेयर डेवलपमेंट (Software Development)

पायथन सॉफ्टवेयर डेवलपमेंट प्रोसेस के लिए सहायक है । यह एक सपोर्ट लैंग्वेज के रूप में काम करता है और इसका इस्तेमाल बिल्ड कंट्रोल और मैनेजमेंट, टेस्टिंग आदि के लिए किया जा सकता है ।

4. वैज्ञानिक और न्यूमेरिक (Scientific and Numeric)

पायथन लोकप्रिय और व्यापक रूप से वैज्ञानिक और संख्यात्मक कम्प्यूटिंग में उपयोग किया जाता है कुछ उपयोगी लाइब्रेरी और पैकेज **SciPy, Pands, IPython** आदि हैं । **SciPy** इंजीनियरिंग, विज्ञान और गणित के पैकेज का समूह है ।

5. व्यावसायिक एप्लीकेशन (Business Applications)

पायथन का उपयोग ईआरपी और ई–कॉमर्स सिस्टम जैसे **Bussines application** के निर्माण के लिए किया जाता है । **Tryton** एक हाई लेवल एप्लीकेशन प्लेटफॉर्म है ।

6. कंसोल आधारित एप्लीकेशन (Console Based Application)

हम कंसोल आधारित एप्लीकेशन्स को विकसित करने के लिए पायथन का उपयोग कर सकते हैं । उदाहरण के लिए – **IPython|**

7. ऑडियो या वीडियो आधारित एप्लीकेशन (Audio or video based Applications)

पायथन कई कार्यों को करने के लिए बहुत बढिया है और इसका उपयोग मल्टीमीडिया एप्लीकेशन्स को विकसित करने के लिए किया जा सकता है । वास्तविक एप्लीकेशन्स में से कुछ हैं – **TimPlayer, Cplay** आदि ।

8. 3D CAD एप्लीकेशन (3D CAD Applications)

CAD एप्लीकेशन बनाने के लिए **Fandango** एक रियल एप्लीकेशन है जो **CAD** की पूर्ण विशेषताएँ प्रदान करता है ।

9. एंटरप्राइज एप्लीकेशन (Enterprise Applications)

पायथन का उपयोग उन एप्लीकेशन्स को बनाने के लिए किया जा सकता है जो किसी एंटरप्राइज या किसी संगठन के भीतर उपयोग किए जा सकते हैं । कुछ रियल टाइम एप्लीकेशन जैसे– **OpenErp, Tryton, Picalo** आदि ।

10. इमेज के लिए एप्लीकेशन (Applications for Images)

पायथन का उपयोग करके इमेज के लिए कई एप्लीकेशन विकसित किए जा सकते हैं । विकसित किए गए एप्लीकेशन हैं जैसे– **VPython, Gogh, imgseek** आदि ।

पायथन डाटा टाइप्स (Python Data Types)

Data Types डाटा आइटम का **Classification** या **Categorization** है । यह उस तरह के **Value** का प्रतिनिधित्व करता है जो बताता है कि किसी विशेष डाटा पर कौन से ऑपरेशन किए जा सकते हैं । चूँकि पायथन प्रोग्रामिंग में सब कुछ एक **Object** है, **Data Type** वास्तव में **Classes** है और **Array** इन **Classes** के ऑब्जेक्ट है ।

Variable विभिन्न **Data Types** के मान रख सकता है । पायथन एक डायनामिक लैंग्वेज है इसलिए हमें इसे **Declare** करते समय **Variable** के प्रकार को **Define** करने की आवश्यकता नहीं है । पायथन हमें प्रोग्राम में उपयोग किए जाने वाले **Variable** के प्रकार की जाँच करने में सक्षम बनाता है । पायथन हमें **Type** () फंक्शन प्रदान करता है ।

उदाहरण –

```
A =10
B = "Hi Python"
C = 10.5
print(type(a));
print(type(b));
print(type(c));
```

Output:

```
<type 'int'>
<type 'str'>
<type float'>
```

Standard data types

Variable विभिन्न प्रकार की value को धारण कर सकता है । उदाहरण के लिए, किसी व्यक्ति के नाम को एक स्ट्रिंग के रूप में स्टोर किया जाना चाहिए, जबकि इसकी आईडी को Integer के रूप में स्टोर किया जाना चाहिए । पायथन विभिन्न Standard Data Types प्रदान करता है जो उनमें से प्रत्येक पर Storage Method को डिफाइन करता है । पायथन में Data Types नीचे दिए गए है–

