PYTHON

Python is a general purpose p. lang.

→ High level
→ interpreted language

with easy syntax. It was created by Guidovan Rossan in 1989.
(Because he wanted to make a language that beautiful & easy to use.)

* focus on identations, {} removed.

To do this, he used identations instead of curly braces to describe the block of code.

* The language is went on the market, but didn't recieved love, bcz it's slow as compared to other languages.

* Now, with the rise of ML and AI, python came into the spotlight.

* Bcz, it makes the work productive and much easier.

High level languages → (translator) → Low level language
(compiler / interpreter)

S/W
↑
(set of) Program
↑
Set of instruction

Python Prog. language

in Python, type conversion is a process of converting variable type.

[NOTE:] in python, we can't declare variables before assigning the value.

```
c/c++
int a;
a=5
```

X Not possible in Python.

```
x=10
```

type(x) ⇒ shows type of variable.

str(x) : converts x to string
float(x) : converts x to float

⇒ How to take user input in python. : input function.

* input func^n is always takes a value as string

```
x = int(input("Enter the Number"))  // displays this message
```

26 Oct - Python

Operators in Python

+ - / * %. // **

x=10
y=5

x/y = 2.0
y/x = 0.5

here divide gives the float value (unlike in C++)
5/10 = 0

To get same output as C++. (i.e; get whole nums.)

x=5
y=10

x // y = 0 ← floor division Operator

x/y = 0.5

x %. y = 5

⇒ ** : exponential op.

x * y = Product

x **y = x^y

input:

```
x = int ( input (" Enter value of x " ) )
print (x+y)
```

in python; we can't declare a variable without initialization

No tornary operator here. (? :)

tornary in Python ⇒ [on-true] if [expression] else [on-false]

~~min = a if a.~~

```
min = a if a<b else b
```

Logical operator

are used to combine conditional statements.

and , or , not

Identity Operator → is / isnot

are used to compare objects.

Identity op. in python are used to determine whether a value is of a certain class or type.

'is' operator : returns true if both variables are of same type or class

'isnot' : returns true if " : are not " ".

(# type function int x = 5 type(x) → int)

if (type(x) is int) → gives true

if (type(x) is float) → gives false.

isnot

if (type(x) is not float) → true

Membership Operators: are used to check if a sequence is present in an object. → in / not in

(in) : Returns true if a sequence with the specified value is present in the object.

(not in) : Returns true " " " " " "

" is not present " " " ".

Q WAP in Python to calculate area of triangle
" " " " " " square
" " circle

Q swap values using 3rd variable
" without using 3rd variable

Q To accept electricity unit consumption and calculate total price @ 5/- per unit.
Give a discount of 10% on overall Bill

---

① b = int (input ("enter base"))

h = int (input ("enter height"))

print ("Area of triangle")

A = 1/2 * b * h

② s = int (input ("enter side of square"))

Area = s * s

print ("Area of square -", Area)

or

print ("Area of square -" + str (Area))

③ Single line swap

```
a, b = b, a
```

Python

Bitwise Operators are used to compare binary numbers;
Bitwise op are used to perform operations on bits.

& - Bitwise AND

Sets each bit to 1 if both bits are 1

| = Bitwise OR

Sets each bit to 1 if ~~both~~ any of bits are 1

^ = Bitwise EXOR

Sets each bit to 1 if only one of the two bits is 1

~ = Bitwise ~~XOR~~ NOT

<< = Bitwise left shift

>> = Bitwise Right Shift

Shift left by pushing ~~zeroes~~ zeroes from the right and let the leftmost bits fall

Shifts Right =

Example

14 << 1 ← Shift by 1 value.

8 Bit

1 1 1 0 1
8 4 2 1

8 4 2 1

1 1 1 0 << 1 = 1 1 1 0 0 = 28
16 8 4 2 1

17 << 1

0 0 0 1 0 0 0 1 << 1 = 1 0 0 0 1 0 = 34
32 16 8 4 2 1

Here 17 << 1 = 34

n << i is Equivalent to n * 2^i

Python Control statements

Conditional statements: Python statements are executed one by one in the same order as they appear in the program.

But, there will be various situations while writing a program, where u will have to take care of different possible conditions that might occur while execution of the program.

To Such Situations, if conditions can be used.

Selection Statement: If there is some condition, Based upon which it's decided that whether a statement will be executed or not. Then its called selection Statement & Python Provides us 'if' selection statement.

If statement: decision making; executes a block of code if condition given is true.

```
if condition :          or  if (condition) :
(Identation) # Body of if
```

```
if(a>b):
    print("a is large")
print("outside if")
```

Rendy If.

else statement:

If condition of if is false, then the optional else statement Runs.

```
if(a>b):
    print("a")
else:
    print("b")
```

Identation — python reads indentations (white space at the beginning of the lines) to define scope in the code.

(Other programming lang. often use curley Braces for the purpose)

```
age = int ( input ("Enter age") )
if ( age >= 18) :
    print (" eligible")
else :
    print (" not eligible")
```

elif statement

If any situation occurs in the program, when there is more than one condition then we use elif statement.

```
if ( - ):
    print ("Hey")
elif ( condition ) :
    print "hello")
else :
    print ("Bye")
```

Q WAP to find largest among 3 values. (a, b, c)

```
if ( a>b and a>c):
    print("a is largest")
elif ( b>a and b>c):
    print (" b is largest")
```

WAP to input marks of 5 subjects & calculate its total & % and after that display the grade.

* What are loops?

Allows us to execute a group of statements multiple times. There are certain condition defined in the beginning for entering the loop.
Once the condition is false, the loop will terminate & the control moves out of the loop.

NOTE: In Python prog; there are no Post-Tested loop available (like do-while in C++).

Adv: Reduce the size of the code. Reduce the complexity & make it more efficient; increase the execution speed.

Types of loops in Python

→ Two types: * While Loop : indefinite loop.
* For Loop :

In python: While loops are known as indefinite or conditional loops. They will keep iterating until certain condition is met. There is not guarantee ahead of time Regarding the number of iterations.

Syntax:

```
while expression :        # Condition
    # code to be executed
```

(++i, --i are not available in python)

```
i = 1
while ( i <=100):
    print(i)
    i = i+1
print ("done")
```

instead

```
i+=1
```

or

```
i=i+1
```

this indentation shows that these statements are a part of loop.

```
print ("value =", i)        ✓
print ("value =" + i)       // ERROR
```

we need to typecast explicitly

```
print ("value =" + str(i))
```

print ("hello", "world") // prints Hello ␣ World (␣ = space)

Python automatically adds a space b/w two comma separated values to be printed.

print ("hello" + "world") // concatenates to single string "helloworld", then prints it

Printing table of Number entered by User.

```
num = int ( input ("Enter number") )
i = 1
while ( i <= 10 )
    print (str(num) + " x " + str(i) + " = " + str(num * i))
    i += 1
```