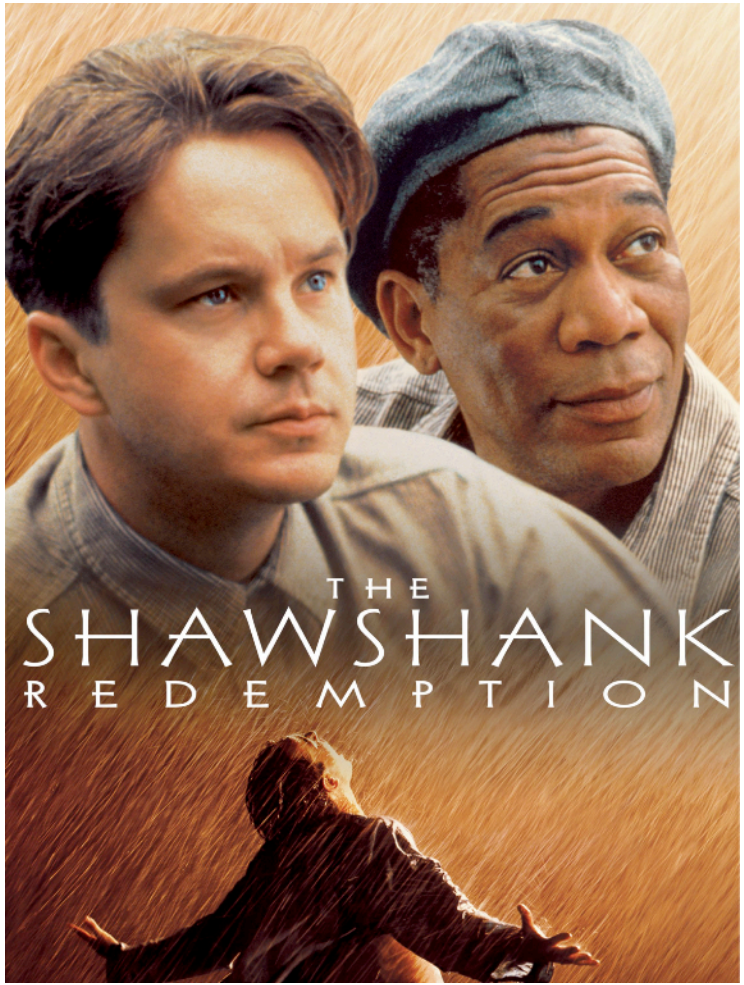


Information, Calcul et Communication

Partie Programmation

Cours 5: Modélisation d'un problème, celui des «100 prisonniers»

20.10.2023

Patrick Wang

Au sommaire cette semaine

1. Rappel du problème
2. Modélisation et implémentation de l'approche naïve
3. Modélisation et implémentation de l'approche optimale
4. Extra : Les modules Python

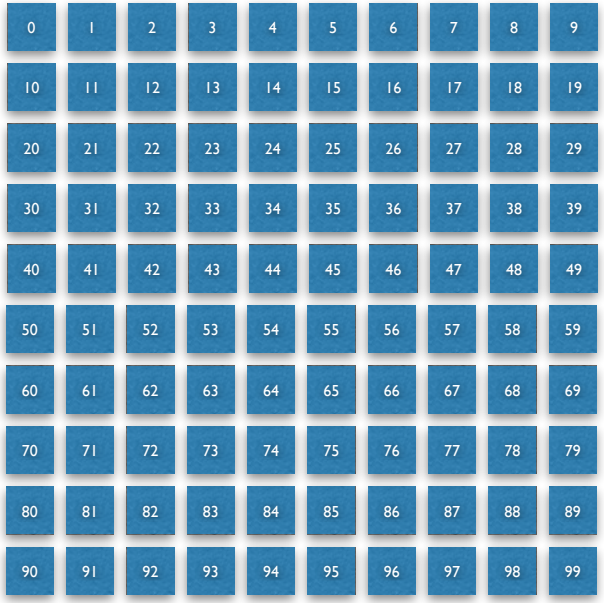
Au sommaire cette semaine

1. Rappel du problème
2. Modélisation et implémentation de l'approche naïve
3. Modélisation et implémentation de l'approche optimale
4. Extra : Les modules Python

1. Rappel du problème

Application «concrète» : Problème des 100 prisonniers

- 100 prisonniers sont dans des cellules numérotées de 0 à 99.
- Les clés de cellules sont mélangées et placées aléatoirement dans des boîtes, elles aussi numérotées de 0 à 99.
- Chaque prisonnier ne peut ouvrir que 50 boîtes.
- Si tous les prisonniers trouvent leurs clés, alors ils peuvent s'échapper. Si un seul ne trouve pas, alors ils restent enfermés.
- Approche naïve : $\frac{1}{2^{100}}$
- Approche optimale : environ 31% !
- https://www.youtube.com/watch?v=iSNsgj1OCLA



Au sommaire cette semaine

1. Rappel du problème
2. Modélisation et implémentation de l’approche naïve
3. Modélisation et implémentation de l’approche optimale
4. Extra : Les modules Python

2. Approche naïve

Description de l'approche naïve

- L'approche naïve est celle qui repose sur le hasard :
  - Chaque prisonnier choisit aléatoirement un coffre qu'il n'a pas encore ouvert
  - Il a 50 essais pour retrouver sa clé parmi les 100 coffres.

