



1

Quelques travaux de T^{le} NSI

NSI



T^{le} NSI



Approfondir la programmation

Programmation Orientée Objet

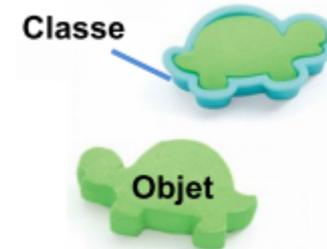
2

Programmation Orientée Objet

Privé
Encapsulation
Classes

Public
Attributs
Constructeur

Modèles
Accesseurs



**Une nouvelle façon
d'écrire des
programmes**



Approfondir la programmation

Programmation Orientée Objet

3

Un mini jeu de rôle du genre donjons et dragons

| MONSTRE TUE CREATION D'UN NOUVEAU MONSTRE |

CREATION DU MONSTRE

Point de vie : 10

Force d'attaque : 3

Attaque par le héros : 6

Attaque du monstre : 4

!!! Le monstre est touché

Point de vie du héros : 15

Point de vie du monstre : 5

Attaque par le héros : 10

Attaque du monstre : 5

!!! Le monstre est touché

Point de vie du héros : 15

Point de vie du monstre : 0



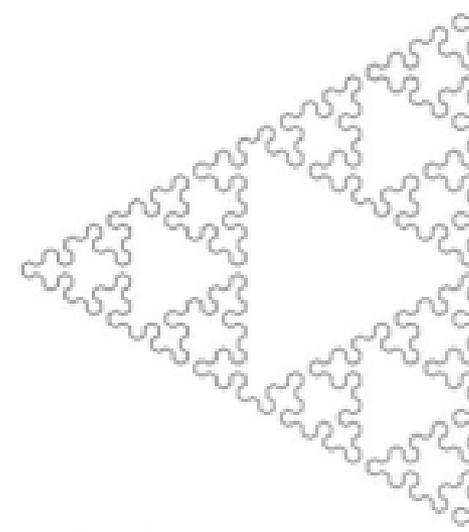
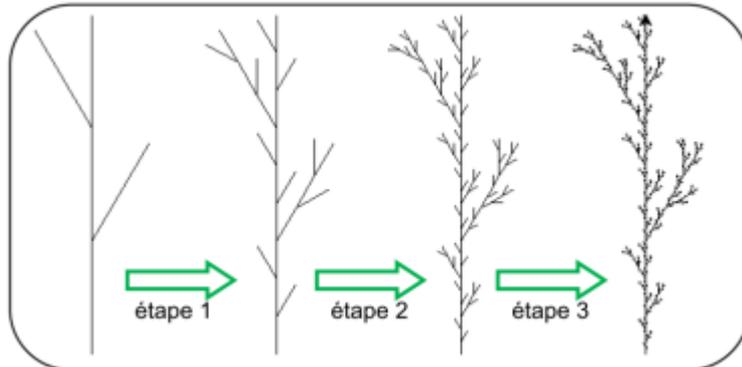
**Mise en œuvre autour
d'un mini jeu**



Approfondir la programmation

4

Programmation récursive





Approfondir la programmation

5



Script_récuratif_3. Coder la fonction de Fibonacci en complétant le script  Fibonacci-Simple-depart-eleve.py

```
from functools import lru_cache
```

Votre fonction à réaliser

```
@lru_cache(maxsize = 3000)
```

```
def fibonacci(n):
```

```
    ...
```

Mise en oeuvre du cache lru

Statistiques à posteriori avec la commande

```
>>> fibonacci.cache_info()
```

```
CacheInfo(hits=1994, misses=1000, maxsize=1000, currsize=1000)
```

```
    ...
```

```
if n == 0:
```

```
    return 0
```

```
elif n == 1:
```

```
    return 1
```

```
elif n == 2:
```

```
    return 1
```

```
else:
```

```
    return fibonacci(n-2) + fibonacci(n-1)
```

La suite de Fibonacci



Programmation récursive



Les types abstraits de données

6

Structures de données : les listes

Liste, Pile, File



Mise en œuvre de types de données permettant des algorithmes de plus en plus performants.



Les types abstraits de données

7

Utilisateur qui se propose d'utiliser la structure de donnée pour son prochain algorithme de recherche de chemin dans un jeu.



