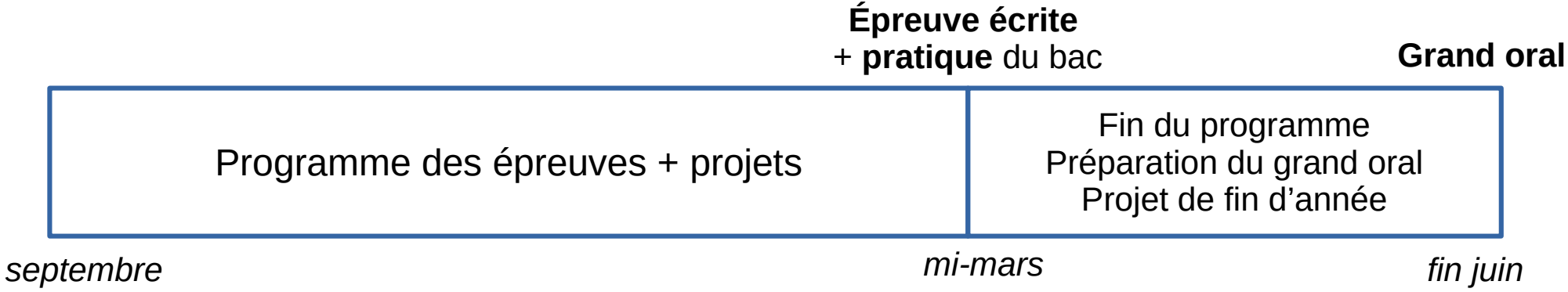
Programme de terminale NSI

L. TREBAUL

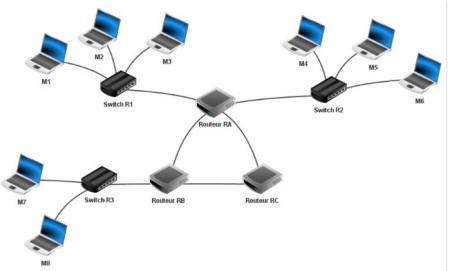
Organisation

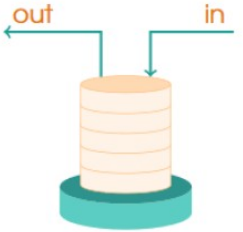
- 6h par semaine
- Une année rythmée par le bac :

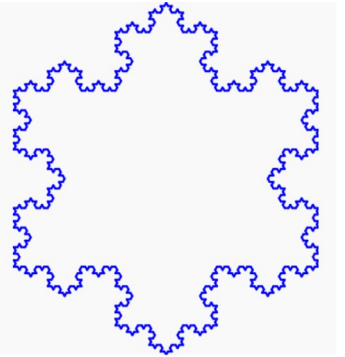


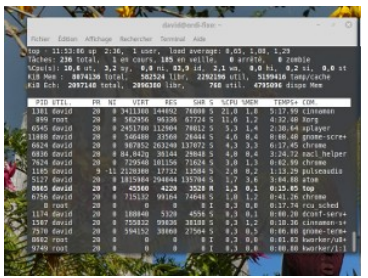
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) Architecture matérielle & systèmes d’exploitation
5) Bases de données
6) Algorithmique









Programme

1) Programmation

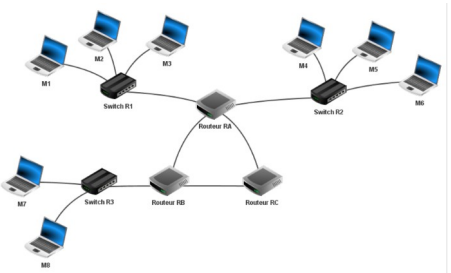
2) Structures de données

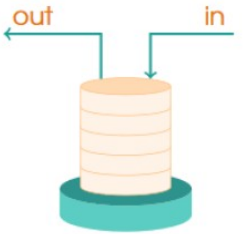
3) Réseaux informatiques

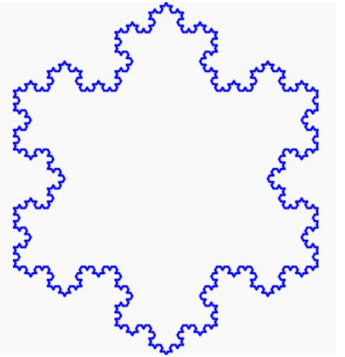
4) Architecture matérielle & systèmes d’exploitation

