**Programming in Python**

**Variables and Types:**

* A variable name must start with a letter or underscore character.
* A variable name can’t start with a number.
* A variable name contains: A-Z, a-z, 0-9 and \_

>>> a = "Hello, World!" # a is a variable of type string

>>> print(a)

Hello, World! # Try Here: <https://tinyurl.com/yvbsuapr>

>>> print(type(a))

<class ‘str’>

>>> b = 'Python' # b is a variable of type string

>>> print(b)

Python

>>> print(type(a))

<class ‘str’>

>>> x = 5 # x is a variable of type integer

>>> print(x)

5

>>> print(type(x))

<class ‘int’>

>>> y = 2.0 # y is a variable of type float

>>> print(y, type(y))

2.0 <class ‘float’>

**Input:**

# String Input

s = input("Enter String: ")

print("Your String: ", s)

Output:

Enter String: Python

Your String: Python

Try Here: <https://rb.gy/5yvbgu>

# Integer Input

a = int(input("Enter Integer: "))

print("Your Integer: ", a)

Output:

Enter Integer: 5

Your Integer: 5

# Float Input

b = float(input("Enter Float: "))

print("Your Float: ", b)

Output: Practice:

Enter Float: 5.0 <https://rb.gy/cd7ozp>

Your Float: 5.0 <https://rb.gy/3kmeyk>

**Comments:**

* # Single line comment.
* ‘‘‘ Multiple line comment. ’’’

**Type Casting:**

# Integers

x = int(1)

y = int(2.8)

z = int("3")

print(x,y,z)

Output:

1 2 3

# Floats

x = float(1)

y = float(2.8)

z = float("3")

w = float("4.2")

print(x,y,z,w)

Output:

1. 2.8 3.0 4.2

# Strings

x = str("s1")

y = str(2)

z = str(3.0)

print(x,y,z)

Output:

s1 2 3.0

**Escape Sequences:**

>>> print("Hello, World! ")

Hello, World!

>>> print("Hello,\tWorld!") # Hello World with tab

Hello, World!

>>> print("Hello,\nWorld!") # Hello World with new line

Hello,

World!

**Operators:**

1. **Arithmetic Operators:** ( +, -, \*, /, %, \*\*, // )

a = 5

b = 2

print(a+b) # 7

print(a-b) # 3

print(a\*b) # 10

print(a/b) # 2.5, float division

print(a%b) # 1, remainder of 5 divided by 2

print(a\*\*b) # 25, 5 to the power of 2

print(a//b) # 2, floor division (Integer)

**Practice:** <https://rb.gy/e1hsun> <https://rb.gy/352fly>

<https://tinyurl.com/v8st3ppu> <https://tinyurl.com/5n8cy6xn>

1. **Comparison Operators:** ( <, >, ==, !=, <=, >= )

a = 4

b = 7

print(a<b) # True

print(a>b) # False

print(a==b) # False

print(a!=b) # True

print(a<=b) # True

print(a>=b) # False

1. **Logical Operators:** ( and, or, not )



# Logical AND

print( True and True) # True

print( True and False) # False

print( False and True) # False

print( False and False) # False

# Logical OR

print( True or True) # True

print( True or False) # True

print( False or True) # True

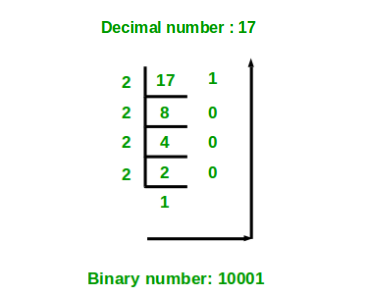
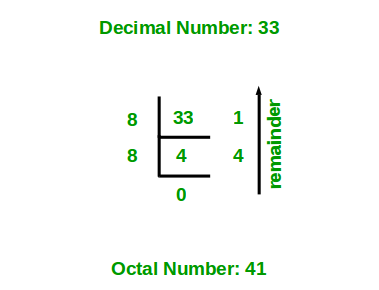
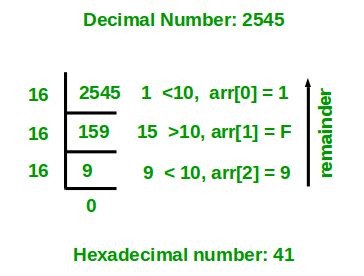
print( False or False) # False