Interface

```
ajouteElement(Liste)  
estVide(Liste)  
supprimeElement(Liste)
```

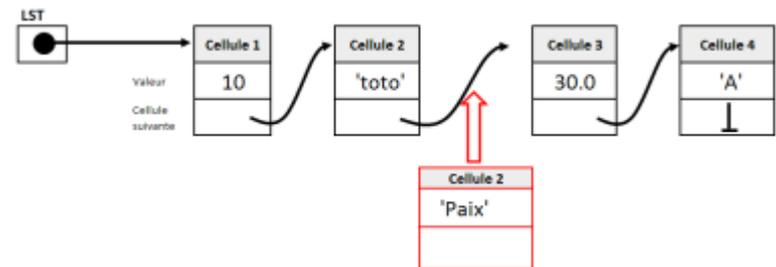
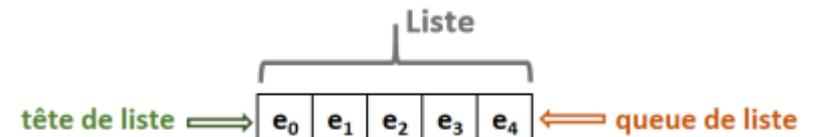
Implémentation

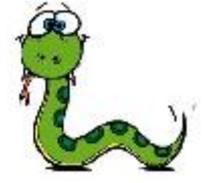
```
def supprimerElement(self):  
    if len(self.__elements)==0:  
        raise IndexError("Pile Vide")  
    return self.__elements.pop()
```



Développeur qui implémente la structure de donnée. Il se demande s'il ne va pas en réaliser une autre version plus rapide prochainement.

Structures de données : les listes





Les types abstraits de données

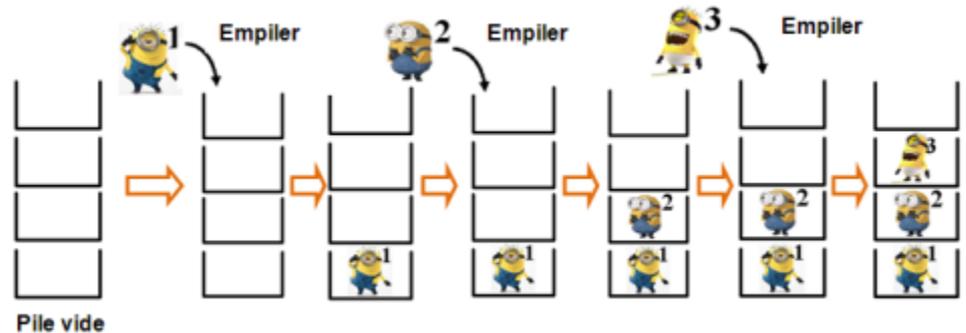
8

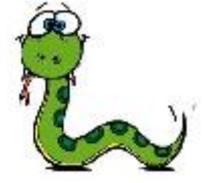
Structures de données : les piles et les files

L_{iste}, P_{ile}, F_{ile}



Une Pile d'assiettes





Les types abstraits de données

9

```
class Pile:  
    def __init__(self):  
        self.lst = []  
  
    def estVide(self):  
        return self.lst == []  
  
    def empiler(self, x):  
        self.lst.append(x)  
  
    def depiler(self):  
        if self.estVide():  
            raise ValueError("  
        return self.lst.pop()  
  
    def longueur(self):  
        return len(self.lst)
```

L iste, P ile, F ile



Script_TAD_7. Coder le script ci-dessus et ajouter une méthode `__str__` qui renvoie la valeur de l'élément au dessus de la pile si il existe ou une chaine 'Pile vide' dans le cas contraire. Tester votre classe avec l'exemple des trois minions ci-dessus.

Structures de données : les piles et les files



Les types abstraits de données

10

Structures de données : les piles et les files

First In First Out.

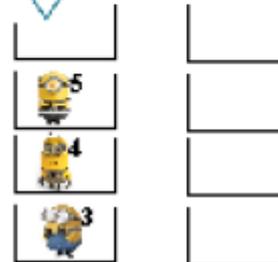


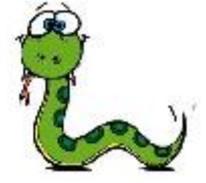
Enfiler()