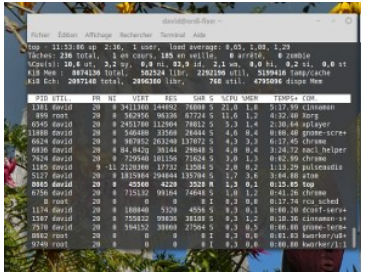
5) Bases de données

6) Algorithmique

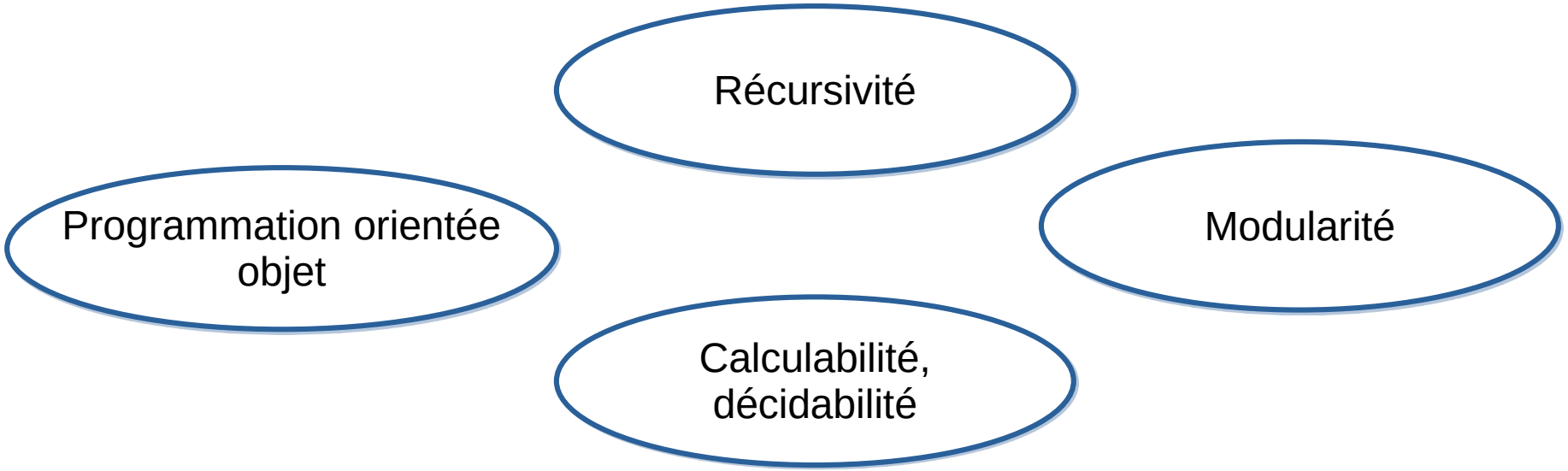




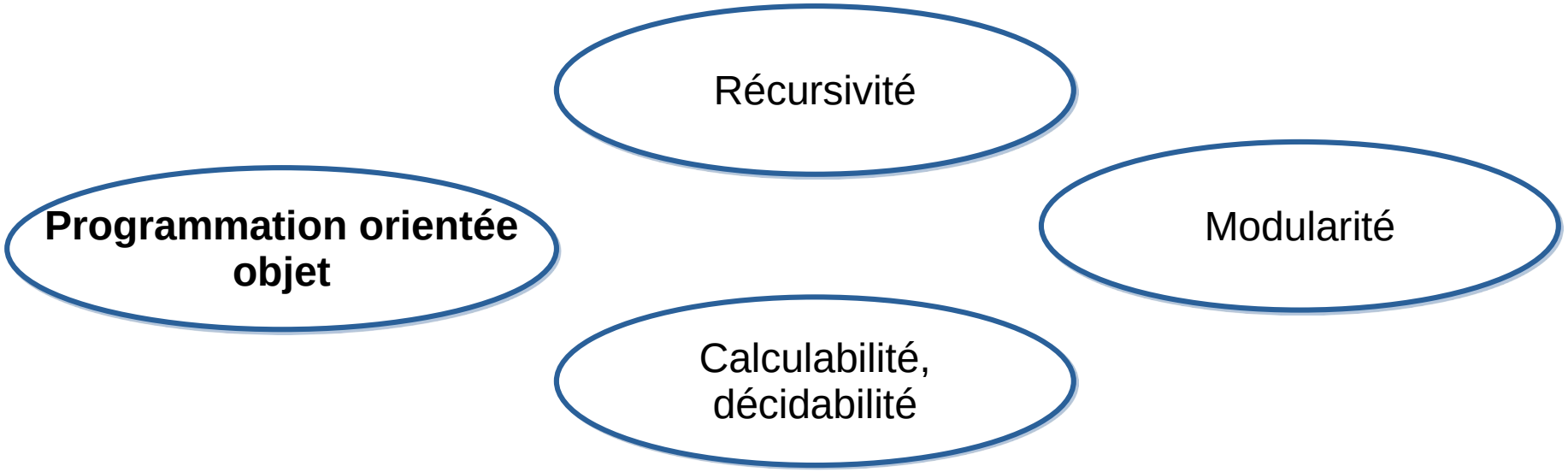




1- Programmation

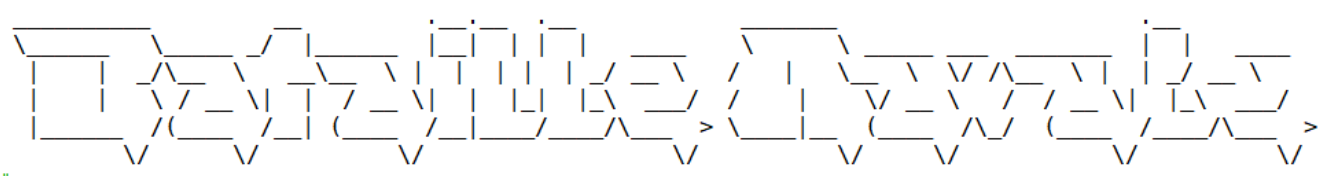


1- Programmation



Projets en Programmation Orientée Objet

Un **jeu de bataille navale**



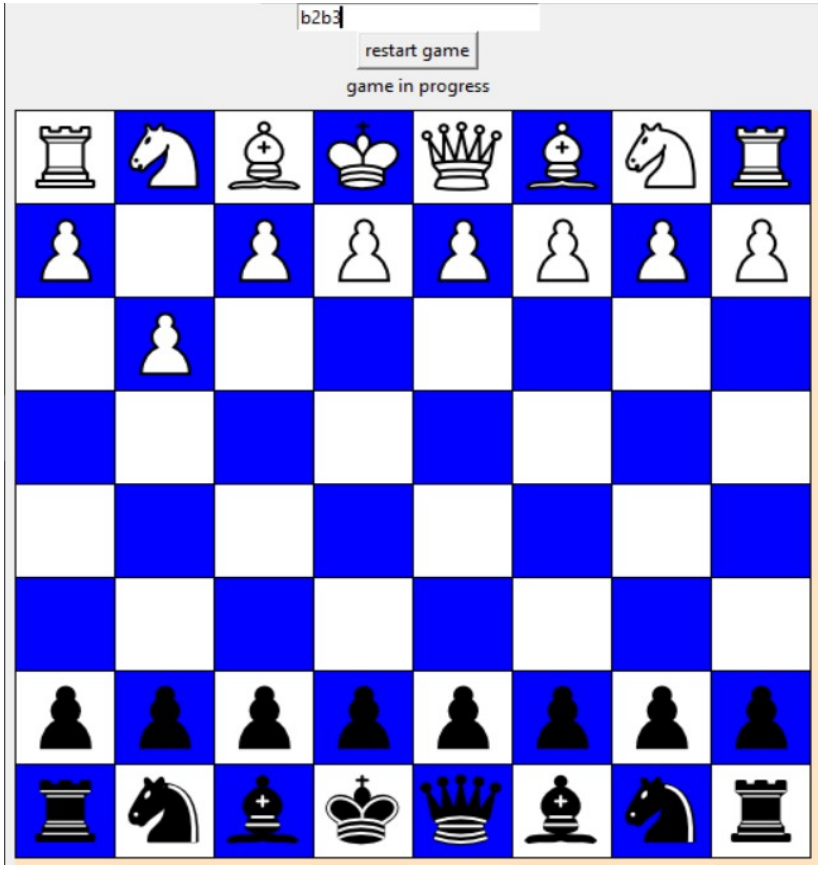
```
print("Placement du torpilleur (deux cases)")
while torpilleurUser.setPlace(x, y, direction) == False:
    while True:
        try:
            x = ligne[input("Entrez une ligne (entre A et J): ")]
            break
        except KeyError:
            x = ligne[input("Entrez une ligne (entre A et J): ")]
    while True:
        try:
            y = int(input("Entrez une colonne (Entre 1 et 10): "))
            break
        except ValueError:
            y = int(input("Entrez une colonne (Entre 1 et 10): "))
    while True:
        try:
            direction = orientation[input("Entrez une direction (haut, droite,
            break
        except KeyError:
            direction = orientation[input("Entrez une direction (haut, droite,
            torpilleurUser.setPlace(x, y, direction)
    print("Les coordonées que vous avez entré ne fonctionnent pas, veuillez re

print(tab(l))

x = 0
y = 0
direction = 0

sousmarinUser.setPlace(x, y, direction)

print("Placement du sous-marin (trois cases)")
while sousmarinUser.setPlace(x, y, direction) == False:
    while True:
        try:
            x = ligne[input("Entrez une ligne (entre A et J): ")]
            break
        except KeyError:
            x = ligne[input("Entrez une ligne (entre A et J): ")]
```

```
[' ', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10']

['A', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['B', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['C', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['D', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['E', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['F', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['G', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['H', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['I', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

['J', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']

Placement du torpilleur (deux cases)

Entrez une ligne (entre A et J):
```

