A/L

தகவல் தொடர்பாடல் தொழில்னுட்பவியல்

Information and Communication Technology

Programming

2019 Batch

PYHTHON PROGRAMMING LANGUAGE

பைத்தன் செய்நிரல் மொழியானது வரிமொழிபெயர்ப்பு அடிப்படையில் அமைந்துள்ள இலக்குப்பொருள் சார்ந்ததாக அமைந்திருக்கும் உயர்மட்டச் செய்நிரல் மொழியாகும்.

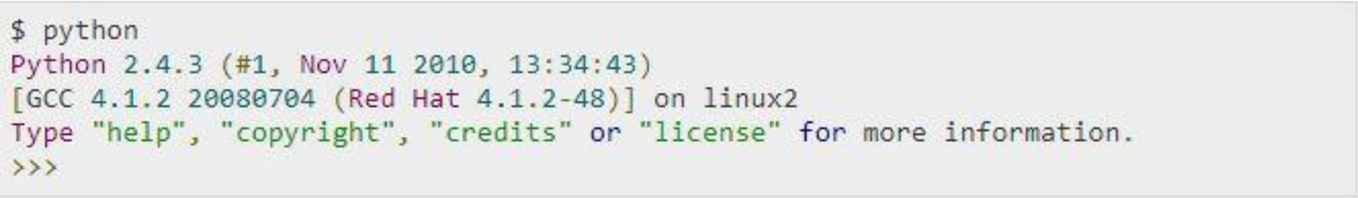
பைத்தன் செய்நிரல் மொழியினை http://www.python.org/ எனும் இணையத்தளத்தின் மூலம் பதிவிறக்கம் செய்து பெற்றுக்கொள்ள முடியும்

Python’s feature

1. கற்பதற்கு இலகுவானது (Easy to Learn)
2. வாசிப்பதற்கு இலகுவானது (Easy to read)
3. பராமரிப்பதற்கு இலகுவானது (Easy to maintain)
4. A broad standard library
5. Interactive
6. Portable
7. Extendable
8. Databases
9. GUI Programming
10. Case sensitive

