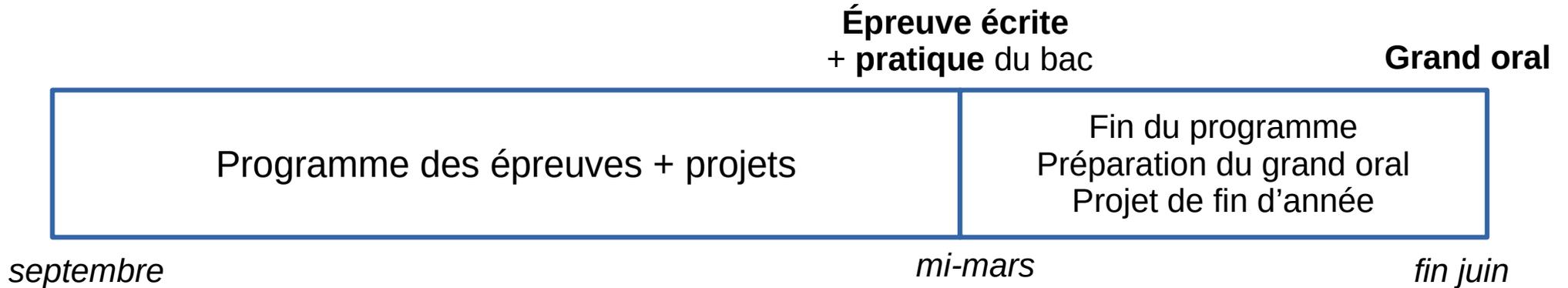


Programme de terminale NSI

L. TREBAUL

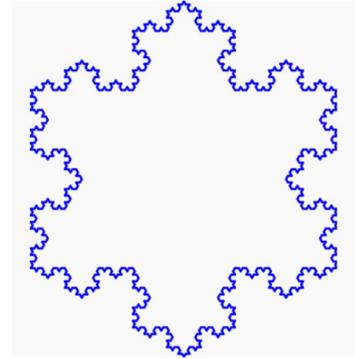
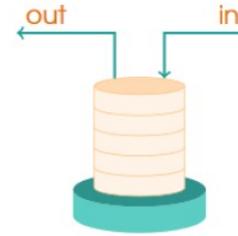
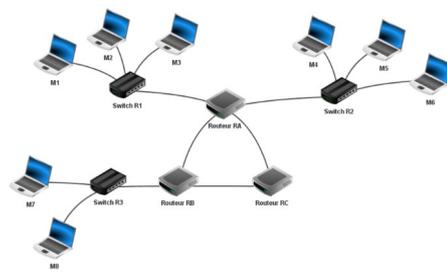
Organisation

- 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



```
dash@dash:~$ top
top - 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
Dashed: 280 total, 1 in state, 189 on wall-tick, 8 on hold, 8 running
VCalls: 10, 0 ut, 5, 2 sy, 6, 8 ni, 82, 9 id, 2, 1 wa, 0, 0 hi, 0, 2 si, 0, 0 st
kib Mem - 887428 total, 58224 free, 229228 mem, 338848 tmp/cache
kib Dch - 2097140 total, 2090260 lib, 746 util, 4795984 disp Mem

PID  UTML  PP  NI  VIRT  RES  SHR  S  CPU  MEM  TDPSSA  COM
1381  dev/d  20  0  911180 14482 74088  S  21.0  1.0  5.17 50  clistman
809  root   20  0  902960 98328 87724  S  11.6  1.2  4.32 80  nmap
1245  dev/d  20  0  243170 112884 78812  S  5.3  1.4  2.38 84  nglayer
1188  dev/d  20  0  50680 2156 24884  S  4.6  0.4  0.86 48  game-scrn
6824  dev/d  20  0  98392 263240 137072  S  4.3  3.3  6.17 45  chrome
1826  dev/d  20  0  843628 26144 2888  S  4.0  0.4  2.42 12  mc-helpt
7824  dev/d  20  0  72958 181158 71624  S  3.8  1.3  6.82 59  chrome
1120  dev/d  20  0  1422880 3772 12084  S  2.8  0.2  3.12 20  polystyle
5177  dev/d  20  0  1815884 28464 13784  S  1.7  3.4  3.44 48  straz
6860  dev/d  20  0  45580 4228 3284  R  1.9  0.1  6.15 45  top
9126  dev/d  20  0  153760 9416 7488  S  1.8  1.2  0.41 38  nmap
0  root   20  0  0  0  0  I  1.3  0.0  6.17 74  you-subst
1174  dev/d  20  0  188840 3228 4268  S  1.3  0.1  0.86 28  nmap-scrn
1287  dev/d  20  0  715822 99820 38188  S  1.3  1.2  1.18 30  clistman-ss
276  dev/d  20  0  39412 2168 2168  S  1.3  0.2  0.86 48  game-scrn
682  root   20  0  0  0  0  I  1.3  0.0  6.81 83  hawker/jst
276  root   20  0  0  0  0  I  1.3  0.0  0.86 88  hawker/jst
```