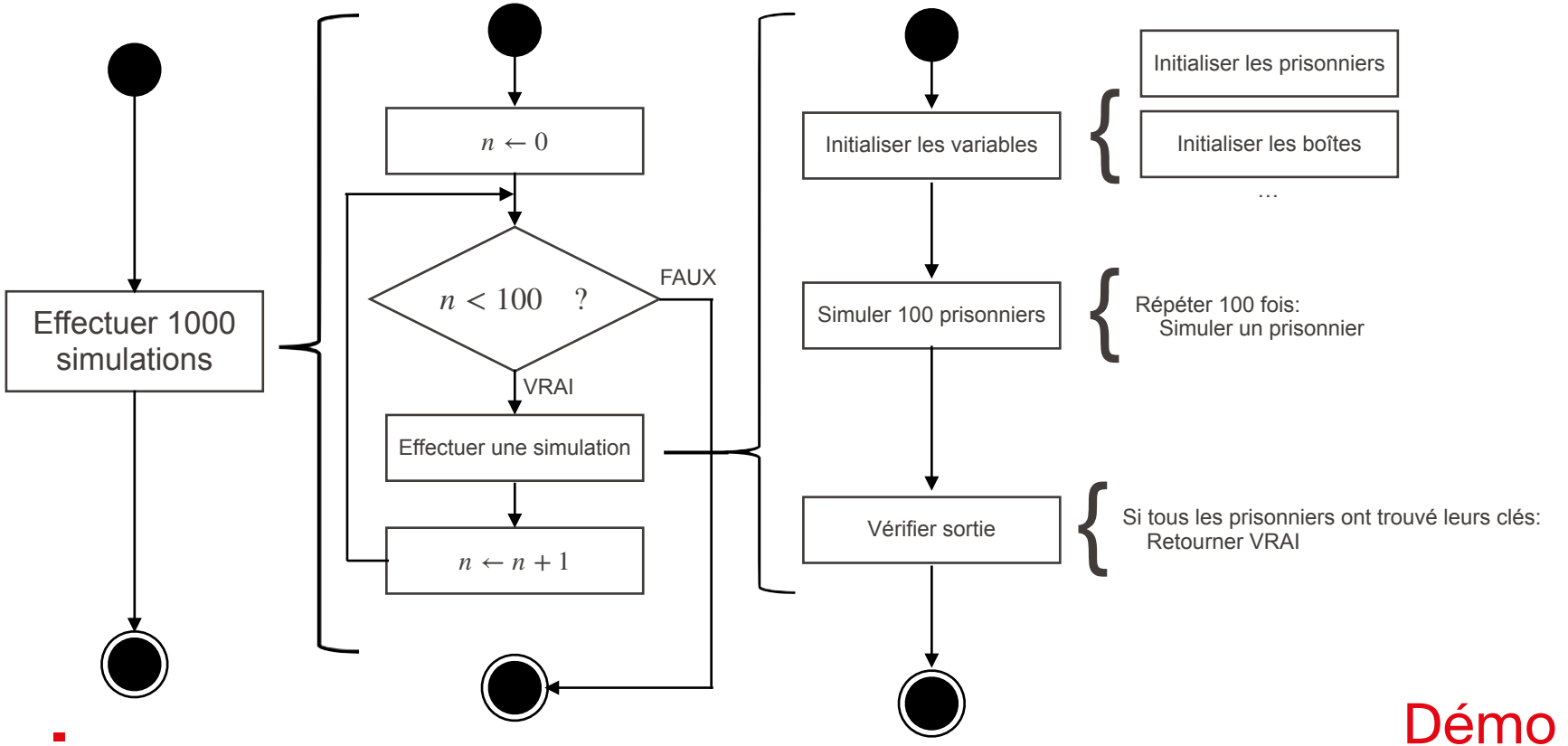
2. Approche naïve

Modélisation et *stepwise refinement*

- Le stepwise refinement est une méthode qui guide la conception d'un algorithme et son implémentation.
- Il s'agit de déterminer décomposer un problème en problèmes plus petits et faciles à résoudre. Ces-derniers peuvent eux-mêmes être décomposés par la suite.