Python program ஐ கணினியில் திறக்கும் (Open) முறை

- Start Menu விற்கு சென்று Python இல் IDLE (Integrated Development Environment) ஐ Click செய்து Open செய்ய முடியும் . இது “Python Shell” என்ற பெயரில் காணப்படும்.
- தோன்றும் IDLE இல் >>> எனும் குறியீடு காணப்படும். அதில் தேவையான குறிமுறைகளை (Coding) ஒவ்வொரு வரிசையிலும் (Line) எழுதி வரி வரிசையாகச் செயற்படுத்திக் (Run) கொள்ளமுடியும்.
- கோப்பினைச் (File) சேமிக்கும் போது (Save) கோப்பு நீட்சியானது (File Extension) .py OR .pyw என கொடுத்துச் சேமிக்க வேண்டும்



```
$ python
Python 2.4.3 (#1, Nov 11 2010, 13:34:43)
[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

பைத்தன் செய்நிரல் மொழிகளில் ஏற்படக் கூடிய வழுக்கள்

1. தொடரியல் வழுக்கள் (Syntax Errors)
2. சொற்பொருளியல் வழுக்கள் (Semantic Error)

தொடரியல் வழுக்கள் (Syntax Errors)

தொடரியல் வழுக்களானது பைத்தன் செய்நிரலில் காணப்படும் மூலச்சொற்களில் (Keyword) ஏற்படுகின்ற எழுத்துப் பிழைகள் மற்றும் உரிய குறியீடுகள் பயன்படுத்தப்படாமையினால் உருவாகின்றது.

செய்நிரலானது மொழிபெயர்ப்புச் செய்யப்படுகின்ற பொழுது பிழைகளுக்குரிய வரிகள் சுட்டிக் காட்டப்படும். ஆகவே தொடரியல் வழுக்குளுடன் கூடிய செய்நிரல் அதன் முடிவு வரை இயங்க மாட்டாது. அச் செய்நிரலுக்குரிய வெளியீட்டினையும் பெற முடியாது. வழுக்கள் திருத்தப்பட்டு மீண்டும் அச் செய்நிரலை செயற்படுத்துவதன் மூலம் வெளியீட்டினைப் பெற முடியும்.

தொடரியல் வழுக்கள் ஏற்படக்கூடிய சந்தர்ப்பங்கள்

1. மூலச் சொல்லில் (Keyword) ஏற்படும் எழுத்துப் பிழைகள்
2. தேவையான இடங்களில் colon (:) குறியீடு இடப்படாமை
3. உரிய இடங்களில் அடைப்புக்குறிகள் தொடங்கப்பட்டு முடிக்கப்படாமை.
4. சொற்களிற்கு (String) " " எனும் குறியீடு இடப்படாமை
5. சரியான முறையில் உட்தள்ளல் (indentation) இடப்படாமை

சொற்பொருளியல் வழுக்கள் (Semantic Error)

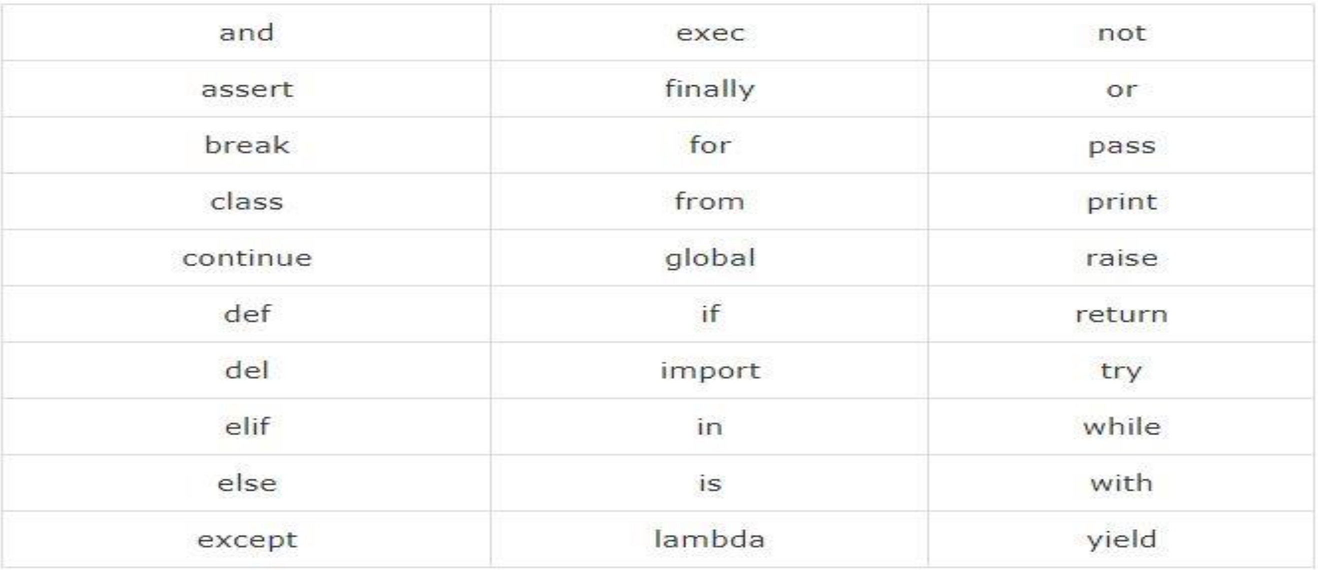