Entrée



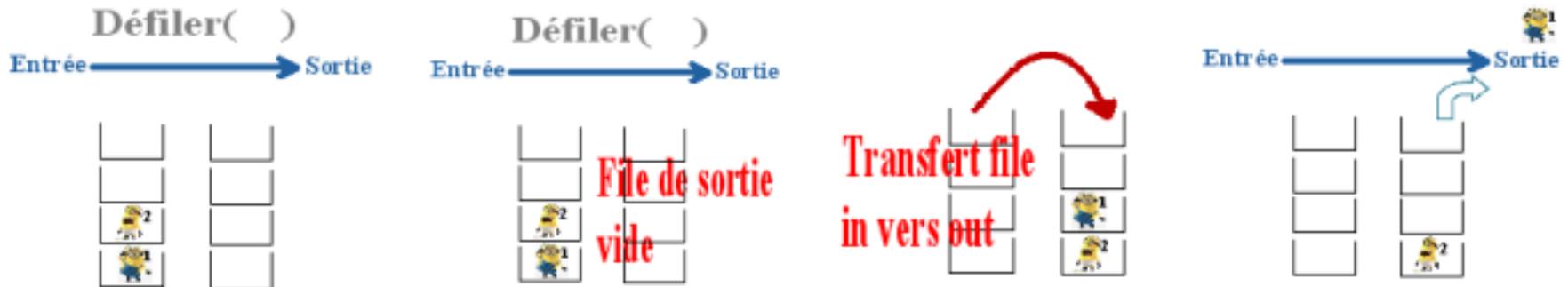
Sortie





Les types abstraits de données

11



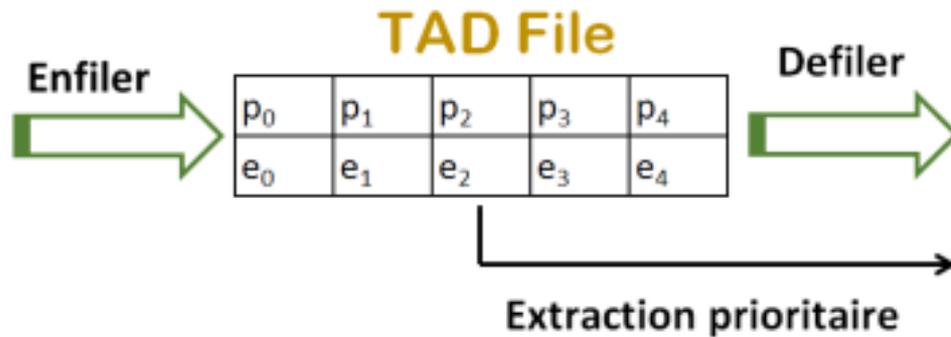
Structures de données : les piles et les files



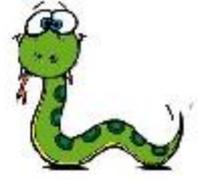
Les types abstraits de données

12

Projet : Implémentation d'une File à priorité



Structures de données : les piles et les files



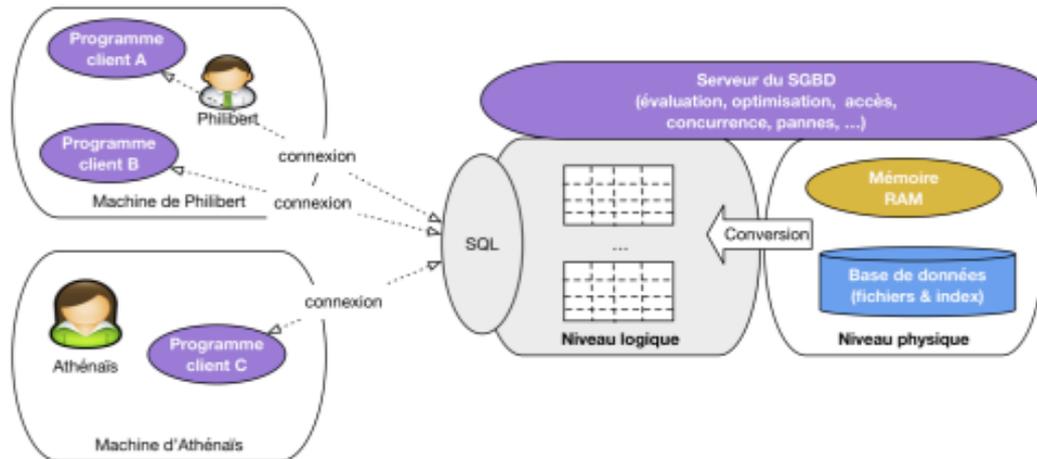
Étude et application des bases de données

13

Les bases de données



Le plus connu est le langage SQL : Structured Query Language





Étude et application des bases de données

14

Les bases de données TP Zoo¹

Résumé :

Le plan du zoo²

Voilà une mise en œuvre d'une base de données autour d'un jardin zoologique. Cela nous permettra d'appliquer des requêtes élaborées telles que les jointures ou opérations ensemblistes.

L'implémentation en Python est également proposée.



Python



Database System



Étude et application des bases de données

15

Le schéma de la base de données zoo

LesCages

<u>noCage</u>	fonction	noAllee
integer	varchar(100)	integer

LesAnimaux

<u>nomA</u>	sexe	typeA	pays	anNais	#noCage
varchar(100)	varchar(12)	varchar(100)	varchar(100)	integer	integer

LesMaladies

<u>#nomA</u>	<u>nomM</u>
varchar(100)	varchar(100)

LesResponsables

<u>noAllee</u>	#nomE
integer	varchar(100)

LesEmployees

<u>nomE</u>	adresse
varchar(100)	varchar(200)

LesGardiens

<u>noCage</u>	#nomE
integer	varchar(100)