2. Approche naïve

Modélisation et *stepwise refinement*



Démo !

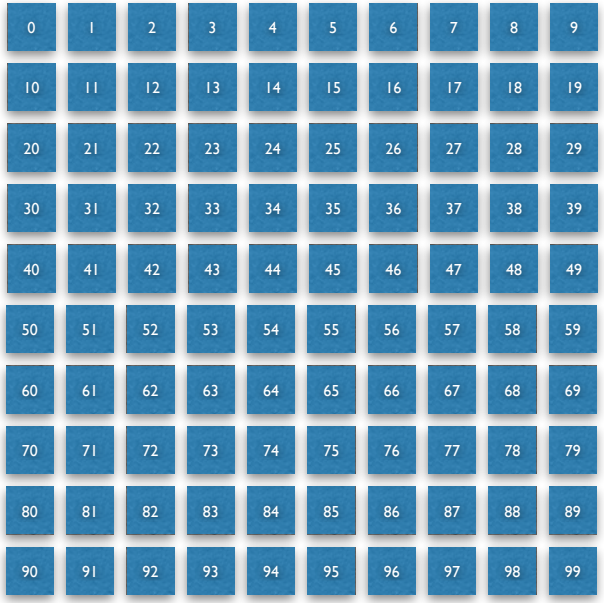
Au sommaire cette semaine

1. Rappel du problème
2. Modélisation et implémentation de l'approche naïve
3. Modélisation et implémentation de l'approche optimale
4. Extra : Les modules Python

3. Approche optimale

Description de l'approche optimale

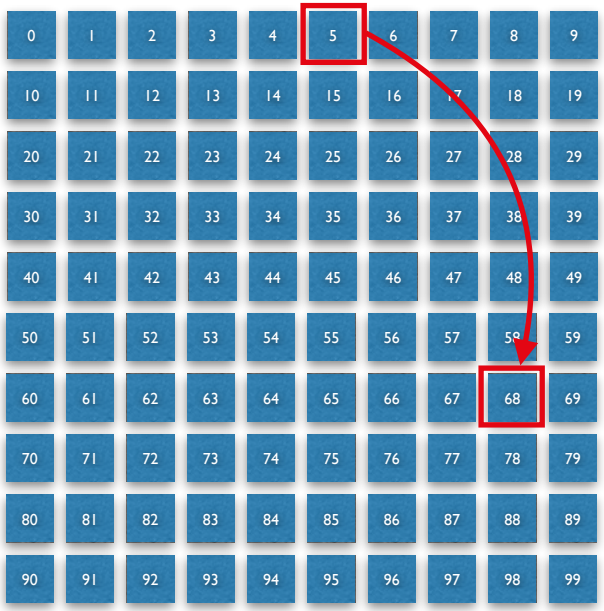
- Chaque prisonnier commence par la boîte portant son numéro
- Si la caisse contient sa clé, alors c'est fini
- Sinon, il ouvre la boîte portant le même numéro que la clé trouvée



