### Ternary operator

condition = True

# traditional approach

if condition:

    x = 1

else:

    x = 0

print(x)

# ternary operator

condition = False

x = 1 if condition else 0

print(x)

### Underscore placeholders

# separating every thousand using 'underscore'

num1 = 10\_000\_000\_000

num2 = 10\_000\_000\_000

total = num1 + num2

print(f'{total:,}')

### Using Enumertor

# using enumerate()

names = ['John', 'Doe', 'Eric']

for index, name in enumerate(names, start=1):

    print(f"{index} - {name}")

# another example

names = ['Peter Parker', 'Clark Kent', 'Wade Wilson', 'Bruce Wayne']

heroes = ['Spiderman', 'Superman', 'Deadpool', 'Batman']

for index, name in enumerate(names):

    hero = heroes[index]

    print(f'{name} is actually {hero}.')

### Using Zip()

names = ['Peter Parker', 'Clark Kent', 'Wade Wilson', 'Bruce Wayne']

heroes = ['Spiderman', 'Superman', 'Deadpool', 'Batman']

for name, hero in zip(names, heroes):

    print(f"{name} is actually {hero}.")

# Peter Parker is actually Spiderman.

# Clark Kent is actually Superman.

# Wade Wilson is actually Deadpool.

# Bruce Wayne is actually Batman.

# second example

names = ['Peter Parker', 'Clark Kent', 'Wade Wilson', 'Bruce Wayne']

heroes = ['Spiderman', 'Superman', 'Deadpool', 'Batman']

universes = ['DC', 'Marvel', 'IStudio', 'Marvel']

for name, hero, universe in zip(names, heroes, universes):

    print(f"{name} is actually {hero} from {universe}.")

# Peter Parker is actually Spiderman from DC.

# Clark Kent is actually Superman from Marvel.

# Wade Wilson is actually Deadpool from IStudio.

# Bruce Wayne is actually Batman from Marvel.

# third example

for value in zip(names, heroes, universes):

    print(value)

# ('Peter Parker', 'Spiderman', 'DC')

# ('Clark Kent', 'Superman', 'Marvel')

# ('Wade Wilson', 'Deadpool', 'IStudio')

# ('Bruce Wayne', 'Batman', 'Marvel')

### unpacking

a, b = (2, 3)

print(a)  # 2

print(b)  # 3

# \_ means we don't want to use this var

# \_ is helpful since some IDEs are checking for unused var

a, \_ = (10, 20)

# print(a)  # 10

# asterisk (\*) hold all other values of the tuple

a, b, \*c, d = (1, 2, 3, 4, 5, 6)

print(a)   # 1

print(b)   # 2

print(\*c)  # 3 4 5

print(d)   # 6

### getpass()

from getpass import getpass

firstname = input('Username: ')

password = getpass('Password: ')

print('Logging in ...')

### setattr, gettr

class Person:

    pass

# 1

p = Person()

print('Before modification:', p.name)

# setting 'name' to 'John'

setattr(p, 'name', 'John')

print('After modification:', p.name)

# 2

person = Person()

person.firstName = 'John'

person.lastName = 'Smith'

print(person.firstName)  # John

print(person.lastName)  # Smith

# 3 - getattr, setattr

first\_key = 'firstName'

first\_val = 'Adriena Garciela'

setattr(person, first\_key, first\_val)

print(person.firstName)  # Adriena Garciela

res = getattr(person, first\_key)

print(res)  # Adriena Garciela

# 4 - another example

person\_info = {'first': 'Maria', 'last': 'Payenda'}

for key, value in person\_info.items():

    setattr(person, key, value)

for key in person\_info.keys():

    res = getattr(person, key)

    print(res)

    # Basir

    # Payenda