**Day-01**

**20-02-2025**

**===============**

**Input Statement:**

**============**

-> When we want to provide the input to the computer during the execution of the program, input statement can be used.

-> input statement in python can be defined with "input() function".

**input():**

**=====**

Syntax:

identifier(variable) = input('Text')

Note:

====

-> input() can allowed to read any value but in the string format only.

**How we can take the values other than the string format using input()?**

========================================

-> To take any value other than string format, we can perform type conversion to the input()

-> Type Conversion: converting the data from one format to another format is called as "Type conversion" or "Type casting".

-> Type casting/Type conversion can be defined in two ways:

1) Implicit Typecasting/Automatic Typecasting

2) Explicit Typecasting

**1) Implicit Typecasting/Automatic Typecasting**

**===============================**

-> The python can perform the typecasting according to the operation automatically is called as "implicit typecasting".

a = 10

b = 3

print(type(a))

print(type(b))

b = a/b

print(type(a))

print(type(b))

**2) Explicit Typecasting**

**===============**

-> Programmer can define the typecasting as per need is called as "Explicit Typecasting".

-> For explicit typecasting, we have some pre-defined functions in python:

1) int() ==> can use to convert the data into integer

2) float() ==> any data into float

3) complex() ==> any data into complex

4) bool() ==> any data into boolean

5) str() ==> any data into string

**int():**

**===**

Syntax:

int(data/variable)

1) any integer value (binary/octal/hexadecimal) can convert into decimal.

2) any float value can convert into decimal.

3) Complex value cannot convert into decimal.

If we can apply the complex value to the int() can give "type error".

4) Boolean value can convert into decimal.

5) Not all the strings can allow for conversion into an integer.

If a string with only decimal can possible for the conversion into an integer.

a = 0b10101

b = 0o123

c = 0xaf

d = 102.202

e = 102e-45 # 0.0000000......0102

# f = 12.23-12j

f = True

g = False

h = "123"

i = "0b11001"

print(int(a))

print(int(b))

print(int(c))

print(int(d))

print(int(e))

print(int(f))

print(int(g))

print(type(h))

print(type(int(h)))

# print(int(i))==> Value error

**float()**

**====**

Syntax:

float(value/variable)

1) any integer value allow to convert into float.

2) complex value not allowed for the conversion into float.

3) Boolean value can convert into float.

4) When a string with decimal or float can convert into float.

a = 123

b = 0b101

c = 0o121

d = 0xaf

# e = 12-23j

e = True

f = False

g = '123'

h = '123.234'

print(float(a))

print(float(b))

print(float(c))

print(float(d))

print(float(e))

print(float(f))

print(type(float(g)))

print(type(float(h)))

**complex()**

**=======**

Syntax:

1) with single parameter

complex(value/variable)

2) with two parameters

complex(val1/var1, val2/var2)

1) Any integer can convert into complex by adding 0j as an imaginary to the given value.

the given value can understood as "real".

2) Any float can convert into complex

3) Boolean value can convert into float.

4) When a string with decimal/float/complex can convert into complex.

a = 123

b = 0b11001

c = 0o123

d = 0xaf12

e = 12.234

f = True

g = "123-321j"

h = '12.234'

print(complex(a))

print(complex(b))

print(complex(c))

print(complex(d))

print(complex(e))

print(complex(f))

print(type(complex(g)))

print(complex(h))

**Day-02**

**21-02-2025**

**===============**

**bool()**

**======**

-> bool() can be used to convert any type of value into boolean.

Note:

=====

bool() can return "True" for any non-zero value.

bool() can return "False" for zero value.

Syntax:

bool(data/variable)

a = 123

b = -123

c = 0b11001

d = 0o123

e = 0x1af2

f = 1-2j

g = 1.2e-5

h = 'Python'

i = 0

j = 0.0

k = 0-0j

l = ''

print(bool(a))

print(bool(b))

print(bool(c))

print(bool(d))

print(bool(e))

print(bool(f))

print(bool(g))

print(bool(h))

print(bool(i))

print(bool(j))

print(bool(k))

print(bool(l))

**str()**

**====**

-> str() can be used to convert any value into string.

Syntax:

str(value/variable)

a = 112

b = 0b1101

print(type(a))

print(type(b))

print(type(str(a)))

print(type(str(b)))

print(type(str(123.23e-5)))

print(type(str(123-23.4j)))

print(type(str(True)))

**# WAP IN PYTHON TO ACEPT INTEGER, FLOAT, COMPLEX, BOOLEAN AND STRING AS AN INPUT AND PRINT THOSE.**

# to take integer using input():

# int(input())

a = int(input("Enter some value:"))

b = float(input("Enter some value:"))

c = complex(input("Enter some value:"))

d = bool(input("Enter some value:"))

e = str(input("Enter some value:"))

f = input("Enter some value:")

print(type(a))

print(type(b))

print(type(c))

print(type(d))

print(type(e))

print(type(f))

**Q: WAP TO TAKE BINARY, OCTAL, HEXADECIMAL AND DECIMAL AS AN INPUT. AND PRINT THOSE VALUES IN THE CORRESPONDING BASE ONLY.**

Syntax:

for taking binary:

int(input(),2)

for taking octal:

int(input(),8)

for taking hexadecimal:

int(input(),16)

a = int(input("Enter a binary value:"),2)

b = int(input("Enter an octal value:"),8)

c = int(input("Enter an hexadecimal value:"),16)

d = int(input("Enter a decimal value:"))

print(a,bin(a))

print(b,oct(b))

print(c,hex(c))

print(d)

**Base Conversion Functions:**

**==================**

-> three base conversion functions:

1) bin()

2) oct()

3) hex()

**1) bin()**

**======**

-> bin() can convert any base value into base-2 value (binary)

Syntax:

bin(value/variable)

**2) oct()**

**======**

-> oct() can convert any base value into an octal (base-8)

Syntax:

oct(value/variable)

**3) hex()**

**======**

-> hex() can convert any base value into hexadecimal (base-16)

Syntax:

hex(value/variable)

a = 123

b = 0b10101

c = 0o1321

d = 0xaf123fa

print(bin(a))

print(bin(b))

print(bin(c))

print(bin(d))

print(oct(a))

print(oct(b))

print(oct(c))

print(oct(d))

print(hex(a))

print(hex(b))

print(hex(c))

print(hex(d))