Étude et application des bases de données

16

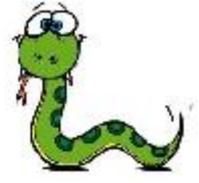
Ajouter un nouvel enregistrement

```
INSERT INTO LesCages VALUES (13, 'grande volière', 2);
```

Table : LesCages

	noCage	fonction	noAllee
	Filtre	Filtre	Filtre
1	11	fauve	10
2	1	fosse	1
3	2	aquarium	1
4	3	petits oiseaux	2
5	4	grand aquarium	1
6	12	fauve	10
7	5	fauve	10
8	13	grande volière	2

**Exemple de résultat
d'une requête**



Étude et application des bases de données

17

Donner les noms des animaux originaires du Kenya et qui ont contracté une grippe.

```
SELECT A.nomA FROM LesAnimaux A, LesMaladies M
WHERE ((A.pays='Kenya') AND
       (A.nomA=M.nomA) AND
       (M.nomM='grippe'))
);
```

Les résultats :

	nomA
1	Charly
2	Charlotte

Exemple de résultat d'une requête



Un jeu autour des bases de données !

18

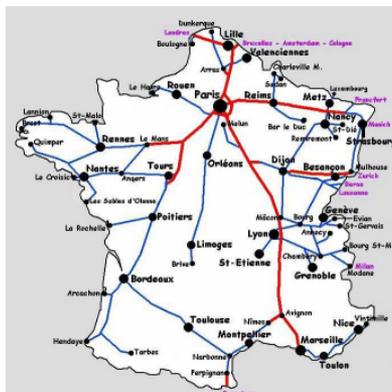
Première partie

Vous devez créer la base :

 VOYAGE_SNCF.db

A partir du fichier :

 DEFI_CREATION_BASE.sql



Train

numT	provenance	destination	horaireDepart	horaireArrivee
integer	varchar2(20)	varchar2(20)	time	time

Exemple : "20:30"

Reservation

numR	nomClient	prenomClient	prix	#numT
integer	varchar2(20)	varchar2(20)	float	integer

Défi n°1 :

Analyse du contenu de la base.



Défi n°2 :

Quels enregistrements, ou n-uplets, posent problèmes dans la base.



Un jeu autour des bases de données !

19

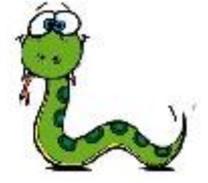
Deuxième partie

On utilise la base :  ZOO.db

Défi n°5 :

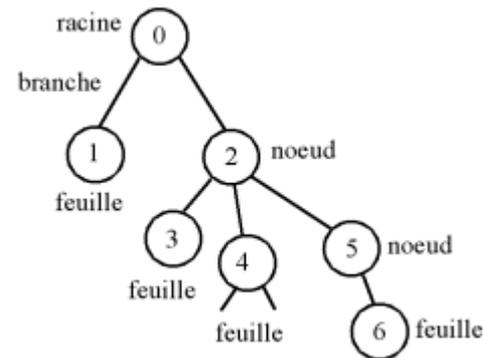
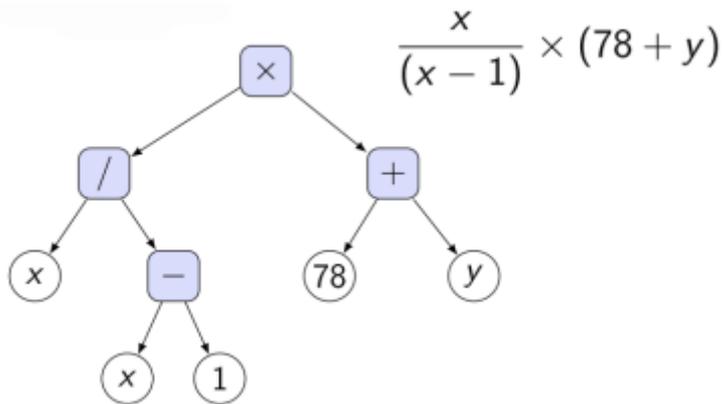
Préparer trois questions que vous proposerez en défi aux trois autres équipes.





Les types abstraits de données suite

Structures de données : les arbres





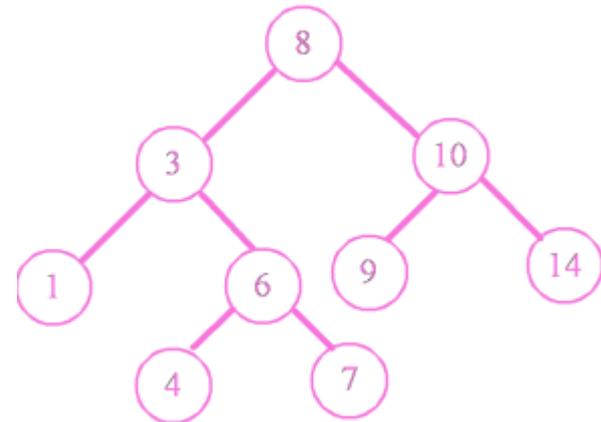
Les types abstraits de données suite

21

Les arbres binaires de recherche ou ABR

Les arbres binaires de recherche sont des arbres binaires possédant une organisation particulière des nœuds.