3. Approche optimale

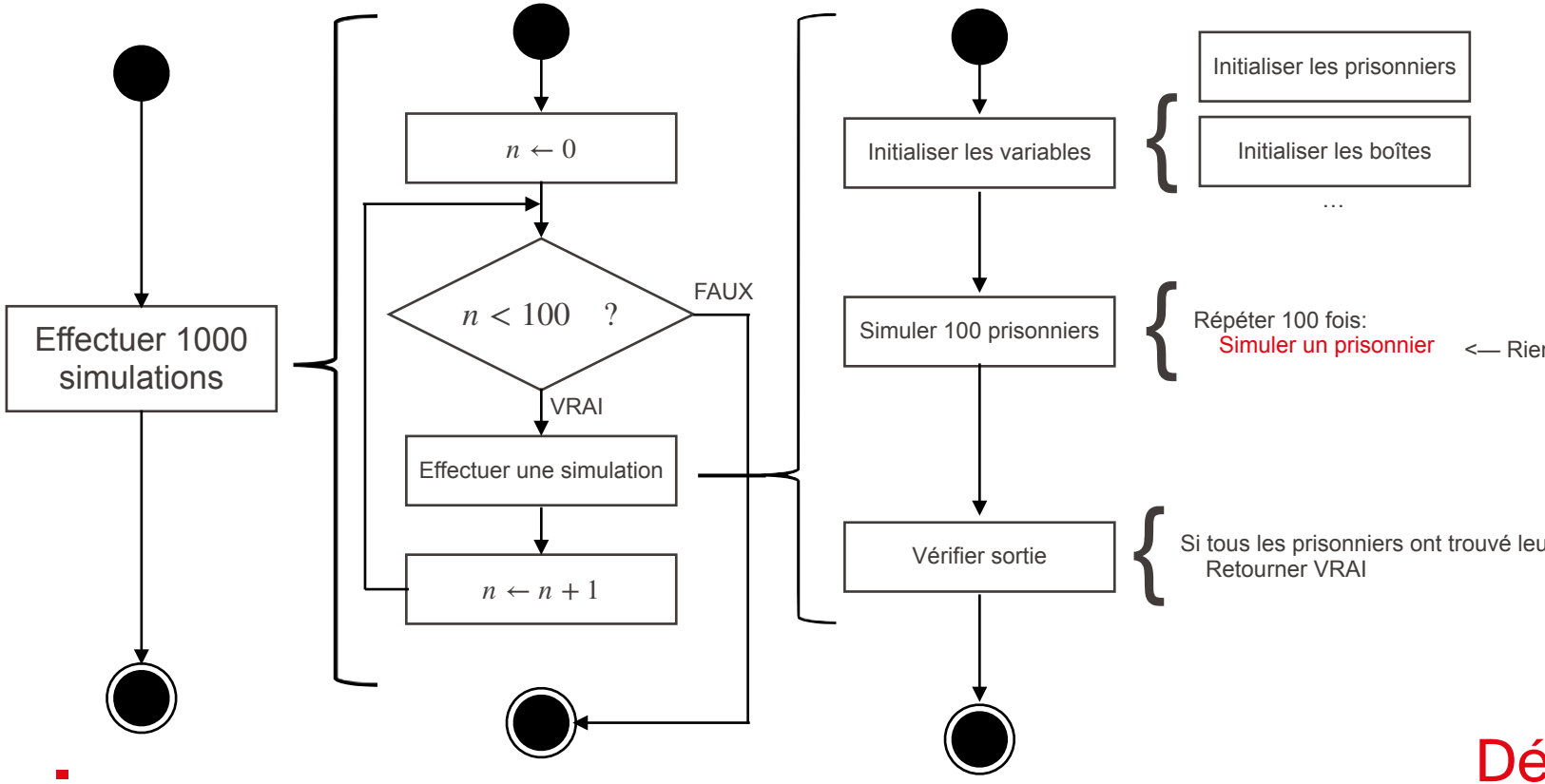
Exemple : Prisonnier #5

- Prisonnier #5 commence par la boîte #5
- La boîte #5 contient la clé #68, la prochaine boîte à ouvrir est donc la #68
- La boîte #68 contient la clé #n, la prochaine boîte à ouvrir est donc la #n
- etc...



3. Approche naïve

Modélisation et *stepwise refinement*



Démo !

Au sommaire cette semaine

1. Rappel du problème
2. Modélisation et implémentation de l'approche naïve
3. Modélisation et implémentation de l'approche optimale
4. Extra : Les modules Python

4. Extra : Les *modules* Python

Exemple de départ

- Nous définissons des fonctions dans pleins de fichier `.py` différents
- On a peut-être même redéfinit plusieurs fois la même fonction dans des fichiers différents...

c05_module.py ICC • c05_module.py/ leap_year

```python
def leap_year(year: int) -> bool:
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

```

c05_test_module.py 2 ICC • c05_test_module.py

```python
print(leap_year(2024))
```

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
❯ PYTHONIOENCODING='utf-8' venv/bin/python3 "/Users/p54244/dev/ICC/c05_test_module.py"
Traceback (most recent call last):
  File "/Users/p54244/dev/ICC/c05_test_module.py", line 1, in <module>
    print(leap_year(2024))
          ^^^^^^^^^
NameError: name 'leap_year' is not defined
```

4. Extra : Les *modules* Python

Les modules, et comment les importer

- Un module est un fichier Python qui définit des fonctions, que l'on peut importer dans d'autres fichiers
- Il est possible de créer des modules, ou d'utiliser ceux qui développés par d'autres

```python
from typing import List
from random import randint, shuffle
from math import ceil, cos, log2

random_number = randint(0, 100)
```

```python
import typing
import random
import math

random_number = random.randint(0, 100)
```

4. Extra : Les *modules* Python

Les modules que l'on crée

c05_module.py ICC • c05_module.py/ leap_year

```python
def leap_year(year: int) -> bool:
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

```

c05_test_module.py ICC • c05_test_module.py

```python
from c05_module import leap_year

print(leap_year(2024))

```

PROBLEMS 20 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
❯ PYTHONIOENCODING='utf-8' venv/bin/python3 "/Users/p54244/dev/ICC/c05_test_module.py"
True
```