1. Numbers
2. String
3. List
4. Tuple
5. Dictionary

1. Numbers

- Number, Numeric value को स्टोर करता है । जब एक Variable के लिए एक नंबर सौंपा जाता है तो पायथन नंबर ऑब्जेक्ट बनाता है ।
  उदाहरण– a = 3, b = 5 #a and b are number objects
- पायथन 4 प्रकार के Numeric Data का समर्थन करता है –
  1. **Int** (signed integers जैसे– 10, 2, 29, आदि)
  2. **Long** (long integer का प्रयोग value की Higher Range के लिए किया जाता है । जैसे – 908090800L, –0X1929292L)
  3. **Float**(फ्लोट का उपयोग फ्लोटिंग पॉइंट संख्या जैसे– 1.9, 9.902, 15.2, आदि को स्टोर करने के लिए किया जाता है)
  4. **Complex**(**Complex number** जैसे– 2.14j, 2.0 + 2.3j आदि)

- पायथन हमें Lower case L का उपयोग करने की अनुमति देता है जिसका उपयोग long integer के साथ किया जा सकता है ।
- Complex number में एक Ordered Pair होता है, यानी, x + iy जहाँ x और y क्रमशः वास्तविक और काल्पनिक भागों को दर्शाते हैं ।

2. String

- स्ट्रिंग को Quotation Marks में दर्शाए गए Characters के Sequence के रूप में Define किया जा सकता है । Python में, हम स्ट्रिंग को Define करने के लिए सिंगल (' '), डबल (" ") या ट्रिपल (''') कोट्स का उपयोग कर सकते हैं । Python को संभालना बहुत आसान है क्योंकि इसमें कई इनबिल्ट फंक्शंस और ऑपरेटर्स होते हैं ।
- स्ट्रिंग हैंडलिंग के मामले में, ऑपरेटर + का उपयोग दो स्ट्रिंग्स को जोडने के लिए किया जाता है क्योंकि ऑपरेशन "Hello" + "Python" "Hello Python return करता है ।

उदाहरण –

```
str1 = 'hello' #string str1
str2 = 'how are you' #string str2
print (str1[0:2]) #printing first two character using slice operator
print (str1[4]) #printing 4th character of the string
print (str1*2) #printing the string twice
print (str1 $ str2) #printing the concatenation of str1 and str2
```

Output -

```
He
o
hello hello
hello how are you
```

3. List

- C में array के समान lists हैं । List में विभिन्न प्रकार के डाटा हो सकते हैं । List में store आइटम अल्पविराम () के साथ अलग किए जाते हैं और square brackets [ ] के भीतर संलग्न होते हैं ।
- हम List के डाटा तक पहुँचने के लिए Slice [:] ऑपरेटरों का उपयोग कर सकते हैं । Concatenation Operator (–) और Repetition Operator (') उसी तरह से List के साथ काम करते हैं, जैसे स्ट्रिंग्स के साथ ।

उदाहरण –

```
l = [1, "hi", "python", 2]
print (l[3:]);
print (l[0:2]);
print(l);
print (l+l);
print (l*3);
```

Output -

```
[2]
[1, 'hi']
[1, 'hi', 'python', 2]
[1, 'hi', 'python', 2, 1, 'hi', 'python',2]
[1, 'hi', 'python', 2, 1, 'hi', 'python',2,1, 'hi', 'python', 2]
```

4. Tuple

Tuple कई मायनों में List के समान है । Lists की तरह, Tuple में विभिन्न Data Types के कंटेंट का कलेक्शन भी होता है । Tuple के Items को अल्पविराम (,) के साथ अलग किया जाता है और कोष्ठकों () में संलग्न किया जाता है ।

Tuple एक रीड–ओनली डाटा स्ट्रक्चर है क्योंकि हम Tuple की वस्तुओं के Size और Value में सुधार नहीं कर सकते हैं ।

उदाहरण –

```
t = ("hi", "python", 2)
print (t[1:]);
print (t[0:1]);
print (t);
print (t+t);
print (t*3);
print (type(t))
t[2] = "hi";
```

Output -

```
('python',2)
('hi',)
('hi', 'python', 2)
```

```
('hi', 'python', 2, 'hi', 'python', 2)
('hi', 'python', 2, 'hi', 'python', 2, 'hi', 'python', 2)
<type 'tuple'>
Traceback (most recent call last):
File "main.py", line 8, in <module>
t[2] = "hi";
TypeError: 'tuple' object does not support item assignment
```

5. Dictionary

- Dictionary आइटम के Key value वाले जोडे का एक व्यवस्थित सेट है । यह एक Associative Array या Hash Table की तरह है जहाँ प्रत्येक Key एक Specific Value स्टोर करती है ।
- Keys किसी भी Primitive Data Type को पकड सकती है जबकि Value एक Arbitrary पायथन ऑब्जेक्ट है ।
- Dictionary में आइटम अल्पविराम के साथ अलग किये गए है और Curly Braces {} में संलग्न है ।