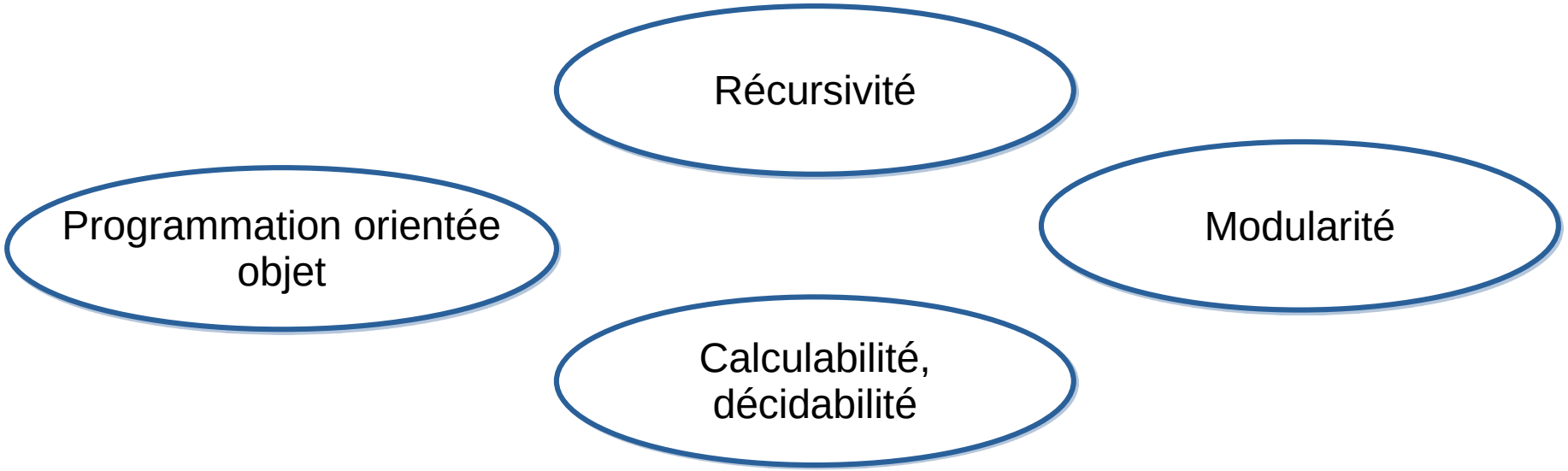
Projets en Programmation Orientée Objet

Un **jeu d'échecs** : vérification par l'ordinateur de la validité des coups

```
class Pawn(Piece):

    def basepos(self):#on établit la position initialle du pion pour savoir de combien i
        if self.color=="b":
            return 6
        else:return 1

    def __init__(self, color, piece,y,x): #init, on prend en considération sa position e
        super().__init__(color, piece,y,x)
        if self.color =="b":
            self.advancement = -1
        else:
            self.advancement = 1
        self.value = 1

    def getmoverange(self): #moverange du pion, dépend de multiples facteurs, renvoie un
        moverange = []
        if isempty(board,self.y+self.advancement,self.x):#avancer de 1
            moverange.append((self.y+self.advancement,self.x))
        if self.y == self.basepos():
            if isempty(board,self.y+2*self.advancement,self.x) and isempty(board,self.y+
                moverange.append((self.y+2*self.advancement,self.x))
        if self.x<7:
            if not isempty(board,self.y+self.advancement,self.x+1):
                if self.color != checkboard(board,self.y+self.advancement,self.x+1):
                    moverange.append((self.y+self.advancement,self.x+1))
        if 0<self.x:
            if not isempty(board,self.y+self.advancement,self.x-1):
                if self.color != checkboard(board,self.y+self.advancement,self.x-1):
                    moverange.append((self.y+self.advancement,self.x-1))
        return moverange

    def geteatrange(self): #les endroits où le pion peut manger une pièce
        eatrange = []
        if self.x<7:
            if self.color != checkboard(board,self.y+self.advancement,self.x+1):
                eatrange.append((self.y+self.advancement,self.x+1))
        if 0<self.x:
            if self.color != checkboard(board,self.y+self.advancement,self.x-1):
                eatrange.append((self.y+self.advancement,self.x-1))
        return eatrange
```