Cette organisation permet d'effectuer des recherches de manière très efficace puisqu'elles sont effectuées dans le meilleur des cas avec une complexité dépendant de la hauteur de l'arbre.





Les types abstraits de données suite

22

Les graphes



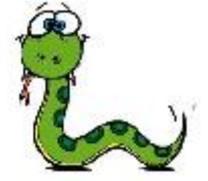
Le réseau d'amis de Facebook

2



Réseau autoroutier

1



Les types abstraits de données suite

23

Les graphes

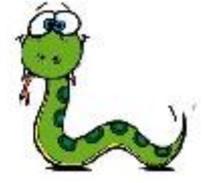
```

5 class Graphe:
6     """un graphe représenté par une matrice d'adjacence,
7     où les sommets sont les entiers 0,1,...,n-1"""
8
9     def __init__(self, n):
10        self.n = n
11        self.adj = [[False] * n for _ in range(n)]
12
13        def ajouter_arc(self, s1, s2):
14            self.adj[s1][s2] = True
15
16        def arc(self, s1, s2):
17            return self.adj[s1][s2]
18
19        def voisins(self, s):
20            v = []
21            for i in range(self.n):
22                if self.adj[s][i]:
23                    v.append(i)
24            return v
    
```



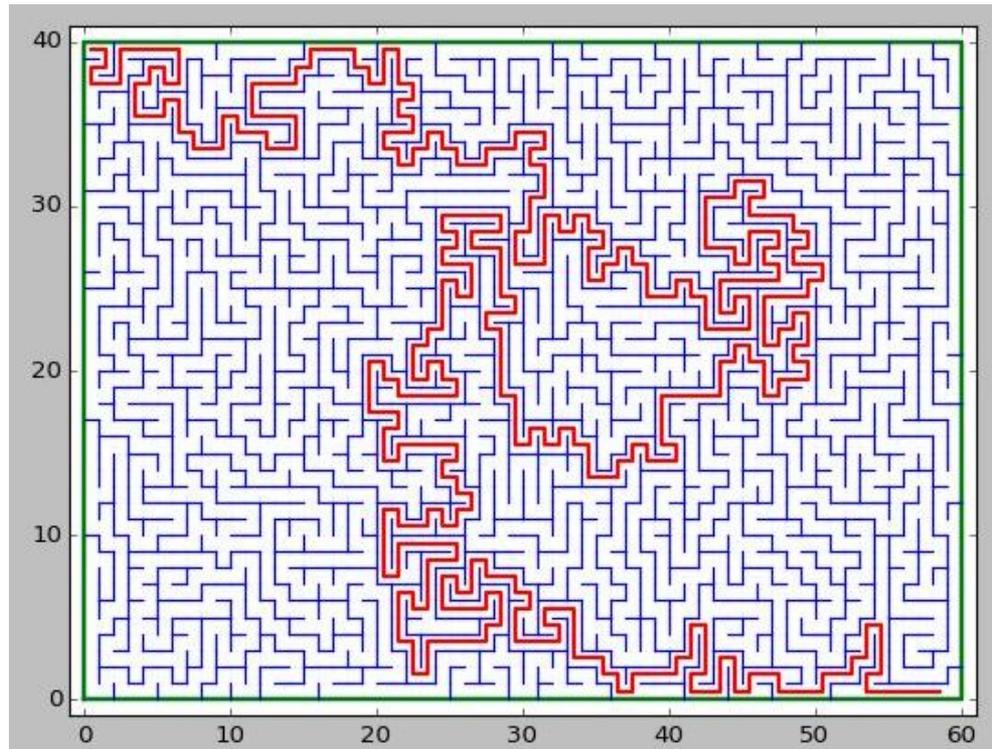
Script_Graphe_1. Compléter le script pour décrire le graphe ci-dessous. (*Graphe identique à celui de la question Q7 mais avec les nœuds numérotés à partir de 0*)