उदाहरण –

```
d = {1:'Jimmy'] 2:'Alex'] 3:'john'] 4:'mike'};
print("1st name is "+d[1]);
print("2nd name is "+d[4]);
print (d);
print (d.keys());
print (d.values());
```

Output

```
1st name is Jimmy
2nd name is mike
{1: 'Jimmy', 2: 'Alex', 3: 'john', 4: 'mike'}
[1,2,3,4]
['Jimmy', 'Alex', 'john', 'mike']
```

Python - Types of Variable

Python में Variable के दो प्रकार है –

1. Local Variables
2. Global Variables

1. Local Variables

Local variables, functions के अन्दर होते है । उनकी visibility सिर्फ function के अन्दर होती है, जब वो function के बाहर आते है तब destroy हो जाते है ।

Source Code

```
def func():
a = 5 #local variable
print(a)
```

```
func()
print(a)
```

Output

```
5
Traceback (most recent call last):
print(a)
NameError: name 'a' is not defined
```

2. Global Variables

Global Variables, function के बाहर होते हैं । उनकी Visibility Function के अन्दर और बाहर होती है, उनका scope पूरे program पर होता है ।

Source Code

```
a - 10 #global variable
def func():
print(a)
func()
print(a)
```

Output

```
10
10
```

Example पर local और global ये दोनों variables declared किये गए हैं । function के बाहर का variable global है और अन्दर का variable local है । global variable का scope function के अन्दर और बाहर होता है, लेकिन function के अन्दर अलग से variable declaration होने के कारण func() call करते ही variable की value change हो जाती है ।

Source Code

```
a = 10 #global variable
def func():
a = 5 #local variable
print(a)
func() #print local
print(a) #print global
```

Output

```
5
10
```

Python & Indenting Code

Python में Code indentation को काफी महत्व दिया गया है । Python में Code Indentation का इस्तेमाल functions, classes, Loops और control statements के लिए किया जाता है ।

Python में curly braces ({}) की जगह code indentation का इस्तेमाल किया जाता है ।

Python में जब code indentation में colon (:) या delimiter दिया जाता है तब automatically अगले line पर interpreter द्वारा tab () दिया जाता है ।

Function

Source Code

```
def func():
a = 5
print(a)
func()
```

5

For Loop

Source Code

```
list = [1, 5, 8, 9]
for i in list
print(i)
```

Output

1
5
8
9

पाइथन ऑपरेटर्स (Python Operators)

ऑपरेटर को एक सिंबल (symbol) के रूप में परिभाषित किया जा सकता है, जो दो ऑपरेंड के बीच एक विशेष ऑपरेशन के लिए जिम्मेदार है । पाइथन में निम्नलिखित ऑपरेटर होते है –

1. Arithmetic operators
2. Comparison operators
3. Assignment Operators
4. Logical Operators
5. Bitwise Operators
6. Membership Operators
7. Identity Operators

1. Arithmetic operators

अंकगणित ऑपरेटरों का उपयोग दो ऑपरेंड के बीच अंकगणितीय क्रियाएँ करने के लिए किया जाता है । इसमें +(जोड), -(घटाव), *(गुणा), /(विभाजित), %(रिमाइंडर), //(फ्लोर डिवीजन), और एक्स्पोनेंट (**) शामिल हैं –

Operator	Name	Example
+	Addition	X+Y
-	Subtraction	X-Y
*	Multiplication	X*Y
/	Division	X/Y
%	Modulus	X%Y
**	Exponentiation	X**Y
//	Floor division	X//Y

2. Comparison operator

- Comparison operator का उपयोग दो ऑपरेंड के बीच वैल्यू की तुलना करने के लिए किया जाता है और यह बूलियन को True या false लौटाता है ।
- तुलना ऑपरेटरों को निम्न तालिका में वर्णित किया गया है –

Operator	Name	Example
==	Equal	x==y
!=	Not equal	x !=y
>	Greater than	x>y
<	Less than	x<y
>=	Greater than or equal to	x>=y
<=	Less than or equal to	x<=y

3. Assignment operators

असाइनमेंट ऑपरेटर्स का उपयोग वैरिएबल का मान असाइन करने के लिए किया जाता है । असाइनमेंट ऑपरेटरों को निम्न तालिका में वर्णित किया गया है–