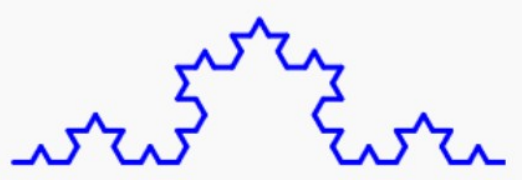


1- Programmation



1- Programmation

Les tours de Hanoï



Le flocon de von Koch

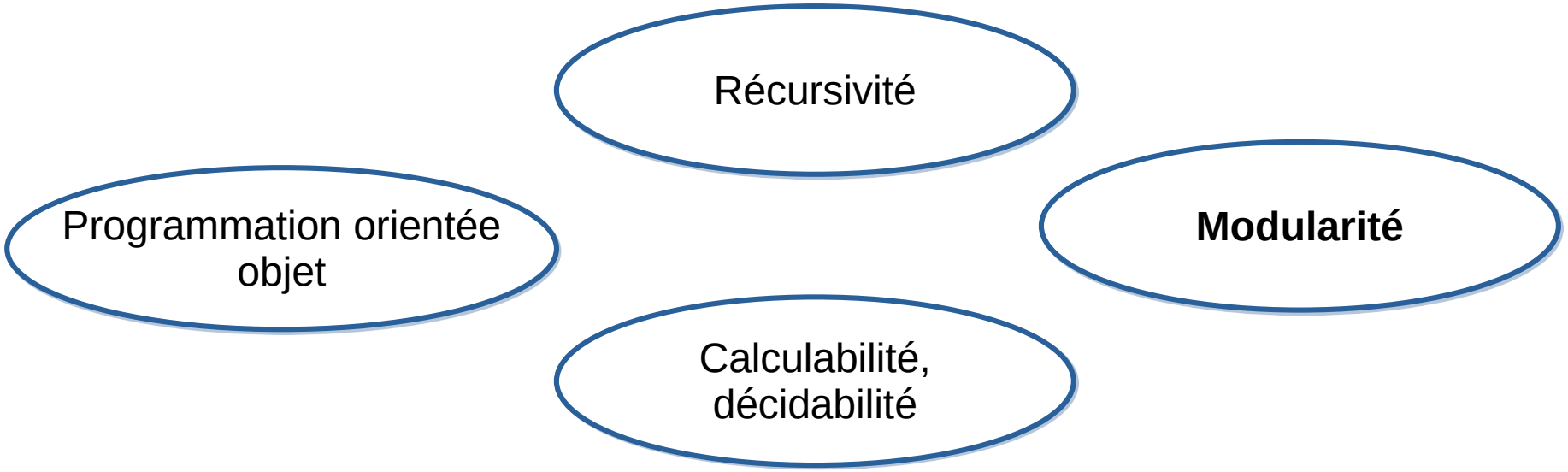
Récursivité

Programmation orientée objet

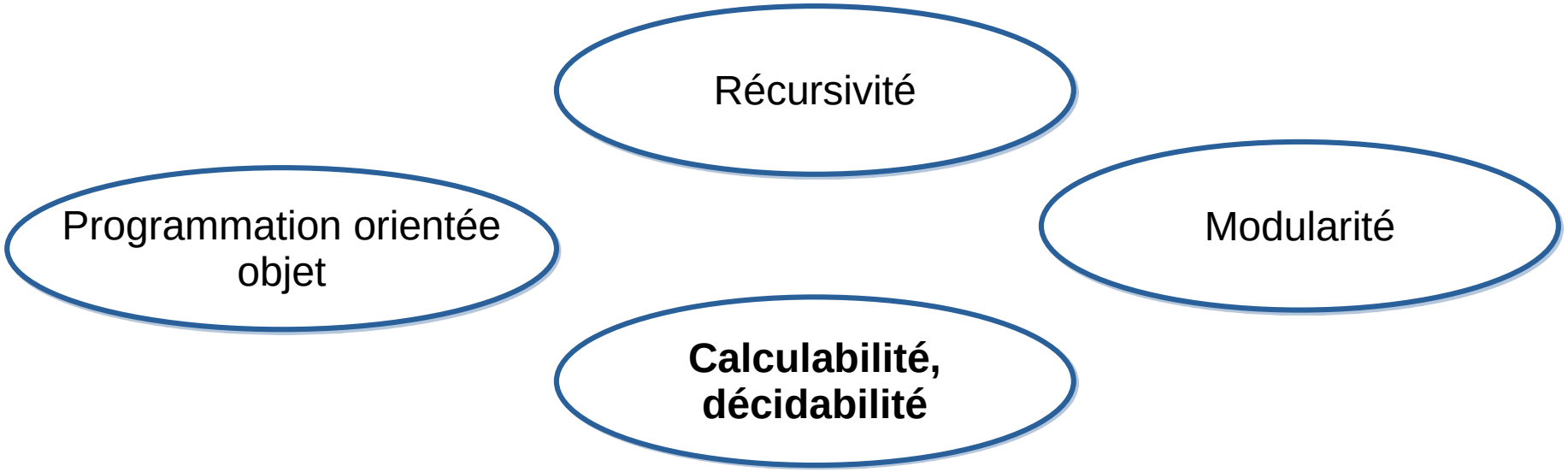
Modularité

Calculabilité, décidabilité

1- Programmation

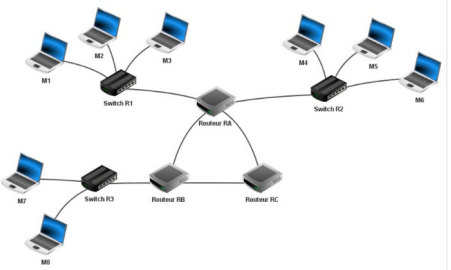


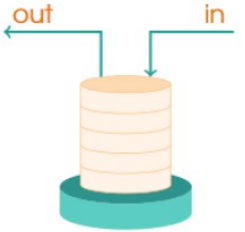
1- Programmation

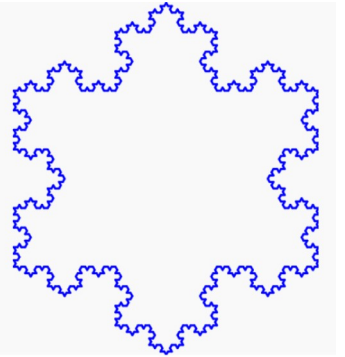


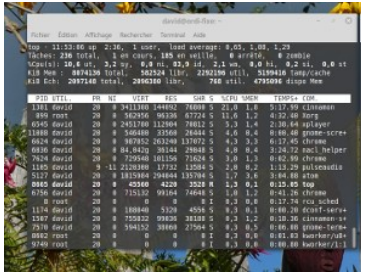
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) Architecture matérielle & systèmes d'exploitation
5) Bases de données
6) Algorithmique