Programme

1) Programmation

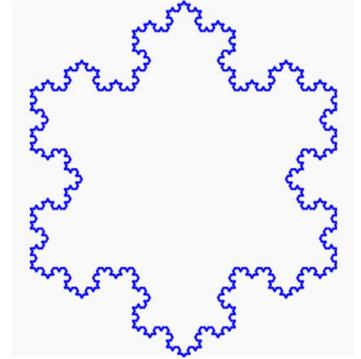
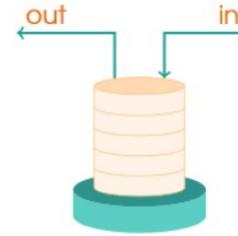
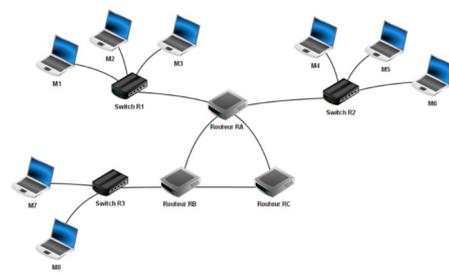
2) Structures de données

3) Réseaux informatiques

4) Architecture matérielle & systèmes d'exploitation

5) Bases de données

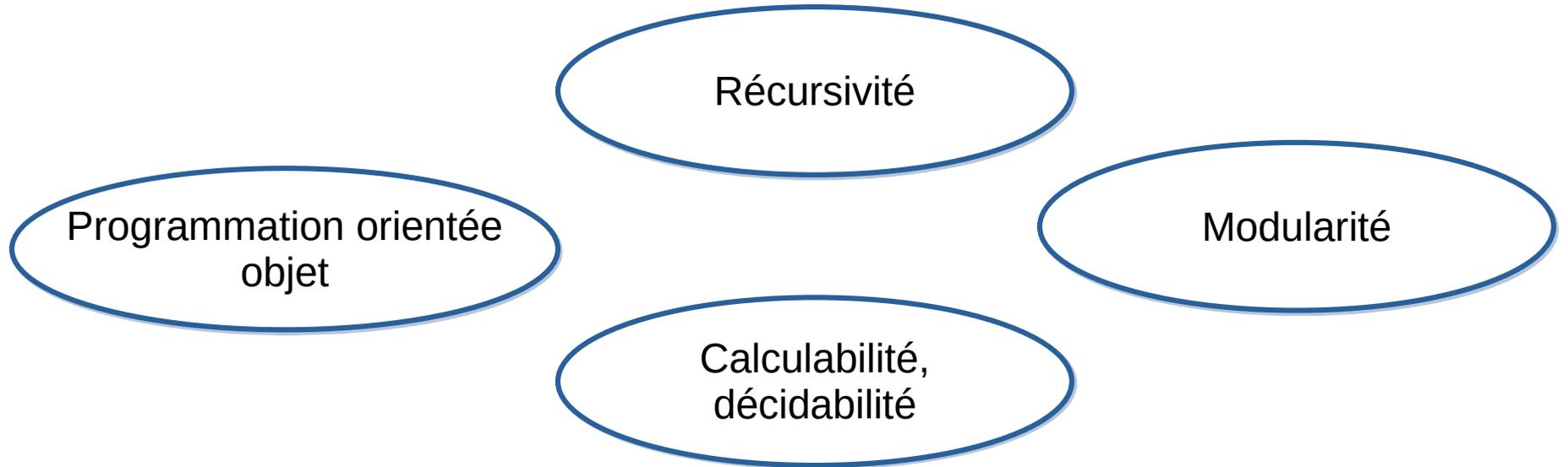
6) Algorithmique



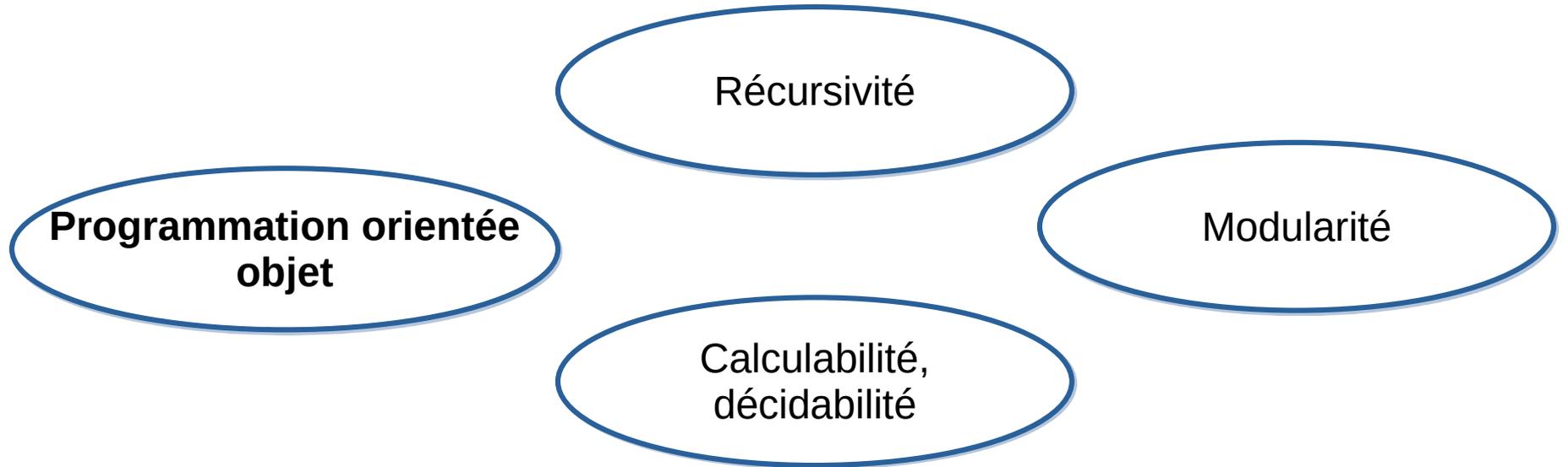
```
top - 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
Tasks: 280 total, 1 in state, 189 in wait, 8 sleep, 8 zombie
VmPeak: 30.8 ut, 5.2 st, 6.8 ni, 82.9 id, 2.1 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem : 387428 total, 58024 free, 239228 in use, 308648 swap free
KiB Dch : 209740 rxmb, 209030 txmb, 746 rxp, 479586 txpmb Mem
PID  UTML  PP  NI  UST  RES  SHR  S  CPU  MEM  TDPSS+  COM
1381  dev/d  20  0  341180 14482 74089  S  21.0 1.0  5.17 50  Cinnamon
809    root  20  0  90290  9839  8724  S  11.6 1.2  4.32 40  naup
1245  dev/d  20  0  243740 11284 7802  S  5.3 1.4  2.38 44  gplayer