Operator	Example	Example
=	X=5	X=5
+=	X+=3	X=X+
-=	X-=3	X=X-3
*=	X*=3	X=X*3
/=	X/=3	X=X/3
%=	X%=3	X=X%3
//=	X//=3	X=X//3
**=	X**=3	X=X**3
&=	X&=3	X=X&3
|=	X|=3	X=X|3
^=	X^=3	X=X^3
>>=	X>>=	X=X>>3
<<=	X<<=3	X=X<<3

4. Bitwise operator

बिटवाइज़ ऑपरेटरों का उपयोग बाइनरी संख्याओं की तुलना करने के लिए किया जाता है ।

OPERATOR	DESCRIPTION	Example
&	Bitwise AND	X&Y
|	Bitwise OR	X | Y
~	Bitwise NOT	-X
^	Bitwise XOR	X^y
>>	Bitwise right shift	x>>
<<	Bitwise left shift	X<<

5. Logical Operators

- लॉजिकल ऑपरेटरों का उपयोग मुख्य रूप से Combine Condition Statement के लिए किया जाता है ।
- पायथन निम्नलिखित तार्किक ऑपरेटरों का समर्थन करता है ।

Operator	Description	Example
And	यदि दोनों कथन सत्य हैं, तो True लौटाता है	X < 5 and x < 10
Or	यदि कथन में से कोई एक सत्य है तो True लौटाता है	X < 5 or x < 4
not	परिणाम को उल्टा करें, यदि परिणाम सही है तो False लौटाता है ।	not (x < 5 and x < 10)

6. Membership Operators

- यदि किसी sequence को किसी ऑब्जेक्ट में present किया जाता है, तो Membership Operators का परीक्षण किया जाता है ।

- पायथन Membership operators का उपयोग पायथन डाटा स्ट्रक्चर के अंदर वैल्यू की मेम्बरशिप की जाँच करने के लिए किया जाता है । यदि मान डाटा स्ट्रक्चर में मौजूद है, इसलिए इसमें रिजल्ट True और False लौटाता है ।

Operator	Description	Example
In	यदि ऑब्जेक्ट में निर्दिष्ट मान वाला कोई sequence मौजूद है, तो यह True लौटाता है ।	X in y
not in	यदि ऑब्जेक्ट में निर्दिष्ट मान के साथ कोई sequence मौजूद नहीं है, तो यह True लौटाता है ।	x not in y

7. Identity Operators

- दो ऑब्जेक्ट की मेमोरी लोकेशन की तुलना करने के लिए, आइडेंटिटी ऑपरेटर्स का उपयोग किया जाता है ।
- पायथन में उपयोग किए जाने वाले दो Identity Operators is, is not।

Operator	Description	Example
Is	यदि दोनों variable एक ही object है, तो True लौटाता है ।	X is y
is not	यदि दोनों Variable एक Object नहीं है, तो True लौटाता है ।	x is not y

Example

```
x = 20
y = 20
if ( x is y):
print("x & y SAME identity")
y=30
if ( x is not y):
print("x & y have DIFFERENT identity")
```

Python - Math Constants

- Python में numbers पर mathematical operations करने के लिए math functions का इस्तेमाल किया जाता है ।
- Python के program में अगर math functions का इस्तेमाल करना हो, तो 'math' module का इस्तेमाल किया जाता है ।

Math Constants in Python

Math Constant	Description
e	Math की 'E' Property Euler का number return करता है ।
inf	infinity को return करता है ।
nan	not a number (nan) का वर्णन करता है ।
pi	pi की value को return करता है ।
tau	tau(π) की value को return करता है ।

Python - Math Functions

- Python में numbers पर mathematical operations करने के लिए math functions का इस्तेमाल किया जाता है ।
- Python के program में अगर math functions का इस्तेमाल करना हो, तो 'math' module का इस्तेमाल किया जाता है ।