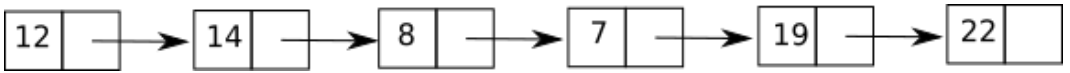




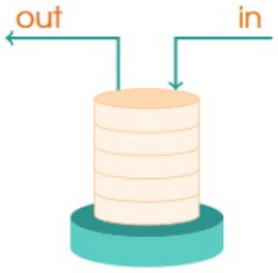


2- Structures de données

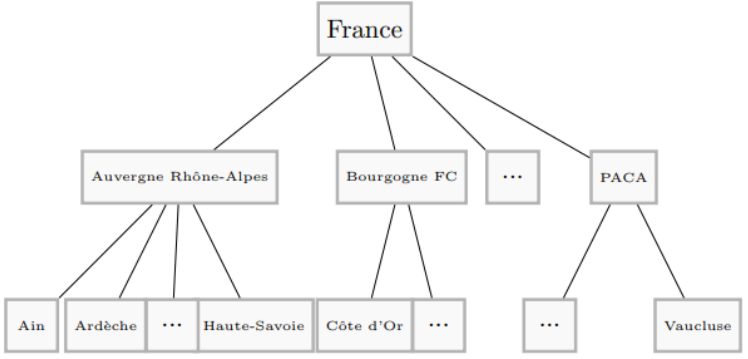
Comment agencer les données
pour mieux répondre à des problèmes ?



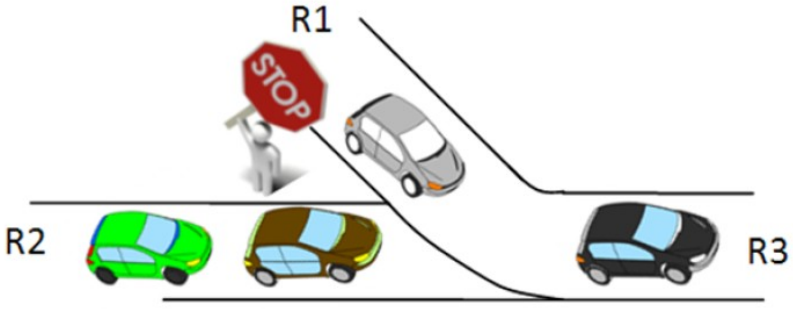
Liste chaînée



Pile



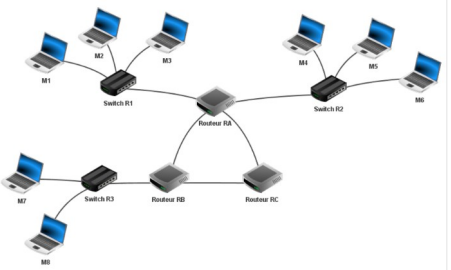
Arbre

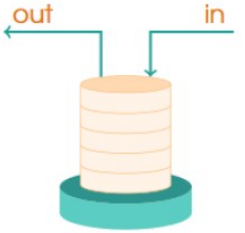


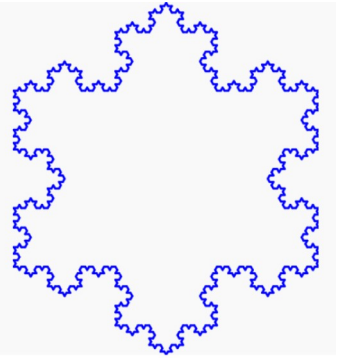
Utilisation des
files

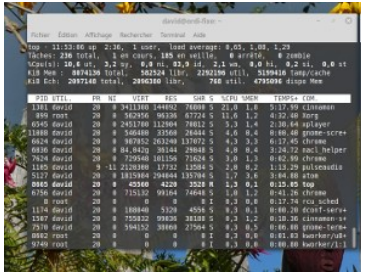
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) Architecture matérielle & systèmes d’exploitation
5) Bases de données
6) Algorithmique



