CHƯƠNG 4 – ĐỊNH NGHĨA HÀM (FUNCTIONS)

1. KHÁI NIỆM HÀM

- Hàm là một khối mã được lưu trữ và tái sử dụng.
- Có 2 loại hàm trong Python:
  + Hàm dựng sẵn (built-in): print(), input(), type(), float(), int(), max(), min()…
  + Hàm do người dùng định nghĩa.

2. ÉP KIỂU (TYPE CONVERSION)

Số ↔ Số

```
print(float(99) / 100)# → 0.99
```

Chuỗi ↔ Số

```
s = '123'
i = int(s)
print(i + 1) # → 124
```

Lỗi:

```
int("hello")# ValueError
"123" + 1# TypeError
```

3. ĐỊNH NGHĨA HÀM (FUNCTION DEFINITION)

Cú pháp:

```
def function_name(parameters):
  # code
```

Ví dụ:

```
def thing():
print("Hello")
print("Fun")

thing()
print("Zip")
thing()
```

Kết quả:

```
Hello
Fun
Zip
Hello
Fun
```

4. HÀM CÓ THAM SỐ (PARAMETERS & ARGUMENTS)

Tham số = biến trong định nghĩa hàm

Đối số = giá trị truyền vào khi gọi

Ví dụ:

```
def greet(lang):
  if lang == 'es':
      print('Hola')
  elif lang == 'fr':
print('Bonjour')
  else:
      print('Hello')

greet('fr')  # → Bonjour
```

5. HÀM CÓ GIÁ TRỊ TRẢ VỀ (RETURN VALUES)

Ví dụ 1:

```
def greet():
  return "Hello"

print(greet(), "Glenn")
```

Ví dụ 2:

```
def greet(lang):
  if lang == 'es':
      return 'Hola'
  elif lang == 'fr':
      return 'Bonjour'
  return 'Hello'

print(greet('es'), "Sally")
```

6. HÀM NHIỀU THAM SỐ

```
def addtwo(a, b):
  return a + b

x = addtwo(3, 5)
print(x)     # → 8
```

7. HÀM KHÔNG TRẢ VỀ (VOID FUNCTION)

Hàm không dùng return, chỉ thực hiện tác vụ.

Ví dụ:

```
def print_lyrics():
  print("I'm a lumberjack, and I'm okay.")
  print("I sleep all night and I work all day.")
```

8. LÝ DO DÙNG HÀM

- Giúp tổ chức code thành các “đoạn” logic.
- Giảm lặp code (“Don’t repeat yourself”).
- Tái sử dụng nhiều lần.
- Dễ sửa, dễ mở rộng, dễ chia sẻ.