# Logical NOT

print(not True) # False

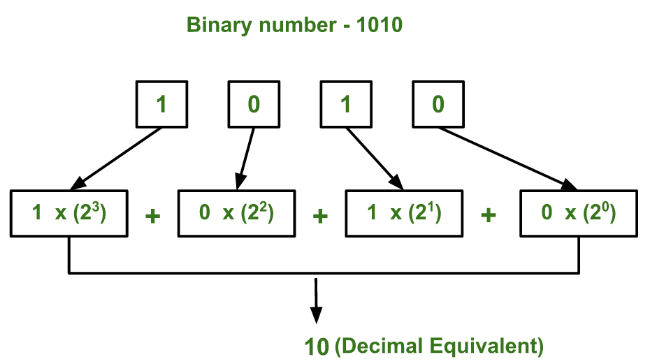
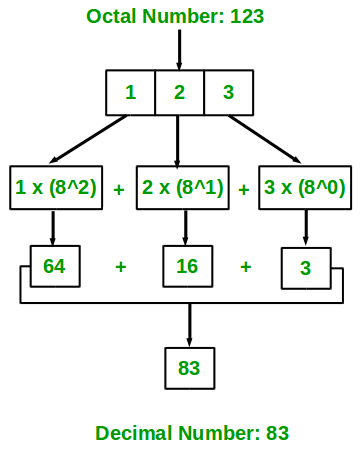
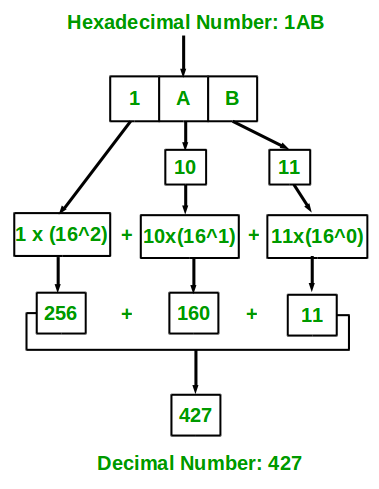
print(not False) #True

**Decimal to Binary:** **Decimal to Octal:** **Decimal to Hexadecimal:**



**Binary to Decimal:** **Octal to Decimal:** **Hexadecimal to Decimal:**

**Other Conversions:**

Binary to Octal/Hexa : Binary to Decimal to Octal/ Hexa

Octal to Hexadecimal : Octal to Decimal to Hexadecimal

1. **Bitwise Operators:** ( &, |, ~, ^, <<, >> )

# Bitwise AND

print(1 & 1) # 1

print(1 & 0) # 0

print(0 & 1) # 0

print(0 & 0) # 0

# 010 (2)

print(2 & 3) # 011 (3)

# 010 (2)

# Bitwise OR

print(1 | 1) # 1

print(1 | 0) # 1

print(0 | 1) # 1

print(0 | 0) # 0

# 010 (2)

print(2 | 3) # 011 (3)

# 011 (3)

# Bitwise NOT

n = 1

print(~n) # -2, -(n+1)

print(~0) # -1, -(0+1)

print(~(-n)) # 0, -((-n)+1)

# Bitwise XOR

print(1^1) # 0

print(1^0) # 1

print(0^1) # 1

print(0^0) # 0

# 010 (2)

print(2^3) # 011 (3)

# 001 (1)

# Bitwise Left Shift

print(5<<1) # 10, 5<<1 🡪 101+0 = 1010

print(5<<2) # 20, 5<<2 🡪101+00=10100

# Bitwise Right Shift

print(5>>1) # 2, 5>>1 🡪 101 = 10

print(5>>2) # 1, 5>>2 🡪 101 = 1

1. **Assignment Operators:** ( =, +=, -=, \*=, /=, %=, \*\*=, //=, &=, |=, ^=, <<=, >>=)

a = 5

b = 2

c = 3

d = a+b # d = 5+2

print(d) # 7

d += c # d = d+c

print(d) # 10

d -= a # d = d-a

print(d) # 5

d \*= b # d = d\*b

print(d) # 10

d /= a # d = d/a

print(d) # 2.0

d %= c # d = d%c

print(d) # 2.0

d \*\*= a # d = d\*\*a

print(d) # 32.0

d //= c # d = d//c

print(int(d)) # 10

d &= b # d = d&b

print(d) # 2

d |= a # d = d|a

print(d) # 7

d ^= b # d = d^b

print(d) # 5

d <<= 1 # d = d<<1

print(d) # 10

d >>=1 # d = d>>1

print(d) # 5

1. **Identity Operators:** ( is, is not )