1188  dev/d  20  0  50680  2766  2484  S  4.6  0.4  0.86 48  gnome-scre
1624  dev/d  20  0  98392  26349 13702  S  4.3 3.3  6.17 45  chrome
1836  dev/d  20  0  843420 3244  2888  S  4.0  0.4  2.42 72  nautilus
7824  dev/d  20  0  72958  18159 71624  S  3.8 1.3  8.62 59  chrome
1120  dev/d  20  0  442880 3772 1204  S  2.8  0.2  3.22 20  polkitd
5177  dev/d  20  0  185584 24664 13784  S  1.7 3.4  3.44 48  straz
1860  dev/d  20  0  45580  4228 3224  R  1.9 0.1  6.32 40  top
9756  dev/d  20  0  153740  464  7484  S  1.8 1.2  0.41 38  naup
0     root  20  0  0  0  0  I  0.3 0.0  8.17 74  you subcl
1174  dev/d  20  0  188480 3228  4556  S  0.3 1.2  0.86 28  cinnamon-va
1267  dev/d  20  0  755822 99820 38185  S  0.3 1.2  0.18 30  cinnamon-va
2716  dev/d  20  0  30412  3168  2764  S  0.3  0.2  0.86 48  gnome-scre
4882  root  20  0  0  0  0  I  0.3 0.0  0.81 83  hmc/hydrat
2740  root  20  0  0  0  0  I  0.3 0.0  0.86 88  hmc/hydrat(1)
```