குறித்த பைத்தன் செய்நிரலில் குறியீடு எது விதமான வழுக்களும் இன்றி செய்நிரலாளரினால் இடப்படுகின்ற தவறுகள் இவற்றுள் அடங்கும். அதாவது செய்நிரலிற்குரிய வெளியீடு பெறப்படும் ஆனால் அவ் வெளியீடு தவறானதாகக் காணப்படும்

சொற்பொருளியல் வழுக்களுடன் கூடிய செய்நிரலானது இறுதி வரை இயங்கி வெயியீட்டினைத் தவறாகத் தரும்

சாவிச் சொற்கள் (Keywords & Reserved words)

குறிப்பிட்ட ஒரு செய்நிரலில் சில விசேட செயற்பாடுகளினை செய்வதற்கு மட்டும் ஒதுக்கப்பட்டுள்ள சொற்கள் Keywords ஆகும். மாறிகளிற்கு வழங்கப்பட்டிருக்கும் பெயர்களையும் Keywords ஐயும் சேர்த்து Reserved words குறிப்பிட முடியும். Keywords இனை மாறிகளின் பெயர்களிற்கு பயன்படுத்த முடியாது

Keywords



and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

மாறி ஒன்றிற்கு பெயரிடும் போது கவனிக்க வேண்டியவை

- மாறி ஒன்றின் பெயர் ஆனது இலக்கங்கள் எழுத்துக்கள் underscore (_) என்பவற்றைக் கொண்டு அமையலாம்
- மாறிகளின் பெயர்கள் எழுத்துக்களில் அல்லது underscore (_) இல் ஆரம்பிக்கலாம். ஆனால் இலக்கங்களில் ஆரம்பிக்கக் கூடாது.
- மாறிகளின் பெயர்களினுள் இடைவெளி இட முடியாது. அதேவேளை எண்கணித தர்க்கச் செய்கைக்குறியீடுகளையும் இட முடியாது.
- ஏனைய விசேட குறியீடுகள் (!, @, #, $, %, ^, &, *, ,) பயன்படுத்த முடியாது.
- பைத்தன் செய்நிரலின் கலைச் சொற்கள் மாறிலிகளின் பெயர்களாகப் பணன்படுத்துதல் தவிர்க்கப்படல் வேண்டும் (if, for, while, break, print)
- மாறிகளின் பெயர்களில் இடம் பெறும் எழுத்துக்களின் எண்ணிக்கைக்கு உச்ச எல்லை கிடையாது.

Example:

Name சரியானது (Correct)

Student_Name சரியானது (Correct)

Mark1 சரியானது (Correct)

8Angle தவறானது (Incorrect)

Student Name தவறானது (Incorrect)

Name# தவறானது (Incorrect)

@school தவறானது (Incorrect)

If தவறானது (Incorrect)

123 தவறானது (Incorrect)

-school தவறானது (Incorrect)

வசனத்தை காட்சிப்படுத்தும் (Print) முறை

சொற்கள், வசனங்கள் குறியீடுகளை Print செய்யும் போது “ ” இனுள் அல்லது ‘ ’ இனுள் கொடுக்கப்படல் வேண்டும்.

Example: 01