print(10 is not 5) # True

print(10 is 5) # False

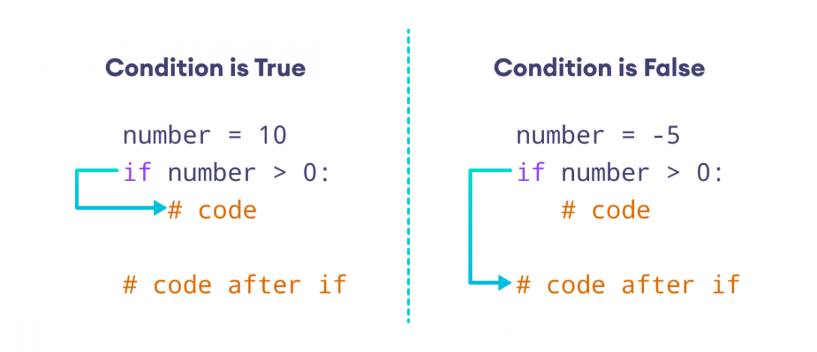
**Precedence and Associativity:**

a = int((10 \* 2 - 3/2 + 1)//3)\*\*2 ^ (5<<1 & 5>>1) % ( 4 | -(~2)) #38

**Order of Evaluation:**

(), \*\*, ~, \*, /, //, %, +, -, <<, >>, &, ^, | # Left to Right

**Control Statements:**

1. **Conditional Statements:**
2. if statement:

Syntax:

if (condition):

statements

next statements

Example:

a = 5

b = 3

if(a > b): # True

print("a is greater than b") # Output:

print("a is",a) a is greater than b

print("b is",b) a is 5

b is 3

Example:

a = 5

b = 3

if(a < b and a>=b): # False

print("a is greater than b") # Output:

print("a is",a) Logical and

print("Logical and")

Example:

if(3 < 5 or 3 == 5): # True

print(" 5 is greater than 3") # Output:

print("Logical or") 5 is greater than 3

Logical or

**Indentation:**

Error Cases:

if (not 3 > 5):

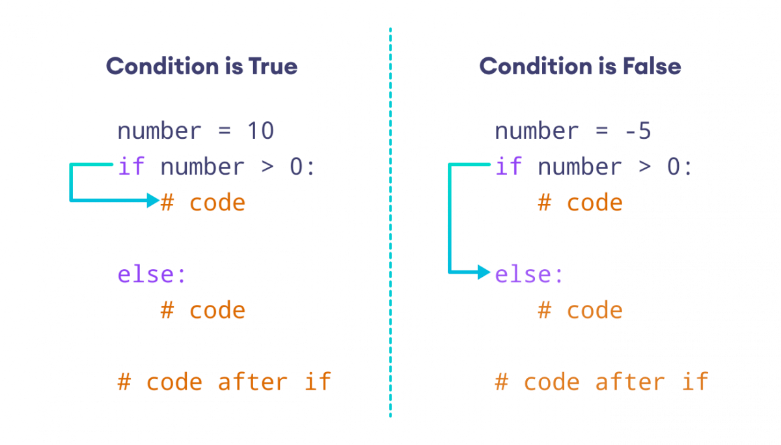
print("3 is less than 5") # space required

if(3 < 5):

print("5 is greater than 3") # space required

print("a is 3")

print("b is 5")

1. else statement:

Syntax:

if (condition):

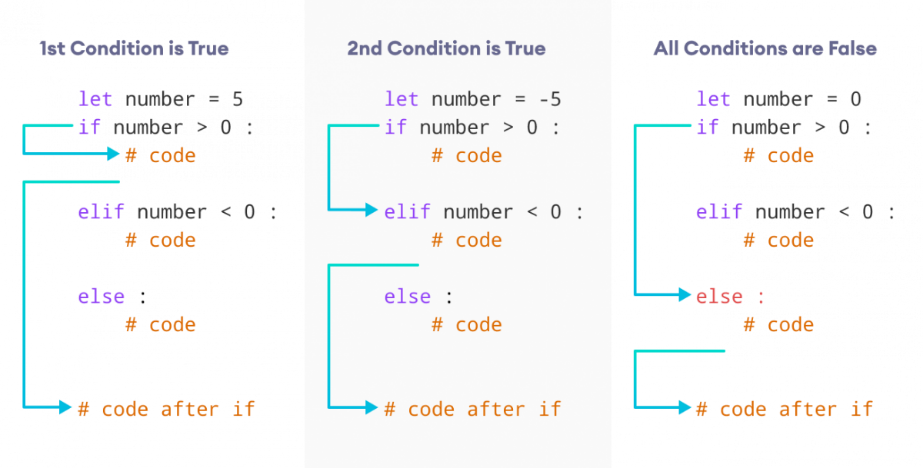
statements

else :

statements

next statements

1. elif statement:



Syntax:

if (condition):

statements

elif (condition):

statements

else:

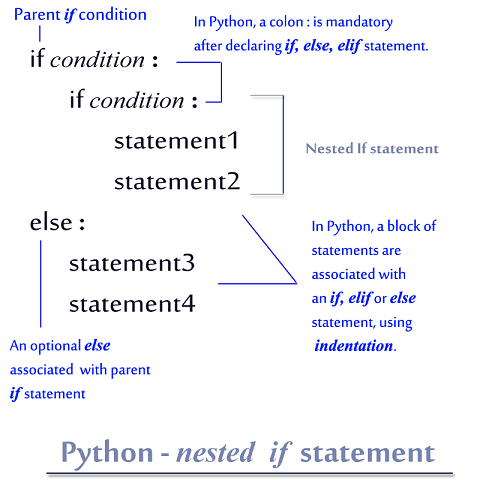
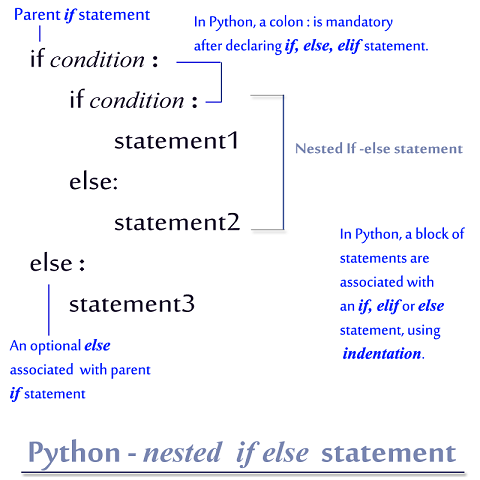
statements

next statements

Practice: <https://rb.gy/a29di6>

<https://tinyurl.com/225me378>

1. Nested Statements:



Example:

if (3 < 5): # True

if (3 > 5): # False

print("Yes")

elif (3 != 5) : # True

print("No") # output: No, go to next statements

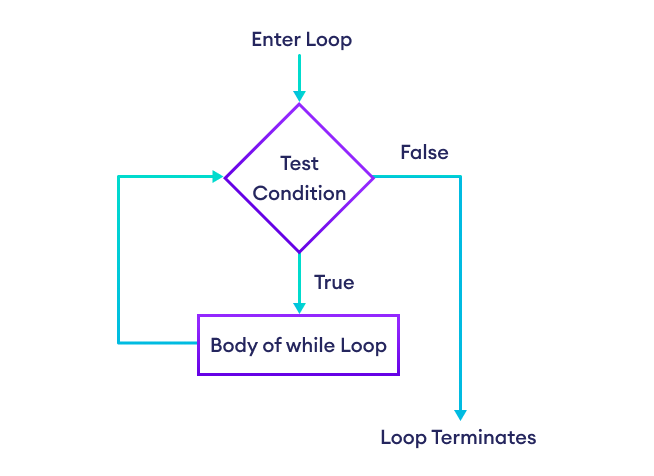
else :

print("else 1")

else :

print("else 2")

print("Here") # Here

1. **Iterative Statements:** (Loops)
2. while loop:

syntax:

while (condition):

statements

next statements

Examples:

<https://rb.gy/mxemmn>

<https://rb.gy/tgakxs>

<https://rb.gy/mz0boi>

<https://rb.gy/erei5i> Practice: <https://tinyurl.com/y2b4azpp>

1. for loop: using range()

Syntax: range(start, stop, step)

Parameter:

* start: [optional] start value of the sequence (int)
* stop**:**next value after the end value of the sequence (int)
* step**:**[optional]integer value, denoting the difference between any two numbers in the sequence. (int)

Examples:

1. range(stop): <https://rb.gy/6raaxd>
2. range(start, stop): <https://rb.gy/jriff1>
3. range(start,stop,step): <https://rb.gy/zxwny0> , <https://rb.gy/ygwqyt>

Practice: <https://tinyurl.com/4ndafn3n>

**Nested Loops:**

while:

<https://tinyurl.com/2w749dnd>

<https://tinyurl.com/bdcr265j>

<https://tinyurl.com/2p8d3fcd>

<https://tinyurl.com/t946kjkf>

for loop:

<https://tinyurl.com/5bd5n24t>

<https://tinyurl.com/574b97r9>

<https://tinyurl.com/56jp8vzm>

<https://tinyurl.com/5n6s8se3>

**Functions:** <https://www.interviewbit.com/problems/functions/>

<https://tinyurl.com/2p9xe6s9>

**DSA:** <https://www.interviewbit.com/courses/programming/>

**Important Programs:**

1. Swapping of two numbers using third variable and without using third variable.
2. Sum of the digits of a number.
3. Reverse the number.
4. Palindrome number.
5. Fibonacci series.
6. Factorial of a number and given number is factorial or not.
7. Given number is prime number or not.
8. LCM and GCD.
9. Square root / Cube root of given number.
10. Patterns