 Script_Graphe_1.py



Application finale : trouver la sortie d'un labyrinthe

24

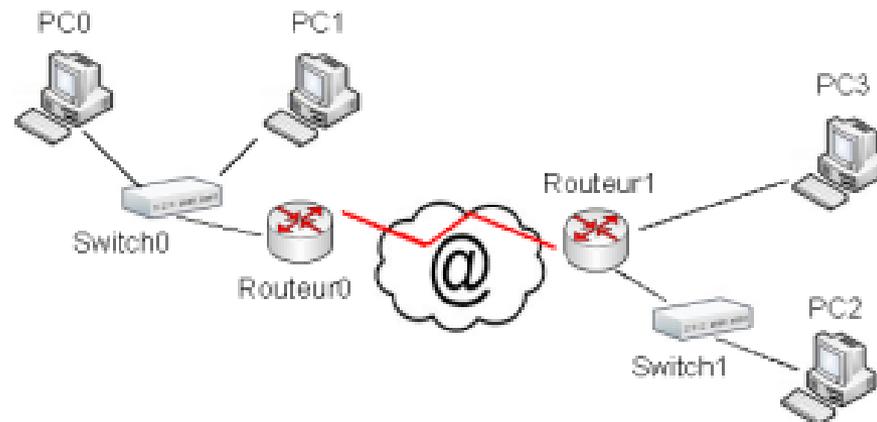




Suite de l'étude des réseaux

25

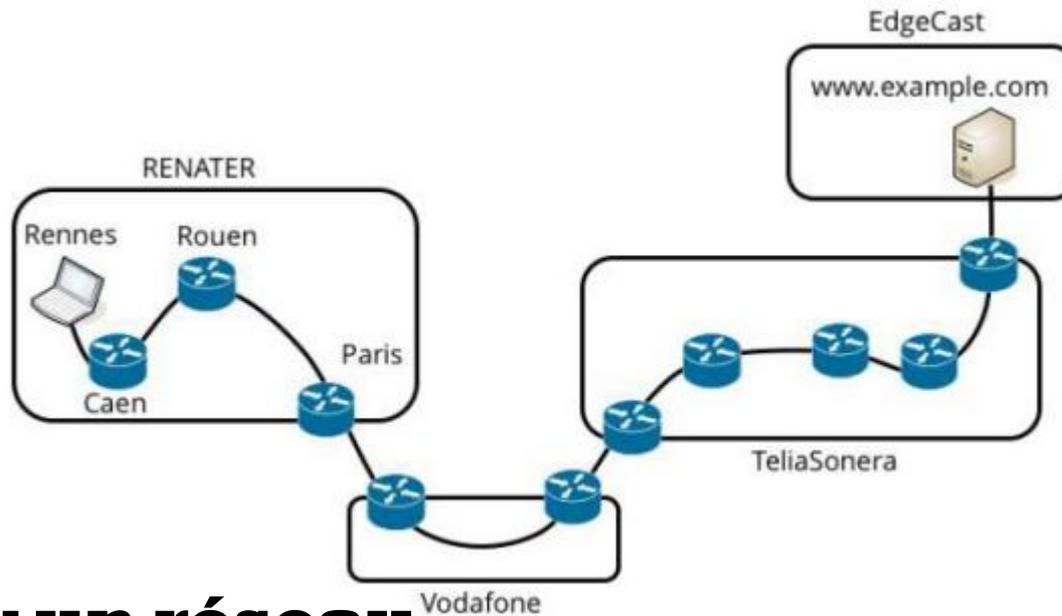
Notion de réseaux : protocole de routage





Suite de l'étude des réseaux

26

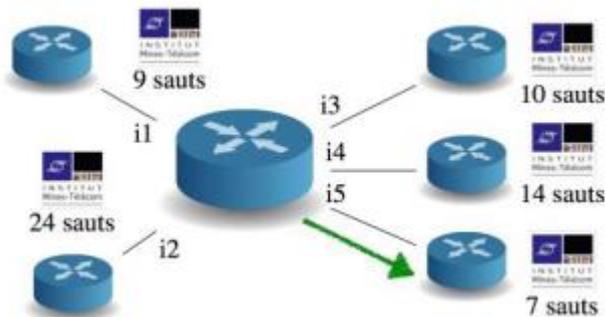


**Internet : un réseau
de réseaux**



Suite de l'étude des réseaux

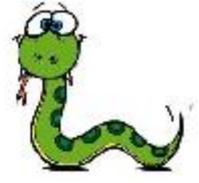
27



Système Autonome

- Une seule entité de gestion
- Une seule entité pour le routage externe
- Exemples:
 - Fournisseur de contenu
 - Fournisseur d'accès à Internet
 - Toute entreprise avec un accès à Internet