```
>>> print "Hello, Python!"
```

Output :

Example: 02

```
print "python"
```

Output :

Example: 03

```
print ‘python’
```

Output :

Example: 04

```
print "Nimal\’s age"
```

Output :

Example: 05

```
print "Roy said,\ "hello\" to me"
```

Output :

Example: 06

```
print "Name\t", "Age"
```

Output :

Example: 07

```
print "Name:\nSchool:\nGrade:"
```

Output :

Note:

- புதியவரியில் Type செய்வதற்கு (Enter/New Lien) :- \n
- Tab மூலம் இடைவெளி அமைப்பதற்கு (Tab Key) :- \t
- Tab மூலம் அமைக்கப்பட்ட இடைவெளியை நீக்குவதற்கு :- \r
- \ எனும் குறியீட்டினை இடுவதற்கு :- \\
- ‘ எனும் குறியீட்டினை இடுவதற்கு :- \’
- “ எனும் குறியீட்டினை இடுவதற்கு :- \”

இவை அனைத்தும் “ ” இனுள் இடப்படுதல் அவசியமாகும்

Explicit line Joining

ஒன்றுடன் ஒன்று தொடர்புபட்ட ஒரே வரியில் அமைய வேண்டிய குறிமுறையை அடுத்த வரியிலும் எழுத வேண்டிய தேவை ஏற்படும் சந்தர்ப்பத்தில் \ எனும் குறியீடு பயன்படுத்தப்படும்

