Manipulation d’une liste ordonnée

Exercice pratique

Situation à résoudre

Problème

Pour une collection ordonnée en mémoire, on veut inverser la deuxième moitié de la liste avec la première moitié.

Ex :

"Fred"	"Adam"	"Simon"	"Louis"	"Marc"	"Denis"	"Roger"	"Luc"

deviendra

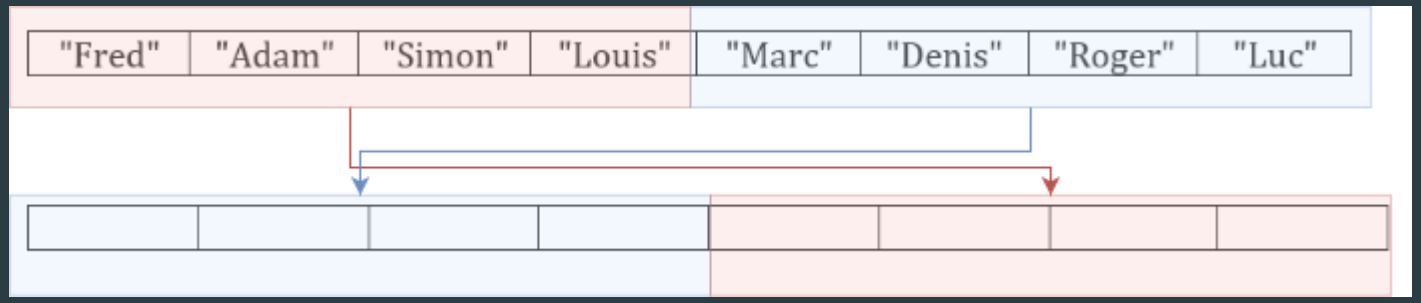
"Marc"	"Denis"	"Roger"	"Luc"	"Fred"	"Adam"	"Simon"	"Louis"

Des idées pour résoudre la situation?

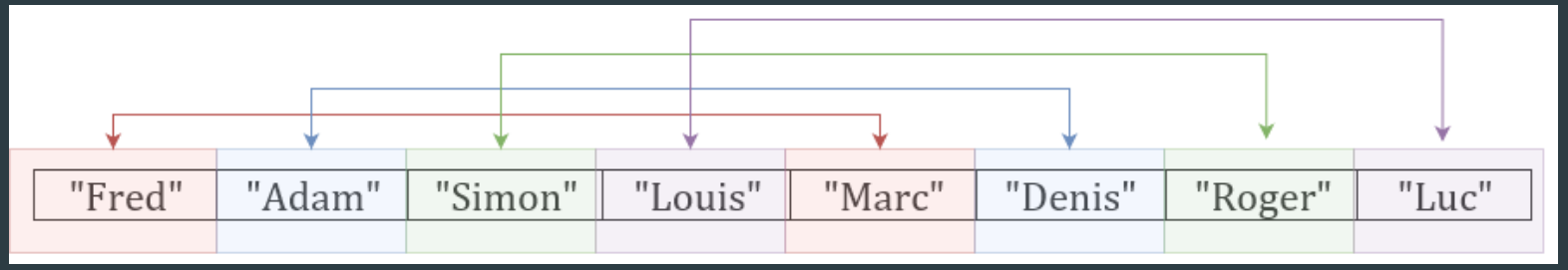
- Avant de penser au code, comment pourrait-on faire sur papier pour transformer la première liste en la deuxième?

Des idées pour résoudre la situation?

- Remplir une deuxième liste à partir de la première



- Échanger les éléments de position dans la liste



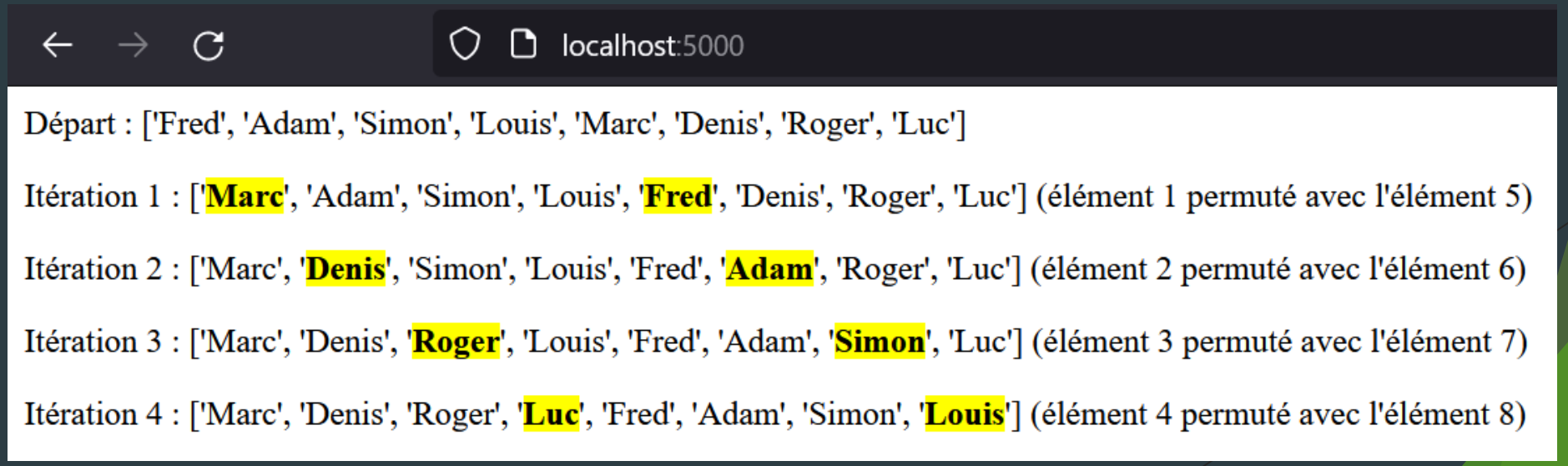
L'idée est de voir qu'il y a généralement plusieurs façons de résoudre un problème donné en informatique! ☺