1- Programmation



1- Programmation



Projets en Programmation Orientée Objet

Un jeu de bataille navale

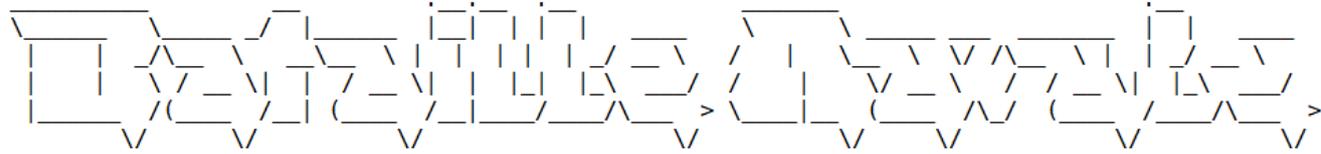
```
print("Placement du torpilleur (deux cases)")
while torpilleurUser.setPlace(x, y, direction) == False:
    while True:
        try:
            x = ligne[input("Entrez une ligne (entre A et J): ")]
            break
        except KeyError:
            x = ligne[input("Entrez une ligne (entre A et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

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): ")]
    while True:
```



```
[' ', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
['A', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['B', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['C', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['D', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['E', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['F', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['G', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['H', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['I', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
['J', '~', '~', '~', '~', '~', '~', '~', '~', '~', '~']
```

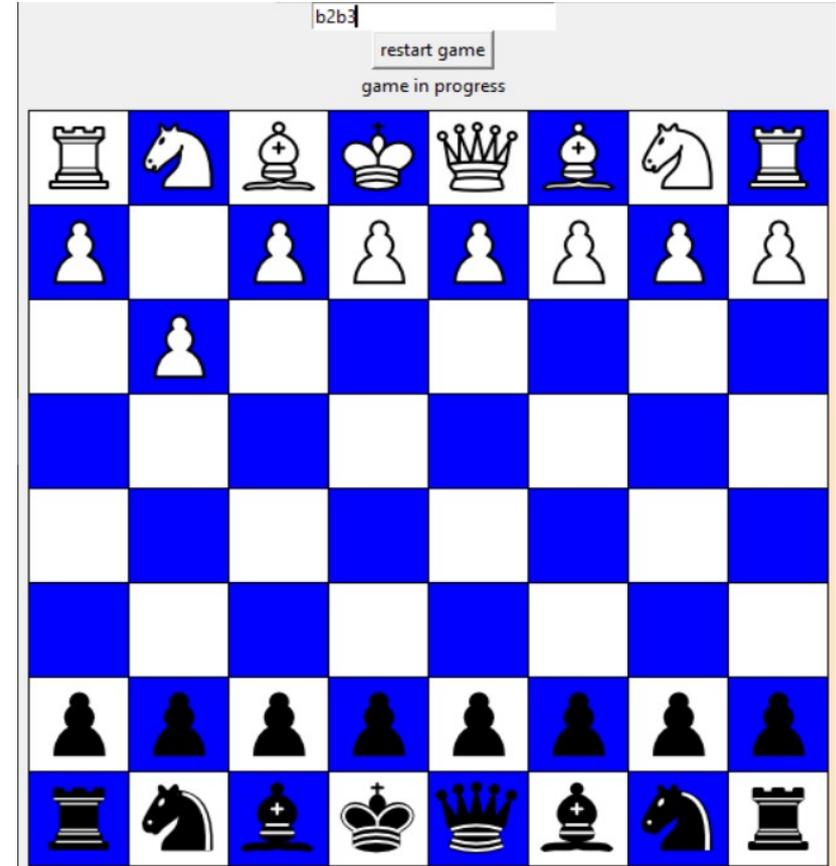
Placement du torpilleur (deux cases)

Entrez une ligne (entre A et J):