Math Functions in Python

Math Function	Description
acos()	दिए हुए number का arc cosine; radians में return करता है ।
acosh()	दिए हुए number का inverse hyperboliccosine; को return करता है ।
asin()	दिए हुए number का arcsine; radians में return करता है ।
asinh()	दिए हुए number का inverse hyperbolicsine; को return करता है ।
atan()	दिए हुए number का arctangent; radians में return करता है ।
atan()	atan(y/x) को radians में return करता है ।
atanh()	दिए हुए number का inverse hyperbolic tangent; को return करता है ।
ceil()	दिए हुए number की ceiling value को return करता है ।
copysign()	दिए हुए x की value पर y के sign को copy करके x को return करता है ।
cos()	दिए हुए number का cosine; radians में return करता है ।
cosh()	दिए हुए number का hyperbolic cosine; return करता है ।
degrees()	दिए हुए number के angle को radians से degrees में return करता है ।
exp()	दिए हुए number का exponential return करता है ।
expm1	दिए हुए number का exponential से '-1' करके return करता है ।
fabs()	दिए हुए number की absolute value को return करता है ।
factorial()	दिए हुए number का factorial को return करता है ।
floor()	दिए हुए number की floor value को return करता है ।
fmod()	दिए हुए x को y से divide करके उनका remainder return करता है ।
frexp()	दिए हुए number से manissa और exponent को (m,e) इस tuple के रूप में return करता है ।
fsum()	दिए हुए sequence या collection के items का sum return करता है ।
gcd()	दिए हुए x और y का gcd(Greatest common Divisor) return करता है ।
hypot()	दिए हुए x side और y side की मदद से hypotenuse(कर्ण) floating-point number में return करता है ।
isclose()	दिए हुए x और y ये close है या नहीं ये boolean value में return करता है–
isfinite()	दिए हुए number funite है या नहीं ये Boolean value में return करता है ।
isinf()	दिए हुए number infunite है या नहीं, ये Boolean value में return करता है ।
Idexp()	x* (2**i) floating-point number में return करता है ।
log()	दिए हुए number का natural logarithm return करता है ।
log10()	दिए हुए Number का base-10logarithm return करता है ।
log1p()	दिए हुए Number पर '+1' करके उसका natural logarithm return करता है ।
log2()	दिए हुए Number का base-2 logarithm return करता है ।

modf()	दिए हुए floating-point या integer number का fractional part और integer part को tuple में return करता है ।
pow()	x**y return करता है ।
radians()	दिए हुए number के angle को degrees से radians में return करता है ।
sin()	दिए हुए number का sine; radians में return करता है ।
sinh()	दिए हुए number का hyperbolic sine; return करता है ।
sqrt()	दिए हुए number का square root return किया जाता है ।
tan()	दिए हुए number का tangent; radians में return करता है ।
tanh()	दिए हुए number का hyperbolic tangent; return करता है ।
trunc()	दिए हुए number का अगर fraction part होता है तो उसे remove करके integer value को return किया जाता है ।

Python - Constructor

- Python के अलावा C++ और java में भी constructor होता है लेकिन उन constructor में खुद class के नाम से ही constructor को बनाया जाता है । लेकिन Python में constructor को create करने के लिए_init_() function का इस्तेमाल किया जाता है ।
- जब class में _init _() function define किया जाता है और उस class का object बनाया जाता है, तब ये function automatically call हो जाता है अर्थात् उसे अलग से call करने की जरुरत नहीं पडती है ।

Syntax for Constructor in Python

```
class className:
class_body (Optional)
def _init_(parameter(s)): #Constructor
constructor_body
class_body (optional)
```

Python - Destructor

C++ और Java में destructor को _(tilde) sign के साथ class के नाम की जरुरत पडती और वो object बनते ही अपने आप call होता है ।

लेकिन Python में Destructor के लिए '_del_() function का इस्तेमाल किया जाता है और जब class का object बनाया जाता है तब वो automatically call नहीं होता है ।
Destructor को destroy करने के लिए 'del' operator से object को delete करना पडता है ।

Example for Destructor in Python

Source Code

```
class MyClass:
def _init_(self, a, b):
self.a = a
self.b = b
print(self.a, self.b)
print("Constructor invoked")
```

```
def _del__(self):
print("Destructor invoked")

if __name__ == "_main_":
obj = MyClass (4, "Hello")
del obj
```

Output

```
4 Hello
Constructor invoked
Destructor invoked
```

Python - Class Attributes

Python में हर class के साथ In-Built Class Attributes होते हैं ।

doc – ये दिए गए class की docstring को return करता है अगर docstring नहीं होती है तो 'None' return होता है ।

name – ये दिए गए class का नाम return करता है ।

module – जहाँ पर class होता है, वो module name return किया जाता है । अगर class current program पर होता है तो '_main_'return होता है ।

bases – ये दिए गए **Class** का **base class** में **tuple** में **return** करता है । अगर कोई **base class** नहीं होता है, तो **empty tuple return** करता है ।

dict – ये दिए गए class के attributes को dictionary में return करता है ।

Example for Class Attributes in Python

Source Code

```
class Employee:
"I am in class Employee"
pass
class Fitness (Employee):
"I am in class Fitness"
pass
class Company (Fitness):
"I am in class Company"
pass

print("Docstring of class Company :", Company. _doc_)
print("Class Name :", Company. _name_)
print("Module Name :", Company. _module_)
print("Base Class of Company :", Company. _bases_)
```