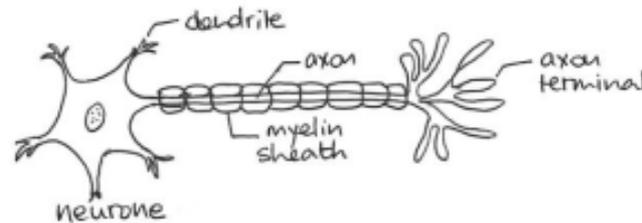
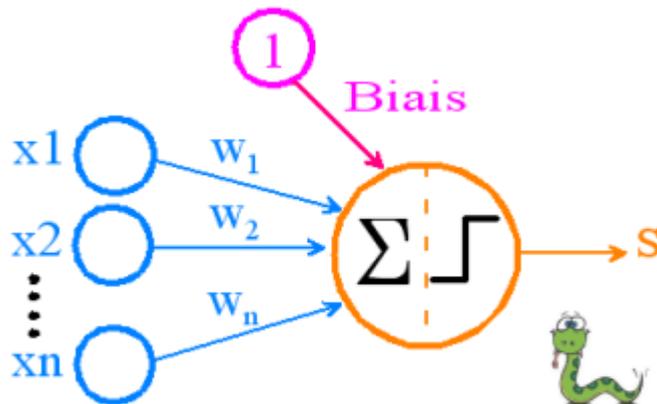
Comparaison de différents protocoles de routage des paquets de données : RIP et OSPF



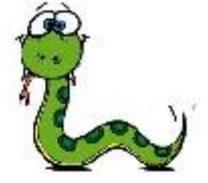
Projets en TP autour du machine learning

28

Machine Learning : le perceptron un premier réseau de neurones



Script_Perceptron_1. Compléter le script  Script_Perceptron_1.py pour faire fonctionner votre premier perceptron.

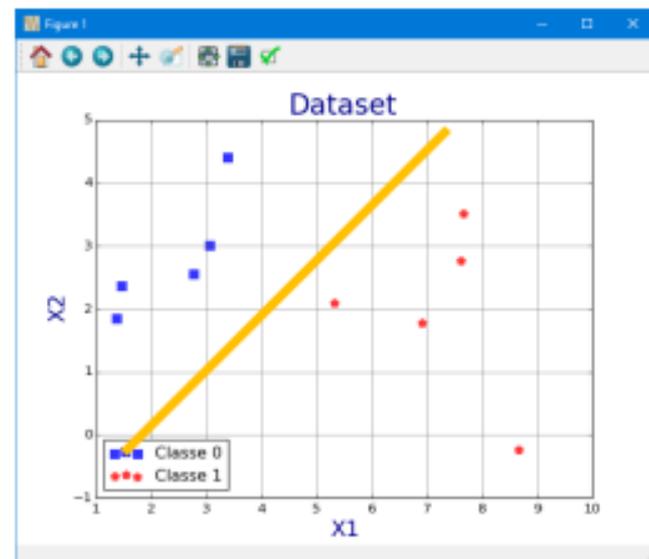
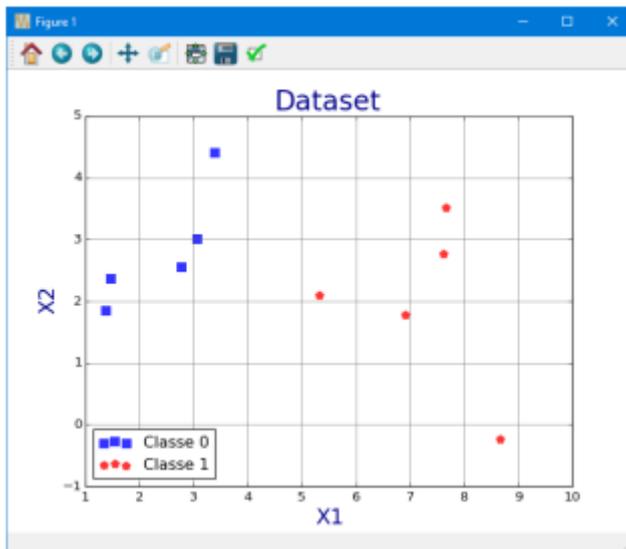


Projets en TP autour du machine learning

29



Le perceptron peut réaliser une classification linéaire en deux classes d'un jeu de données. Le plan est découpé par une droite en deux parties.





Projets en TP autour du machine learning

30

Machine Learning : utilisation du perceptron appliquée à la chasse aux mines marines

Faire la différence entre des données sonar identifiant des mines ou des rochers avec un seul neurone !



J'ai mis un masque je ne crains pas l'eau !!



Projets en TP autour du machine learning

31

Machine Learning : utilisation du perceptron
appliquée à la chasse aux mines marines





Projets en TP autour du machine learning

32

Bravo tout est en place vous pouvez procéder à quelques essais de détection de mines, bonne chance !!

| Machine Learning | Modèle : Perceptron | Analyse : k-folds | Data : sonar.all-data.csv |