```
>>> x, y, z =\

      2, 3, 5
```

Implicit line joining

List, Tuple, Dictionary ஆகியவற்றை [], (), {} ஆகிய அடைப்புக்குறிகளினுள் வரையறை செய்யும் போது பல வரிசைகளில் அமையலாம்

```
      X= { ‘x’:5, ‘y’:

            10, ‘z’:20}
```

```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2  )

list = [ 'abcd', 786 , 2.23, 'john', 70.2  ]

tuple[2] = 1000

list[2] = 1000
```

எண்களை காட்சிப்படுத்தும் (Print) முறை

எண்களை print செய்யும் போது “ ” இனுள் அல்லது ‘ ’ இனுள் இடக்கூடாது. “ ” இனுள் அல்லது ‘ ’ இனுள் இடப்படும் போது இலக்கங்கள் சொற்களாகக் கருதப்படும் அதனால் கணித்தல் செயற்பாடுகள் எதனையும் மேற்கொள்ளமுடியாது.

```
>>> print (25)
```

Output:

```
>>> print ("My age is ",24)
```

Output:

Note:

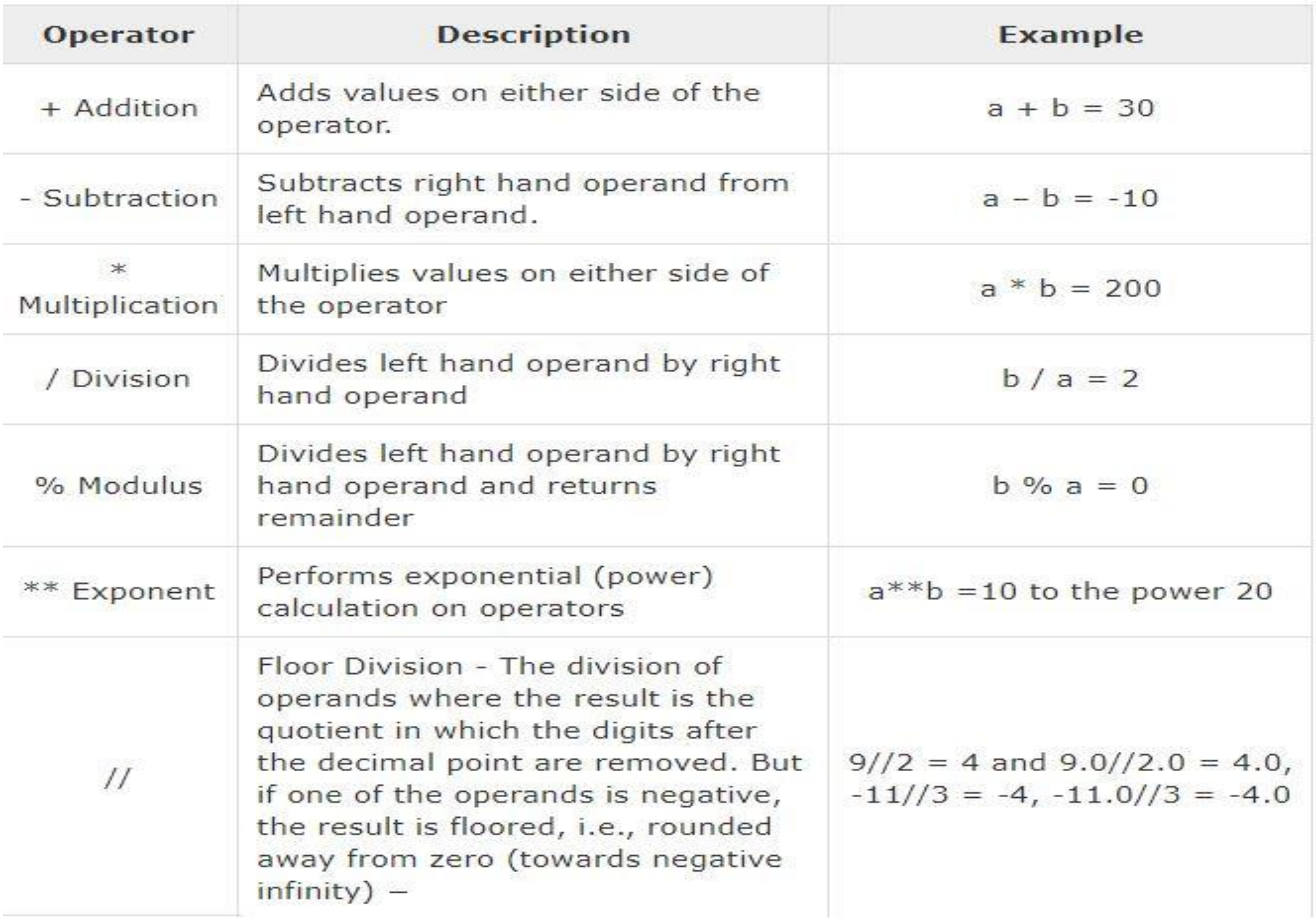
ஒரே வரிசையில் காணப்படும் பைத்தன் கூற்றுக்களை வேறுபடுத்துவதற்கு ; எனும் குறியீட்டினைப் பயன்படுத்தலாம்

குறிப்புரை (Comments) வழங்கும் முறை

1. குறிப்புரைகள் (Comments) செய்நிரலினை (Program) விபரிப்பதற்கு பயன்படுத்தப்படும். அதாவது செய்நிரலைப் பார்வையிடுவோர் அதனை விளங்கிக் கொள்வதற்கு குறிப்புரைகள் உதவுகின்றன.
2. பைத்தன் செய்நிரலில் # குறியீட்டிற்குப் (Sign) பின்னர் அந்த வரியில் வழங்கப்படும் எல்லா விடயங்களும் குறிப்புரைகள் ஆகும்.
3. # குறியீட்டிற்கு (Sign) பின்னர் அந்த வரியில் இடம் பெறும் எந்தவிடயமும் run ஆக மாட்டாது.
4. இவை வெவ்வேறு மொழிகளுக்கு வேறு வேறு குறியீடுகளை கொண்டிருக்கும்
   உதாரணம்: \\java comments