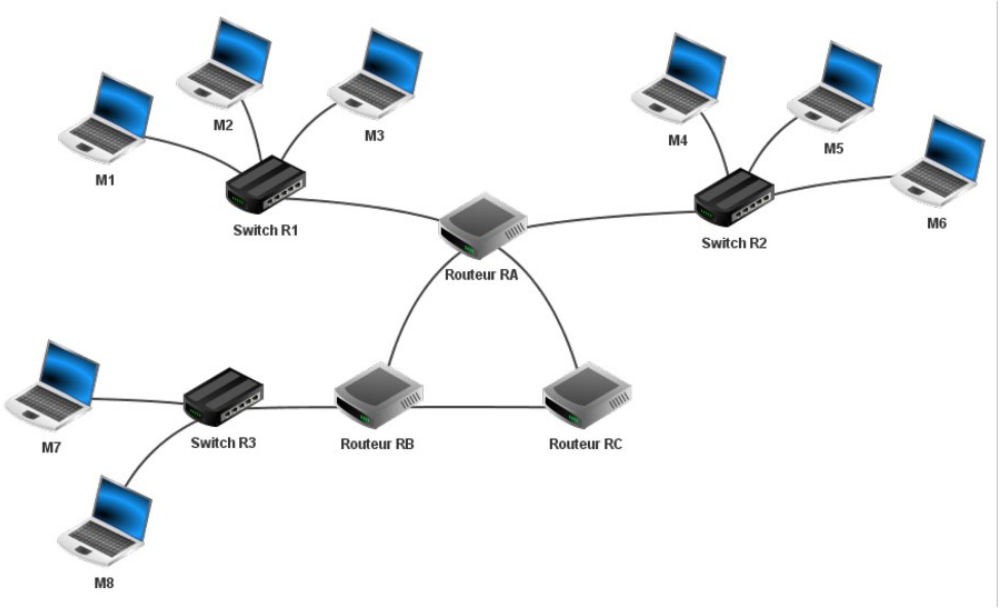






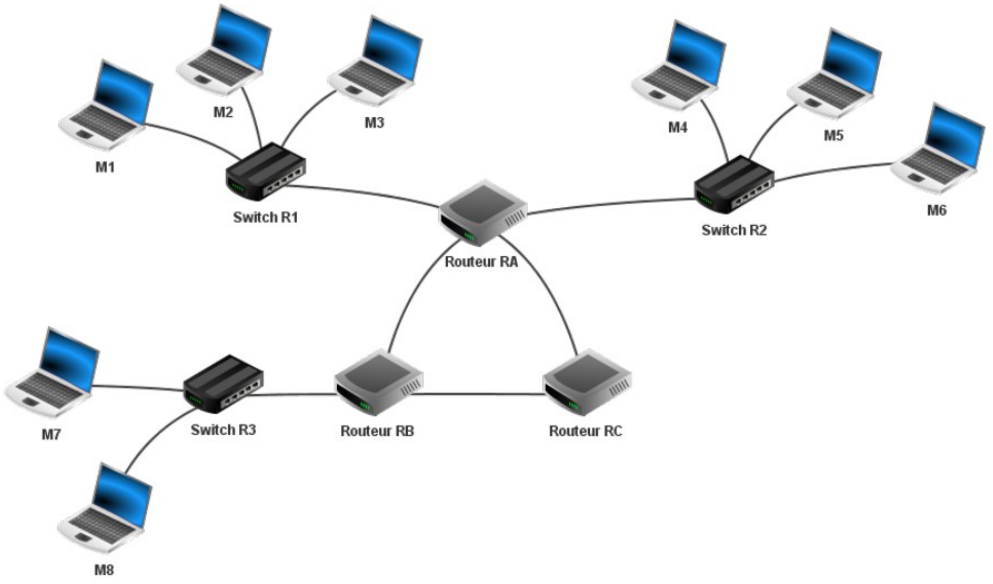
3- Réseaux informatiques

Comment *guider* les paquets dans un réseau ?

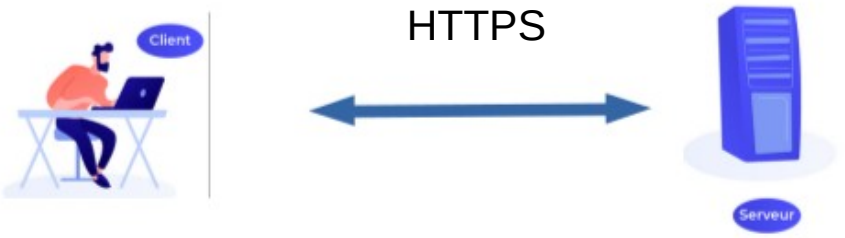


3- Réseaux informatiques

Comment *guider* les paquets dans un réseau ?

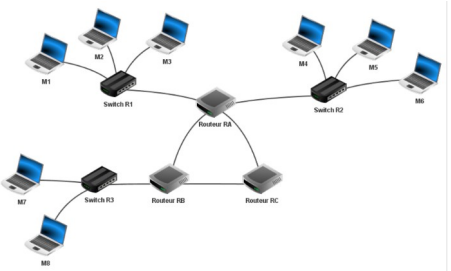


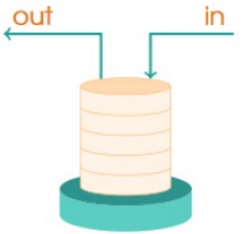
Comment *sécuriser* l'échange d'informations ?

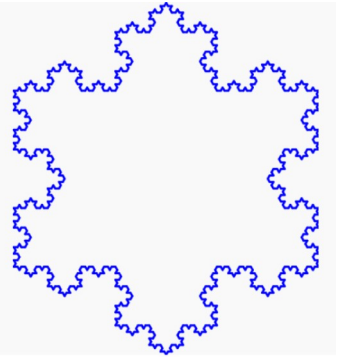


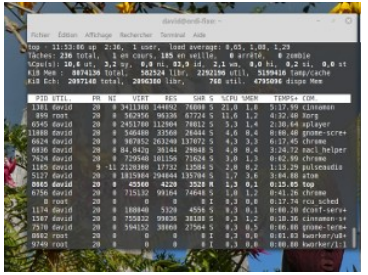
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) **Architecture matérielle & systèmes d'exploitation**
5) Bases de données
6) Algorithmique







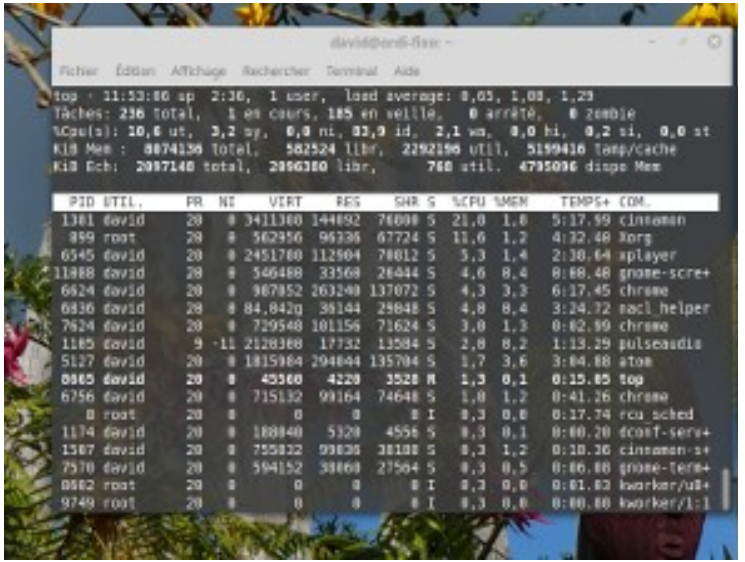


4- Architecture matérielle et systèmes d’exploitation

Des ordinateurs…

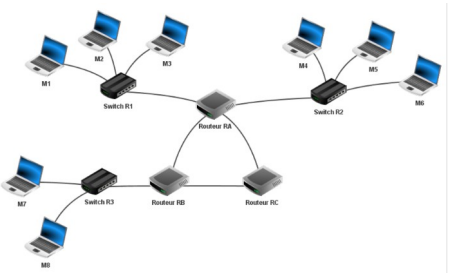
… aux systèmes sur puce

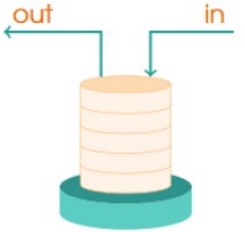
Gestion des ressources par un OS

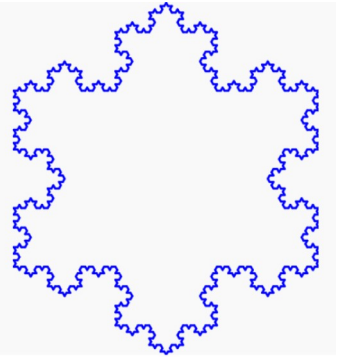


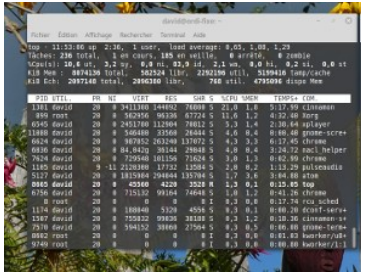
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) Architecture matérielle & systèmes d’exploitation
5) **Bases de données**
6) Algorithmique