P.G Avril 2021

Nombre de folds : 8
Réglage seed : 1
l_rate : 0.1
Epochs : 100

Itération : 0 n° fold test : 0 n° folds entraînement :	[1, 2, 3, 4, 5, 6, 7]	Précision de la prédiction :	80.77%
Itération : 1 n° fold test : 1 n° folds entraînement :	[0, 2, 3, 4, 5, 6, 7]	Précision de la prédiction :	80.77%
Itération : 2 n° fold test : 2 n° folds entraînement :	[0, 1, 3, 4, 5, 6, 7]	Précision de la prédiction :	76.92%
Itération : 3 n° fold test : 3 n° folds entraînement :	[0, 1, 2, 4, 5, 6, 7]	Précision de la prédiction :	80.77%
Itération : 4 n° fold test : 4 n° folds entraînement :	[0, 1, 2, 3, 5, 6, 7]	Précision de la prédiction :	65.38%
Itération : 5 n° fold test : 5 n° folds entraînement :	[0, 1, 2, 3, 4, 6, 7]	Précision de la prédiction :	80.77%
Itération : 6 n° fold test : 6 n° folds entraînement :	[0, 1, 2, 3, 4, 5, 7]	Précision de la prédiction :	65.38%
Itération : 7 n° fold test : 7 n° folds entraînement :	[0, 1, 2, 3, 4, 5, 6]	Précision de la prédiction :	69.23%

Précision de la prédiction sur le dataset complet : 81.25%



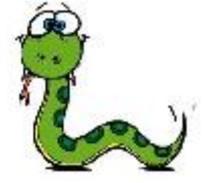
Projets en TP autour du machine learning

33



J'ai mis un
masque je ne
crains pas l'eau !!





Projet : base de données, requête SQL avec Python

34

Projet Base de Données

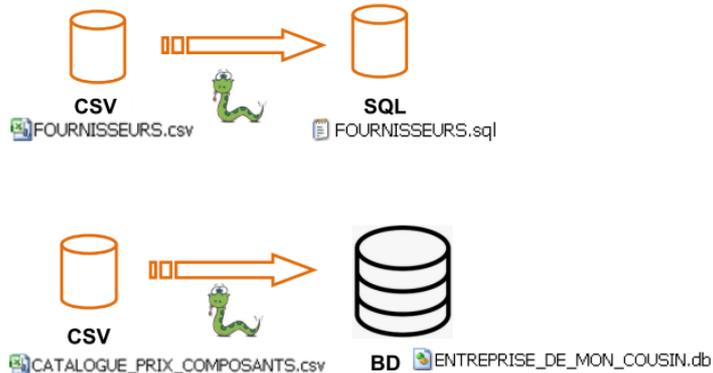


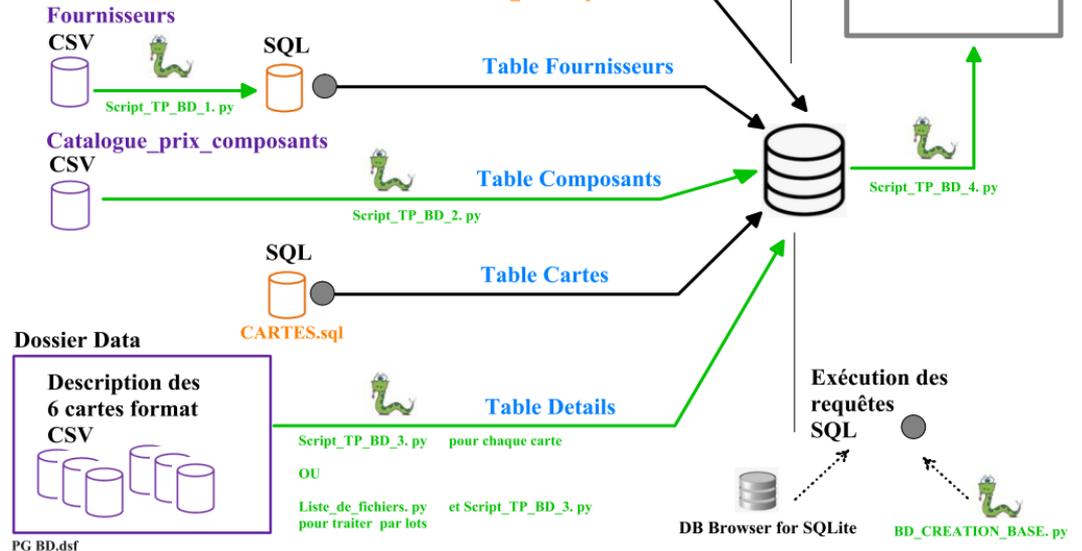
Schéma de la base

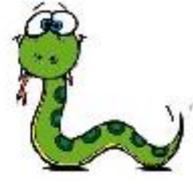
Attribution	SQL	SQL	SQL
Fournisseurs	four	fourier2019	fourier2019
Composants	comp	comp2019	comp2019
Cartes	carte	carte2019	carte2019
Details	detail	detail	detail

Requêtes de création de la base

Création et remplissage de la base

Utilisation de la base





Aperçu des activités en T¹^e NSI

35

**Merci
pour votre
attention**