```
# This is a comment.
# This is a comment, too.
# This is a comment, too.
# I said that already.
```

எண்கணிதச் செய்கைகள் (Arithmetic Operators)



Operator	Description	Example
+ Addition	Adds values on either side of the operator.	a + b = 30
- Subtraction	Subtracts right hand operand from left hand operand.	a – b = -10
* Multiplication	Multiplies values on either side of the operator	a * b = 200
/ Division	Divides left hand operand by right hand operand	b / a = 2
% Modulus	Divides left hand operand by right hand operand and returns remainder	b % a = 0
** Exponent	Performs exponential (power) calculation on operators	a**b =10 to the power 20
//	Floor Division - The division of operands where the result is the quotient in which the digits after the decimal point are removed. But if one of the operands is negative, the result is floored, i.e., rounded away from zero (towards negative infinity) −	9//2 = 4 and 9.0//2.0 = 4.0, -11//3 = -4, -11.0//3 = -4.0

இங்கு செயற்பாடுகளை மேற்கொள்ளும் போது முன்னுரிமை அடிப்படையிலும் சம முன்னுரிமை உடையனவாகக் காணப்படின் இடமிருந்து வலமாகவும் செயற்படும். இம் மதிப்பீட்டின் அடிப்படை ஒழுங்கினை தேவையான இடங்களில் () இடுவதன் மூலம் மாற்றி அமைக்க முடியும். அடைப்புக்குறிக்குள் காணப்படும் விடயங்கள் மிகவும் உள் இடப்பட்ட அடைப்புக்குறியில் தொடங்கி வெளியே இடப்பட்ட அடைப்புக்குறியினுள் இடப்பட்ட விடயம் வரை மதிப்பிடப்படும்.