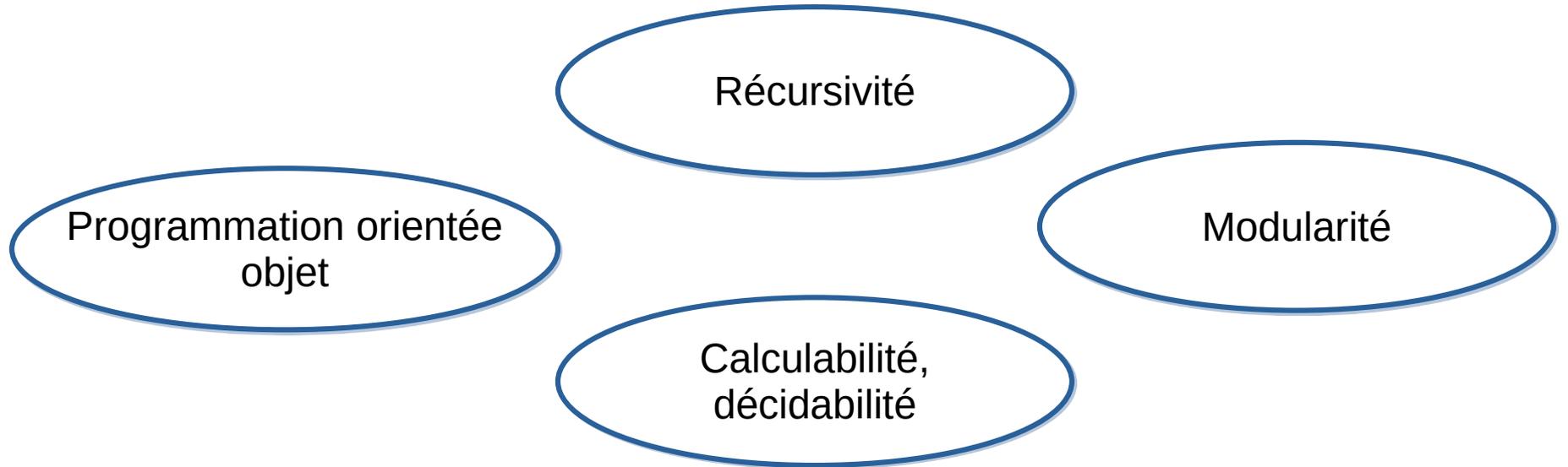
Projets en Programmation Orientée Objet

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

```
class Pawn(Piece):  
  
    def basepos(self): #on établit la position initiale 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
```