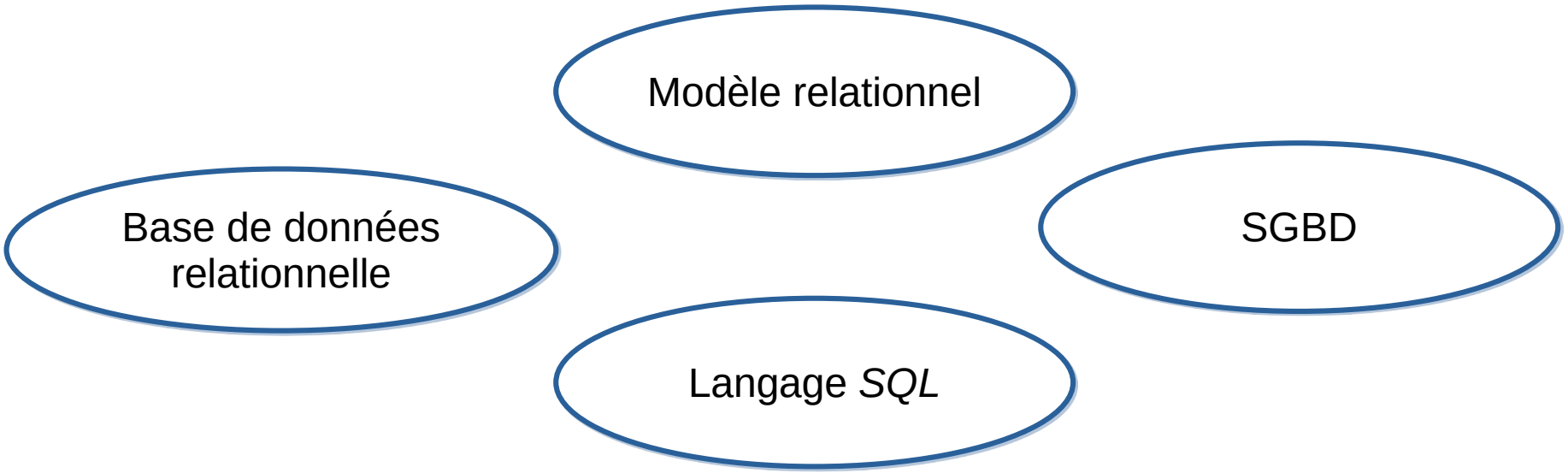






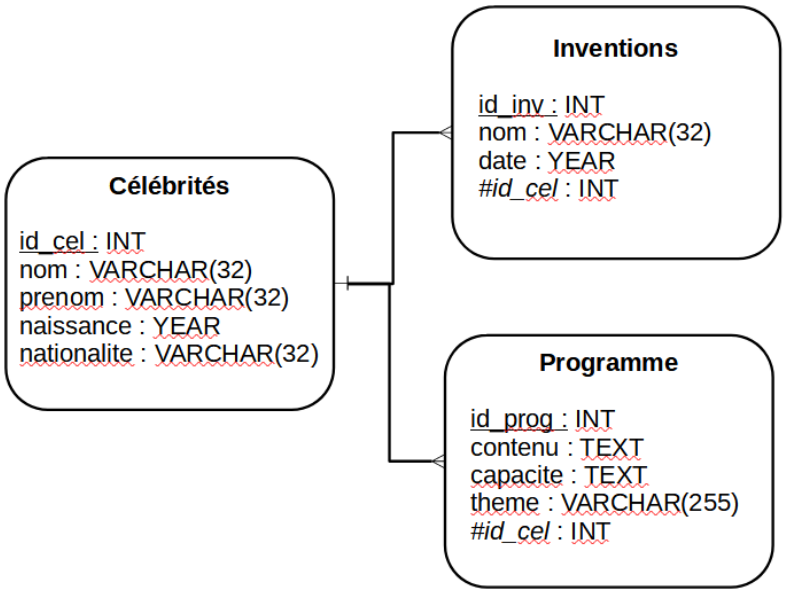


5- Bases de données

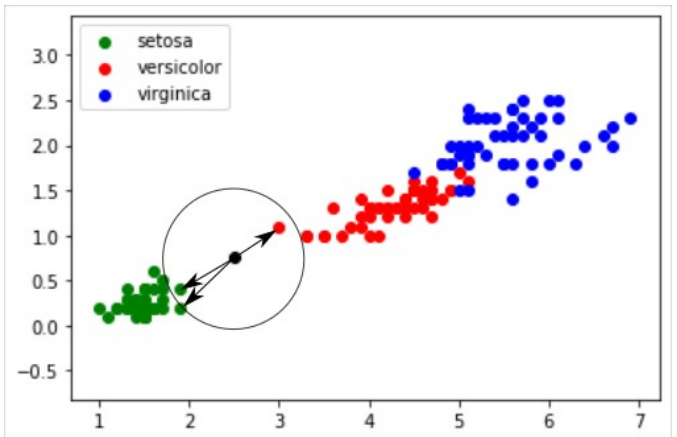


Projet bases de données & IA

Conception et implémentation d'une **base de données**

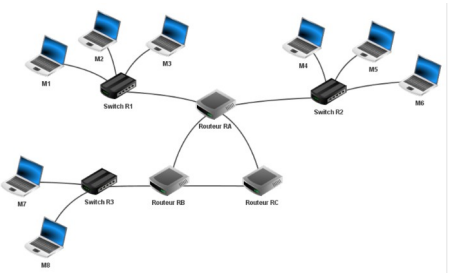


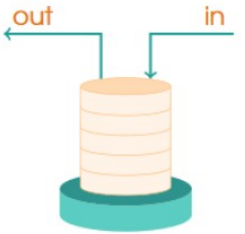
Implémentation d'un **algorithme d'IA** (k plus proches voisins, arbre de décision) pour faire des **prédictions**

