Data Types:

1. Numeric Types
2. Variables
3. Literals
4. Conversion

What is program:

* Set of instructions which understood by computer

Length=15

Breadth=5

Area=length \* breadth

Print(“area is “,area)

Calculate bill amount:

Prices=[10,7,5,8,6]

Quantity=[1,2,3,1,1]

10+14+15+8+6

Total=0

For I in range(len(Prices):

Total=total+ (prices[i] \* Quantity[i]

Print(total)

Progam:

A, Data

B. Instructions

Operations performed on the data 🡪 Instructions

Class 3:

Variables:

1. What are the variables?
2. Declaration
3. Initialization
4. Memory allocation
5. Types of data
6. Multiple declaration

Variables: Names give to data

A blue screen with white text and words

AI-generated content may be incorrect.

Declaration & Initialization

* Length=15

length=5

print(length) # 5

Memory:

Referring to data

A blue screen with white squares and black text

AI-generated content may be incorrect.

Types of Data:

* Integer: 15
* Price: 12.75
* Name: “John” (string)

prices=[10,7,5,8,6]

print(prices) #[10, 7, 5, 8, 6]

Multiple declarations:

* Name, price,qty=”soap”,12.75,5

A screenshot of a computer

AI-generated content may be incorrect.

name,price,qty="Soap",12.75,5

print(name,price,qty)

# Soap 12.75 5

c. a,b,c=1,1,1

a=b=c=1

A screenshot of a computer

AI-generated content may be incorrect.

a=b=c=1

print(a,b,c)   # 1,1,1

print(id(a),id(b),id(c)) #140723107910688 140723107910688 140723107910688

Python dynamically typed:

c/c++/Java -🡪 statically typed language

int a; or int a=15;

float b=12.75;

char c=”a”;

Dynamically typed language:

* Python , Java script

a=15

b=12.75

c=”John”

ii.

a=12.75 (float)

a=15 (int)

a=”Chalo” (str)

A screenshot of a computer

AI-generated content may be incorrect.

Rules for variables names:

* It should be meaningful

Prod\_id =153

Price=23.15

Cust\_name=”Smith”

Cust\_city=”New York”

* Alphanumeric and Underscore

A1=10

Cust\_name=”James”

Cust-name=”James”

Cust name=”James”

\_cust\_name=”james”

1a=10

None=”Delhi” (should not be a keyword)

If=10

False=”Yes”

false=”yes”

* (predefined words/Reserve words)
* Starts with letter or underscore
* Not a keyword

A blue screen with white text

AI-generated content may be incorrect.

* Case sensitive

a. Price=19.99

b. price=29.99

c. false=”yes”

PYTHON Data Types:

A blue screen with text

AI-generated content may be incorrect.

Numeric: Single Valued

Int : x=5

Float y=25.5

Complex x=3+4j

Bool term=False/True

Sequence:

List : lst1=[2,4,6,8,10] -🡪 Mutable [2,7,6,8,10]

Ordered, Indexed, Mutable

Tuple: tpl1=(1,3,5,7) 🡪 Not mutable 🡪 (1,3,5,7) 🡪 Immutable

Ordered, Indexed, Immutable

String” s🡪 “Python” 🡪 Immutable

Ordered,indexed,Immutable

Set:

Set1={2,4,6,8,10} 🡪 Collection of values🡪 No index, Immutable, No dups (unique)

Unordered, unindexed, Immutable, No dups

Dict:

Key:value pair

Collection of key,values pair

Keys are unique

D={name:”john”,roll:126,dept:”cse”}

A screenshot of a computer

AI-generated content may be incorrect.

Ordered, Mutable(changeable),key and value pair

Numeric Datatypes:

Int

Float

Complex

Bool

1. Declaration and Initialization
2. Positive and negative number
3. Memory size -- Independent
4. Data size (unlimited) 🡪 Infinite
5. Immutable (x=25,x=50) 🡪 25 was immutable

Declaration and Initialization

X=101

Y=-15 (positive and negative)

Memory size

X=101 (28 bytes)

Y=123455678999999) 32 bytes

X=101 (Garbage collector)

X=205

Python does not guarantee length (immutable)

INT:

x=101

y=12345678910101010101010110011010

print(y) #12345678910101010101010110011010

print(sys.getsizeof(x)) #28

print(sys.getsizeof(y)) # 40

x=101

print(id(x)) #140723183935648

x=205

print(id(y)) # 2247921997744

Float

1. 29.75/0.075
2. Declaration and initialization 🡪Floating point representation

A=12500.0

125 \* 100

125\*10\*\*2

125E2 (Floating point) 🡪 Power

Mantissa/Exponent

b.23.45

2345/100

2345/10\*\*-2

2345 E -2

1. +ve/ve

A= 29.75 / -1.75 / 2.5E2/-3.1E-2

a=29.75

b=-3.75

c=-2.5E2

d=-3.1E-2

print(a,b,c,d)

# 29.75 -3.75 -250.0 -0.031

print(sys.getsizeof(a)) #24

print(sys.getsizeof(b)) #24

print(sys.getsizeof(c)) #24

print(sys.getsizeof(d)) #24

Numeric Datatype (Bool & Complex)

1. Declaration & Initialization

X=True --- 1

Y=False --0

Z=false

1. Boolean is used in conditions

A=10

B=5

a>b 🡪 True

if a>b:🡪 True

x=True

y=False

print(x,y) # True,False

print(int(x),int(y)) #1,0

# z=true

# print(z)

#NameError: name 'true' is not defined

print(type(x),type(y))

#<class 'bool'> <class 'bool'>

a=10

b=5

print(a>b) # True

Complex:

5+I3 🡪 5+sqrt (-9)

5+ -1 \* sqrt(9)

5+3J (i🡪 Imaginary)

C2=2.5+1.5j

C3=complex(2.5,1.5)

* (+,-,\*,/)

c=10+5j

print(c,type(c))

# 10+5j --> complex

c=complex(-5,-3.1)

print(c)

#(-5-3.1j)

LITERALS/CONSTANTS:

What are Literals:

1. Integer
2. Float
3. Boolean
4. Complex
5. String

A=10 (Direct data)-- Constats

B=5 (Direct data) 🡪 Constants

C=A+B (Indirect data)

D=input(“Enter name”) (Indirect data)

Constants are literals

1. Integer Literals

A=201

B=135\_246\_198 / 13\_52\_46\_198

In between digits

C=358\_

D=\_124

Prog1:

a=201

b=252\_456\_789

print(b) # 252456789

# c=358\_

# print(c) #SyntaxError: invalid token

d=\_789

print(d)

#NameError: name '\_789' is not defined

1. Float literals

C=12E2

D=12.5e-2

E=12\_5.67/12\_5.6\_7

F=5\_.\_9

Prog:

c=12E2

print(c)

#1200.0

D=12.5e-2

E=12\_5.6\_7

print(D,E)

#0.125

#125,67

# F=5\_.\_9

# print(F)

# SyntaxError: invalid token

1. Bool literals

a=True

b=False

print(a,b) #True/#False

1. Complex literals

a=4+5j

b=1\_2+1\_4j

c=1.4+2.5j

print(b)

#(12+14j)

1. String literals

Name=’alexa’

Name1=”alexa”

Name2=”””

Hello how are you

Yes

“””

Prog:

name1='Alexa'

name2="John"

name3="""JOHN1"""

print(name1) #Alexa

print(name2) #"John"

print(name3) #JOHN1"

INTEGER LITERALS:

1. Decimal: 0-9
2. Binary: 0,1

17🡪 10001 (Decimal to Binary)

1. Octal: 0-7

17🡪 21

1. Hexa Decimal: 0-9,a-f

17🡪 11

A blue rectangular table with white text

AI-generated content may be incorrect.

Decimal: 10

Binary: a=0b1010

Octal: 0o12

Hexa: 0xA

C1=0b1010+15j (no literals in imaginary part)

Prog 1:

a=10

print(a)

#10

b=0b1010

print(b)

#10

c=0o12

print(c)

#10

d=0xA

print(d)

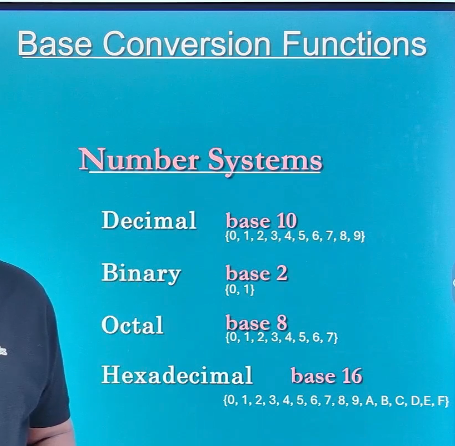
#10

C1=0b1010+15j

print(C1)

# (10+15j)

BASE CONVERSION:



bas (int) 🡪 Binary

oct(int)🡪 Octal

hex(int) 🡪 Hexadecimal

int(str) 🡪 Decimal

CODE:

# Base conversion functions

print(bin(10))

#0b1010

print(type(bin(10)))

#<class 'str'>

print(oct(10))

#0o12

print(type(oct(10)))

#<class 'str'>

print(hex(10))

#0xa

print(type(hex(10)))

#<class 'str'>

print(bin(15))

#0b1111

print(oct(15))

# 0o17

print(hex(15))

#0xf

print(bin(True))

#0b1

print(bin(1.0))

#TypeError: 'float' object cannot be interpreted as an integer

TYPE CONVERSION:

* Converting one datatype to one more datatype.

Int()

Float()

Bool()

Complex()

Str()

Int:

Integer to complex will give error

Integer to valid string (other than number “125”) will gives error

A screen shot of a computer

AI-generated content may be incorrect.

Prog:

print(int(16.59))

print(type(int(16.59)))

#<class 'int'>

#16

print(int(True))

#1

print(int("125"))

#125

print(int('0b1010',2))

#10

print(int("0xA",16))

#10

# print(int("Alexa"))

#ValueError: invalid literal for int() with base 10: 'Alexa'

# print(int(3+1j))

#TypeError: can't convert complex to int

Float

A screenshot of a computer

AI-generated content may be incorrect.

print(float(125))

#125.0

print(float(True))

#1.0

print(float("12.75"))

#12.75

Bool:

Bool(anything) 🡪 True

bool(False)🡪 False

bool(0)

bool()

A screen shot of a computer

AI-generated content may be incorrect.

print(bool(10))

#True

print(bool(-12))

#True

print(bool(-1.2E-3))

#True

print(bool(3+4j))

#True

print(bool("False"))

#True

print(bool())

#False

print(bool(0))

#False

print(bool(False))

#False

Complex:

A screenshot of a computer

AI-generated content may be incorrect.

print(complex(10))

#(10+0j)

print(complex(-12.5))

#(-12.5+0j)

print(complex(True))

#(1+0j)

print(complex(False))

#0J

print(complex("3+4j"))

#3+4j

STR:

A screen shot of a computer

AI-generated content may be incorrect.

print(str(10))

#10

print(str(-12))

#-12

print(str(-1.2E-3))

# -0.0012

print(str(False))

#False

print(str(3+4j))

#3+4j