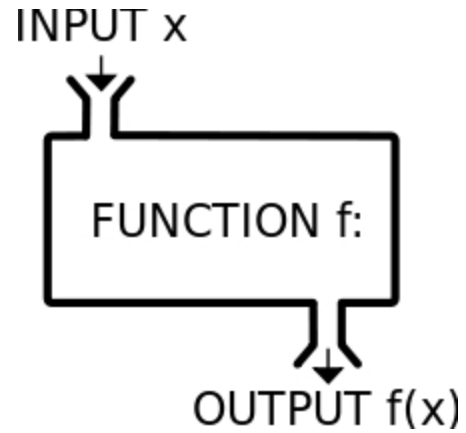


# TP - Programmation fonctionnelle



## Cas pratique

par **Fabien Barbaud** - [@BarbaudFabien](#)

# Environnement de développement

```
$ git clone https://github.com/fabienbarbaud/docker-python.git  
$ cd docker-python  
$ docker-compose run --rm python --version
```

```
Creating docker-python_python_run ... done  
Python 3.9.1
```

# Exercice 1

Convertir en programmation fonctionnelle - *reduce*

```
numbers = [1, 2, 3, 4]
total = 0
for number in numbers:
    total += number
print(total)
```

# Réponse

```
import functools
numbers = [1, 2, 3, 4]
functools.reduce(lambda x, y: x + y, numbers)
```

```
((1+2)+3)+4)
```

## Exercice 2

Convertir en programmation fonctionnelle - *map*

```
items = [1, 2, 3, 4, 5]
squared = []
for x in items:
    squared.append(x ** 2)
print(squared)
```

# Réponse

```
items = [1, 2, 3, 4, 5]  
list(map(lambda x: x ** 2, items))
```

## Exercice 3

Convertir en programmation fonctionnelle - *filter*

```
result = []  
for x in range(-5, 5):  
    if x < 0:  
        result.append(x)  
print(result)
```

# Réponse

```
list(filter(lambda x: x < 0, range(-5,5)))
```



## Exercice 4

Convertir en programmation fonctionnelle - *reduce*

```
L = ['Chef ', 'de ', 'projet ', 'digital']  
''.join(L)
```

```
'Chef de projet digital'
```

# Réponse

```
import functools
L = ['Chef ', 'de ', 'projet ', 'digital']
functools.reduce( (lambda x,y:x+y), L)
```

## Exercice 5

Faire la somme du tableau suivant

```
chiffres = ['un', 'deux', 'trois', 'quatre']
```

# Réponse

```
import functools
chiffres = ['un', 'deux', 'trois', 'quatre']

def conversion(chiffre):
    if chiffre == 'un':
        return 1
    elif chiffre == 'deux':
        return 2
    elif chiffre == 'trois':
        return 3
    elif chiffre == 'quatre':
        return 4

vraiChiffres = map(conversion, chiffres)
functools.reduce(lambda x,y:x+y, vraiChiffres)
```

# Réponse (bis)

```
import functools
chiffres = ['un', 'deux', 'trois', 'quatre']

def conversion(chiffre):
    dico = {"un": 1, "deux": 2, "trois": 3, "quatre": 4}
    return dico[chiffre]

vraiChiffres = map(conversion, chiffres)
functools.reduce(lambda x,y:x+y, vraiChiffres)
```

## Exercice 6

Trouver le nombre de fois que la lettre **e** est présente dans cette phrase :

```
phrase = 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.'
```

# Réponse

```
import functools
phrase = 'Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.'

def compteParLettre(lettre):
    return (lettre, 1)

def filtreLesE(compteParLettre):
    lettre, compte = compteParLettre
    return lettre == "e"

def nombreDeLettres(nombreTotal, compteParLettre):
    lettre, compte = compteParLettre
    lettreTotal, compteTotal = nombreTotal
    return (lettreTotal, compteTotal + compte)

print(
    functools.reduce(
        nombreDeLettres,
        filter(
            filtreLesE,
            map(
                compteParLettre,
                list(phrase)
            )
        )
    )
)
```