```
>>> 1 + 2 * 3
```

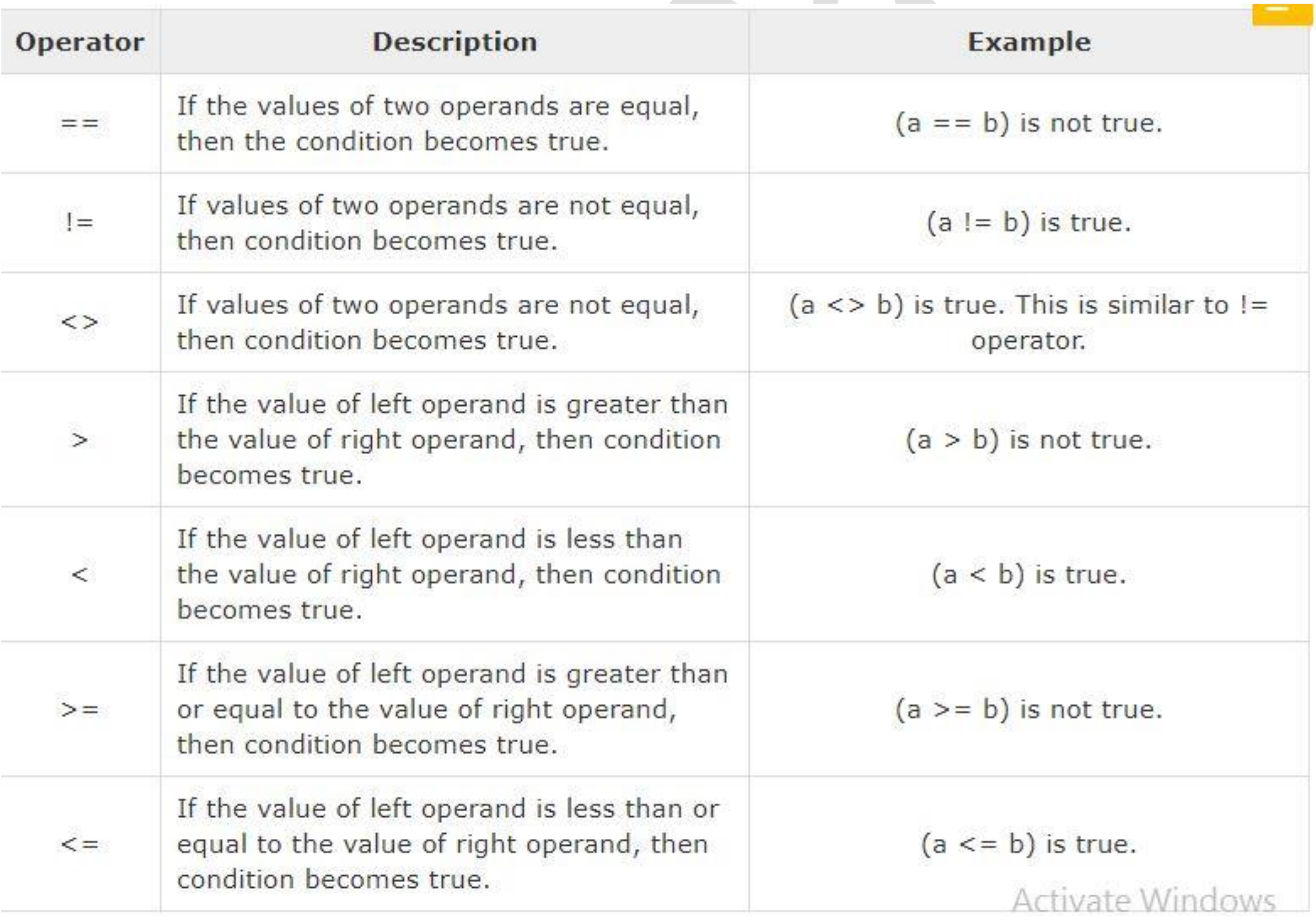
Output:

```
>>> (1 + 2) * 3
```

Output:

ஒப்பீட்டுச் செய்கைகள் (Comparison Operators / Relational Operators)

வழங்கப்படும் எண்களிற்கு எண்களிற்கு இடையே ஒப்பீடு நடைபெற்று உண்மை, பொய் (True, False) ஆகியவற்றில் ஒன்று மட்டுமே வருவிளைவாகக் (output) கிடைக்கும்.



Operator	Description	Example
==	If the values of two operands are equal, then the condition becomes true.	(a == b) is not true.
!=	If values of two operands are not equal, then condition becomes true.	(a != b) is true.
<>	If values of two operands are not equal, then condition becomes true.	(a <> b) is true. This is similar to != operator.
>	If the value of left operand is greater than the value of right operand, then condition becomes true.	(a > b) is not true.
<	If the value of left operand is less than the value of right operand, then condition becomes true.	(a < b) is true.
>=	If the value of left operand is greater than or equal to the value of right operand, then condition becomes true.	(a >= b) is not true.
<=	If the value of left operand is less than or equal to the value of right operand, then condition becomes true.	(a <= b) is true.

```
>>> 9==8
```

Output : False

```
>>> 6!=3
```

Output : True

```
>>> 5>=5
```

Output :

தருக்கச் செய்கைகள் (Logical Operators)

இரண்டு அல்லது இரண்டிற்கு மேற்பட்ட ஒப்பீட்டுச் செய்கைகளை இணைத்துப்பார்ப்பதற்கு இவை பயன்படுத்தப்படும் வருவிளைவு உண்மை (True) அல்லது பொய் (False) ஆகிய இரண்டில் ஒன்றாகவே அமையும்.

Operator	Name	Example	Output
Or	or	5>4 or 8>10	True
And	And	5>4 or 8>10	False
not	Not	Not(5<4)	False

Assignment Operators

Command	Example		Output
+=	X=3 X+=4		
-=	Y=75 y-=50		
*=	Z=10 Z*2		
/=	X=25 x/=5		
%=	X=57 X%=4		

Example:

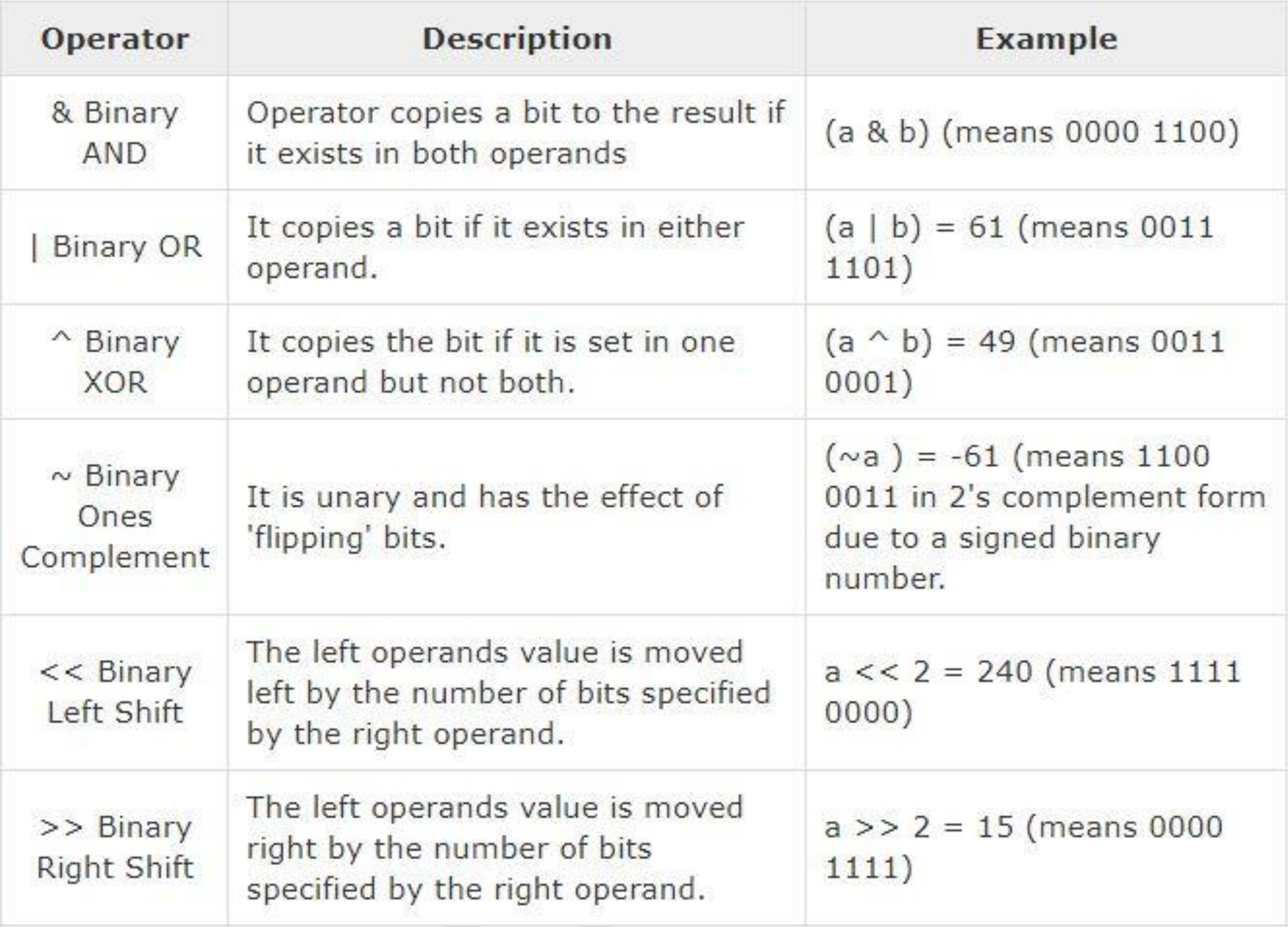
```
>>> x=5

>>> x+=4

>>> print(x)
```

Output:

பிட் சார்ந்த செய்கைகள் (Bitwise Operators)



Operator	Description	Example
& Binary AND	Operator copies a bit to the result if it exists in both operands	(a & b) (means 0000 1100)
| Binary OR	It copies a bit if it exists in either operand.	(a | b) = 61 (means 0011 1101)
^ Binary XOR	It copies the bit if it is set in one operand but not both.	(a ^ b) = 49 (means 0011 0001)
~ Binary Ones Complement	It is unary and has the effect of 'flipping' bits.	(~a) = -61 (means 1100 0011 in 2's complement form due to a signed binary number.
<< Binary Left Shift	The left operands value is moved left by the number of bits specified by the right operand.	a << 2 = 240 (means 1111 0000)
>> Binary Right Shift	The left operands value is moved right by the number of bits specified by the right operand.	a >> 2 = 15 (means 0000 1111)

Example:

a= 60; and b = 13;

a = 0011 1100

b = 0000 1101

a&b = 0000 1100

a|b = 0011 1101

a^b = 0011 0001

~a = 1100 0011