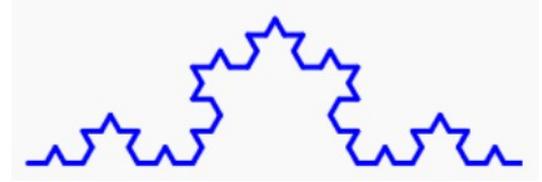
1- Programmation



1- Programmation



Les tours de Hanoï



Le flocon de von Koch

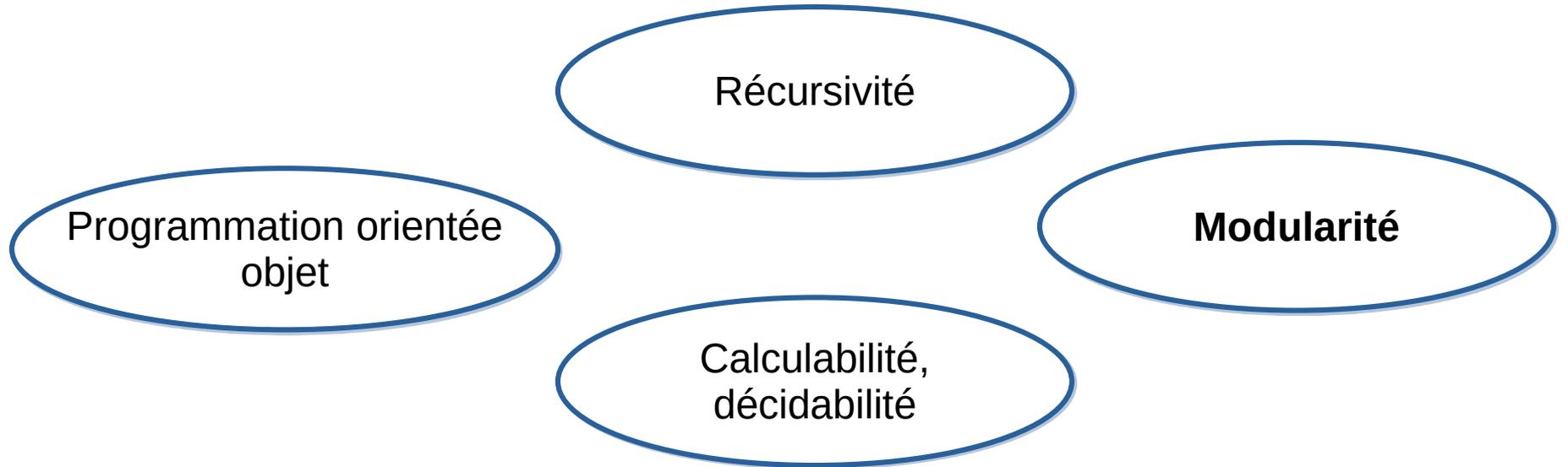
Récurtivité

Programmation orientée
objet

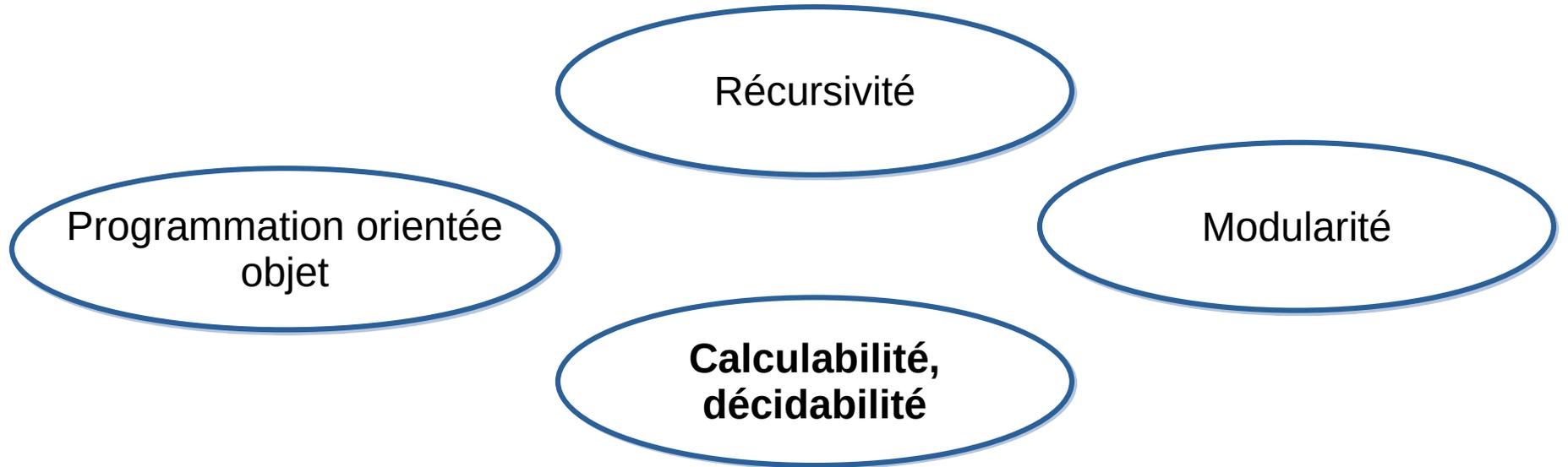
Modularité

Calculabilité,
décidabilité

1- Programmation

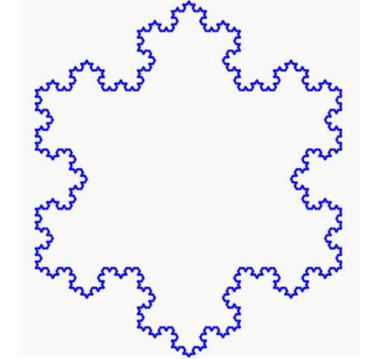
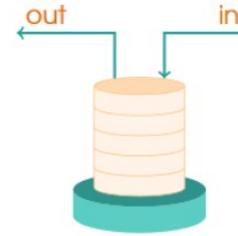
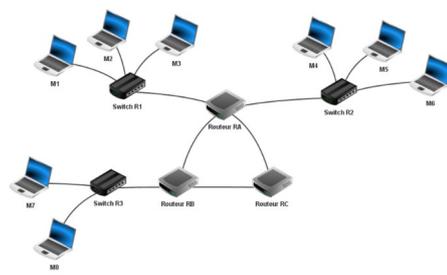