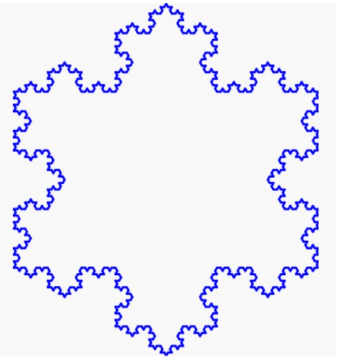


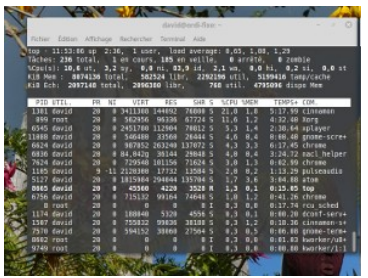
Programme

1) Programmation
2) Structures de données
3) Réseaux informatiques
4) Architecture matérielle & systèmes d’exploitation
5) Bases de données
6) **Algorithmique**









6- Algorithmique

Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

Programmation dynamique

Diviser pour régner

Recherche textuelle

Algorithmique sur les arbres, les graphes

6- Algorithmique

Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

Programmation dynamique

Diviser pour régner

Recherche textuelle

Algorithmique sur les arbres, les graphes

6- Algorithmique

Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

Programmation dynamique

Diviser pour régner

Recherche textuelle

Algorithmique sur les arbres, les graphes

6- Algorithmique

Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

Programmation dynamique

Diviser pour régner

Recherche textuelle

Algorithmique sur les arbres, les graphes

6- Algorithmique

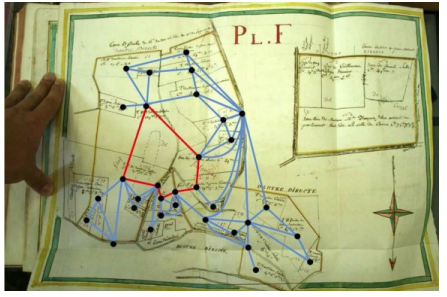
Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

Programmation dynamique

Diviser pour régner

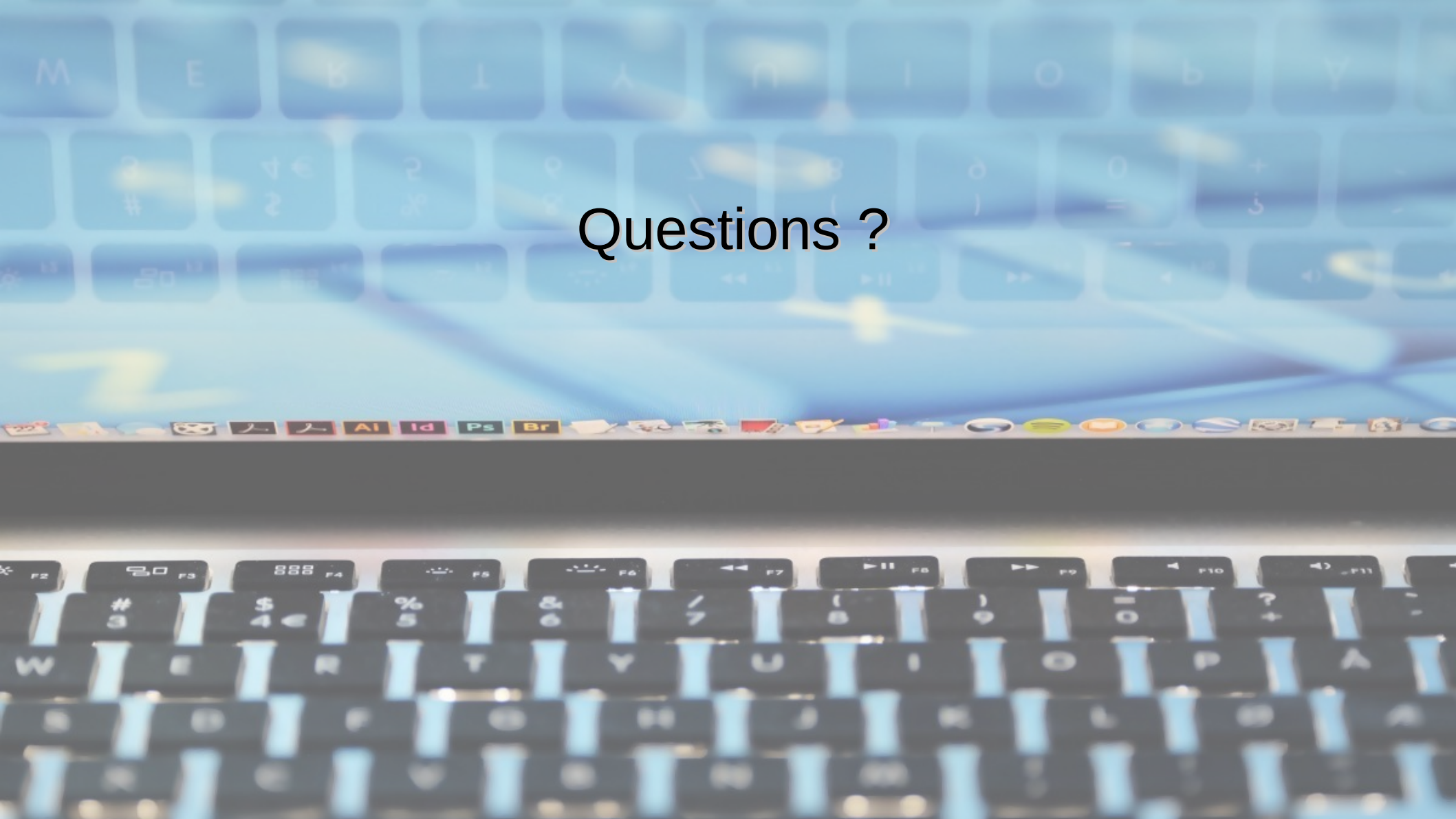
Recherche textuelle

Algorithmique sur les arbres, les graphes



Perspectives

- Profil **NSI + Maths/Physique/SVT** :
  - Licence scientifique
  - BUT (informatique, R et T, STID,...)
  - Écoles d’ingénieur post-bac
  - Classe préparatoire MP2I (Maths Tle + Physique 1ère)
- Profil **NSI + SES** :
  - Licence (sciences éco,…)
  - BUT (information communication, MMI, QLIO...)
  - École de commerce
- D’autres combinaisons sont possibles


Questions ?