06-11-2020 (16) LECTURE - 3

KANAV BANSAL

BITWISE : The operators of it are used to convert them into binary digits.

A	B	and (&)	or (|)	Exon (^)	Tilda ~
1	1	1	1	0	
1	0	0	1	1	
0	1	0	1	1	
0	0	0	0	0.	

Exon : when bits are different then it is '1'
otherwise it is '0'

Tilda → ~1 ⇒ 0
~0 ⇒ 1

* I/P - Print (5 & 10) — O/P - 0.

5 - 0 1 0 1
10 - 1 0 1 0
0 0 0 0.

* I/P - Print (5 | 10) — O/P - 15

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

$\Rightarrow (1111)_2 = 1\times2^3+1\times2^2+1\times2^1+1\times2^0$
$= 8+4+2+1$
$= 15$

* I/P - percent(5^10) — O/P - 15

5 - 0101
10 - 1010
1111

SHIFT :

<< - left shift — MULTIPLY

>> - Right shift — DIVIDE (Int)

* I/P - 5 << 1 O/P - $(10)_{10}$.

$(0101)_2$ << 1

0101

$101_{\sqcup}$ → 1010 ⇒ $(10)_{10}$.

* I/P - 5 >> 1 O/P - 2.

$(0101)_2$ << 1

0101

$101_{\sqcup}$ ⇒ $(2)_{10}$

* I/P-5 << 2 ⇒ $(20)_{10}$.

O/P - $5 \times 2^2 = 20$.

* I/P - 5 >> 2 ⇒ $5//2^2$

ASSIGNMENT : (18)

↳ =

* I/P - a = 10

a = 10 + 5

Print (a)

O/P - 15

Similarly a+ = 5, a- = 5, a* = 5, a/ = 5

Q. Write a program for unique number.

for given a = 10, b = 20, c = 10.

I/P - a = 10

b = 20

c = 10

Print (a ^ b ^ c)

O/P - 20.

a ^ a = 0
b ^ 0 = b

NLP → GPT Algorithm

↓

GPT 1

GPT 2 (complex one)

* I/P - c = 'a'

```
Print(type(c))
s = 'welcome to class!'
Print(type(s))
Print(len(s))
```

O/P - <class 'str'>
<class 'str'>
17

* I/P - s = 'hi'

```
Print(s[1])
Print(s + 'there')
Print(s, 10)
```

O/P - i
hi there
hi 10

* I/P - s = 'hi'

```
Print(s[1])
Print(s[3])
Print(s[1:0])
Print(s + 'there')
Print(s, 10)
```

O/P - i
o

* I/P - s = 'hi'

Print(s[-1])

O/P - i

0	1
h	i
-2	-1

* I/P - s = ' hi Pavani'

Print(s.upper())

Print(s.Lower())

Print(s.strip()) → removes extra spaces from front & back

O/P - HI PAVANI

hi pavani

hi pavani

* I/P - s = 'we are leaning python!'

Print(s.find('leaning'))

s = s.replace('leaning', 'learning')

Print(s)

split_string = s.split(" ")

Print(split_string)

Print(type(split_string))

join_string = ' '.join(split_string)

```
Print(join - string)
Print(type(join - string)
```

o/p - -1

```
we are learning python!
'we', 'are', 'learning', 'python!'
<class 'list'>
we are learning python!
<class 'str'>.
```

* s[start : end : step size]

```
        0 1 2 3 4 5
S →     P A V A N I    → (n-1)
        1 : 3 : 1
```

*i/p - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

```
Print(s[3:5:1])
```

o/p - de

*i/p - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

```
Print(s[3:10:2])
```

o/p - dfhj

* I/P = s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print(s[3:10:1])

O/P - d e f g h i j

* I/P - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print(s[::2])

O/P - a c e g i k m.

* I/P - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print(s[5:])

O/P - f g h i j k l m

* I/P - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print(s[:5]) → by default it takes as 1

O/P - a b c d e.

* I/P - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print[s[-7:-3:1]]

O/P - g h i j

* I/P - s=['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print[s[::-1]] O/P - m l k j i h g f e d c b

*I/p - s['a','b','c','d','e','f','g','h','i','j','k','l','m']

Print(s[4:0:-1])

O/p - e d c b

isalpha : It returns True if all the characters are alphabets letters (a-z).

*(space)!#%&? etc are ~~not alphabet~~ not allowed.

SYNTAX : str.isalpha()

isdigit : It returns True if all the characters are digits, otherwise False.

* Exponents, like ^{2} are also considered to be a digit.

SYNTAX : str.isdigit()

isspace : It returns True if all the characters in a string are whitespaces, otherwise False.

SYNTAX : str.isspace()

Start with: It returns True if the string starts with the specified value, otherwise False.

SYNTAX : str.startswith (value, start, end)

value - Required; the value to check if the string starts with

start - Optional; An Integer specifying at which position to start the search.

End - Optional; An Integer specifying at which position to end the search.

Endwith: It returns True if the string ends with the specified value, otherwise False.

SYNTAX : str.endswith (value, start, end)