



## TP 2 - Ajout décorateurs

### 1 - Décorateur

En utilisant un décorateur, décorer votre fonction à optimiser de la fonction de traduction d'input

```
@input_traduction
def ma_fonction(x):
    return x
```

### 2 - Définir les bornes

Dans le décorateur de la fonction à optimiser, ajouter deux options:

- `higher = 10`, borne maxi de la fonction mathématique
- `lower = -10`, borne mini de la fonction mathématique

Appliquer ces variables à la traduction de génome en input

```
input = (x * (higher - lower)) + lower
```

### 3 - Paramètres supplémentaires

1. Permettre à la fonction `Cell.__init__` d'avoir un kwarg `**fct_cfg`

Stocker ces données sous un attribut `self.fct_cfg`.

Celles-ci seront utilisées pour permettre d'avoir des fonctions à optimiser qui nécessitent des paramètres supplémentaires (ex: Ackley)

Adapter la fonction `child`, on doit passer à la nouvelle cellule créée les paramètres

`self.fct_cfg`

2. Ajouter un kwarg `**fct_cfg` à la fonction décorée

Aux endroits où cette fonction est appelée, passer en kwarg les données stockées dans

`self.fct_cfg`

Le décorateur devra passer ces paramètres en kwargs de la fonction mathématique

### 4 - Implémenter une autre fonction

1. Choisir une autre fonction du site <https://benchmarkfcns.info/fcns>, ayant des paramètres supplémentaires (ex: Ackley)

L'implémenter dans le code, et appliquer le décorateur sur la fonction

2. Ajoutez les deux fonctions mathématiques à un `dict` global, avec leur nom pour clé

Ajoutez en paramètre de la fonction `Cell.__init__` le *nom* de la fonction à appeler

Récupérer depuis le dict global et le nom de la fonction, l'objet fonction à utiliser, et la stocker dans un attribut `self.f`.

Adapter votre code en conséquence

### 5 - Fonction main

Stockez la boucle principale dans une fonction `main`

Remplacez toutes les constantes utilisées précédemment ( `NB_CELLS` par ex) par des kwargs passés à la fonction

Ajoutez en plus de ces arguments des kwargs supplémentaires `**fct_cfg`, que vous passez à la construction des `Cell`