Exemple de solution avec Python

- Permutation des noms, élément par élément

```python
# Fonction d'inversion de la liste
def inverserListe(liste):
    # On prend la taille de la liste et on effectue une division entière [//] par 2. **important pour ne pas avoir une position décimale
    positionMilieu = len(liste)//2
    # Pour chaque élément de la première moitié de la liste (de la position 0 à celle du milieu)
    for i in range(positionMilieu):
        # On effectue une permutation de l'élément à la position du curseur i avec celui à la position du curseur i+positionMilieu
        liste[i], liste[i+positionMilieu] = liste[i+positionMilieu], liste[i]
    # La liste passée dans la fonction est directement modifiée, donc pas besoin de la retourner
```

- Résultat à l'écran



Comment tester notre solution?

- Module de tests unitaires

```python
class TestInversionListe(unittest.TestCase):

    # Test qui valide que la fonction retourne la liste attendue
    def test_resultat_attendu(self):
        prenomsDebut = ["Alexandre", "Benoit", "Camille", "Diane", "Etienne", "Fiona", "Gabriel", "Helene"]
        prenomsFin = ["Etienne", "Fiona", "Gabriel", "Helene", "Alexandre", "Benoit", "Camille", "Diane"]
        inverserListe(prenomsDebut)
        self.assertEqual(prenomsDebut, prenomsFin)

    # Test qui valide que l'inverse de l'inverse nous redonne bien la liste de départ
    def test_inverse_inverse_liste(self):
        prenomsDebut = ["Alexandre", "Benoit", "Camille", "Diane", "Etienne", "Fiona", "Gabriel", "Helene"]
        prenomsDebutCopie = prenomsDebut.copy()
        inverserListe(prenomsDebut)
        inverserListe(prenomsDebut)
        self.assertEqual(prenomsDebut, prenomsDebutCopie)

    # Test assez superflus qui valide que la liste a la même longueur au départ qu'à la fin [Le premier test couvre ce cas indirectement
    def test_meme_longueur_liste(self):
        prenoms = ["Alexandre", "Benoit", "Camille", "Diane", "Etienne", "Fiona", "Gabriel", "Helene"]
        tailleDebut = len(prenoms)
        inverserListe(prenoms)
        self.assertEqual(tailleDebut, len(prenoms))

if __name__ == '__main__':
    unittest.main()
```

Comment tester notre solution?

- Exécution et résultat des tests :

```
Admin@DESKTOP-7IF1KNF MINGW64 ~/Desktop/EntrevueEnseignementCEGEP/simulationEnseignement (main)
$ py tests.py -v
```

```
Admin@DESKTOP-7IF1KNF MINGW64 ~/Desktop/EntrevueEnseignementCEGEP/simulationEnseignement (main)
$ py tests.py -v
test_inverse_inverse_liste (__main__.TestInversionListe) ... ok
test_meme_longueur_liste (__main__.TestInversionListe) ... ok
test_resultat_attendu (__main__.TestInversionListe) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.001s

OK
```

Questions avancées

1. Que se passera-t-il si l'on passe une liste impaire à la fonction?
2. Quelle est la complexité algorithmique de notre fonction InverserListe()?
3. Comment pourrions-nous modifier la fonction pour que notre liste d'origine ne soit pas modifiée et retourne plutôt une nouvelle liste?
4. Que se passe-t-il en mémoire lorsqu'on permute les valeurs dans cette liste?

Références

- https://www.adamsmith.haus/python/answers/how-to-swap-elements-in-a-list-in-python
- Code présenté disponible sur Github : https://github.com/sharktamer/simulationEnseignement