1- Programmation



Programme

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



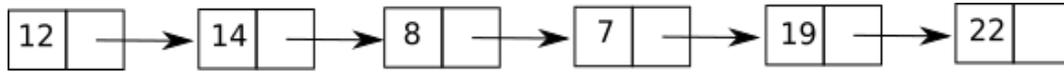
```
dash@dash:~$ top
top - 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
Tasks: 280 total, 1 in state, 189 in wait, 8 sleep, 8 zombie
VmStl: 30.8 ut, 5.2 st, 6.8 ni, 82.9 id, 2.1 wa, 0.0 hi, 0.2 si, 0.0 ot
KiB Mem : 387428 total, 58224 free, 239228 used, 338848 swapped
KiB Dch : 209740 total, 209030 lib, 746 util, 479586 disp Mem

PID  UTML  PP  NI  VIRT  RES  SHR  S  CPU  MEM  TYPSt+ COM
1381  dvrld  20  0  911180 14482 74088  S  21.0 1.0  5:17.90  clistman
809    root  20  0  902960 96328 67724  S  11.6 1.2  4:32.80  nmap
1245  dvrld  20  0  243740 112884 78812  S  5.3 1.4  2:38.84  nglayer
1188  dvrld  20  0  50680 2756 24884  S  4.6 0.4  0:46.48  gnome-scre
1624  dvrld  20  0  98392 26324 13702  S  4.3 0.3  6:17.45  chrome
1826  dvrld  20  0  843628 26344 28848  S  4.0 0.4  2:24.72  nucheller
7824  dvrld  20  0  729568 18158 71624  S  3.8 1.3  0:42.59  chrome
1120  dvrld  20  0  443628 3772 12064  S  2.8 0.2  3:12.20  polystere
5172  dvrld  20  0  185584 24664 13784  S  1.7 0.4  3:44.46  straz
1860  dvrld  20  0  45560 4228 3264  R  1.9 0.1  0:15.45  top
9756  dvrld  20  0  153760 7488 5  S  1.8 1.2  0:41.38  nmap
0    root  20  0  0 0 0  S  1.3 0.0  0:17.74  you-subst
1174  dvrld  20  0  168480 3228 4936  S  1.3 0.1  0:46.78  nucheller
1267  dvrld  20  0  75582 99820 38188  S  1.2 1.2  0:10.30  clistman+
2736  dvrld  20  0  30412 2168 2768  S  1.2 0.3  0:46.48  gnome-scre
1882  root  20  0  0 0 0  S  1.1 0.0  0:41.83  hmcarty/34
2740  root  20  0  0 0 0  S  1.0 0.0  0:46.88  hmcarty/31
```

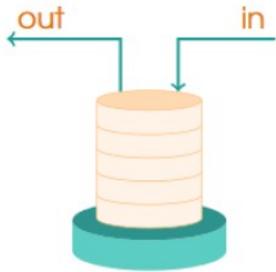


2- Structures de données

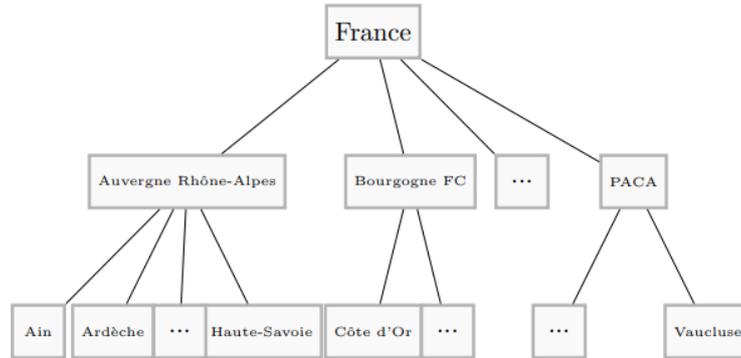
Comment agencer les données pour mieux répondre à des problèmes ?



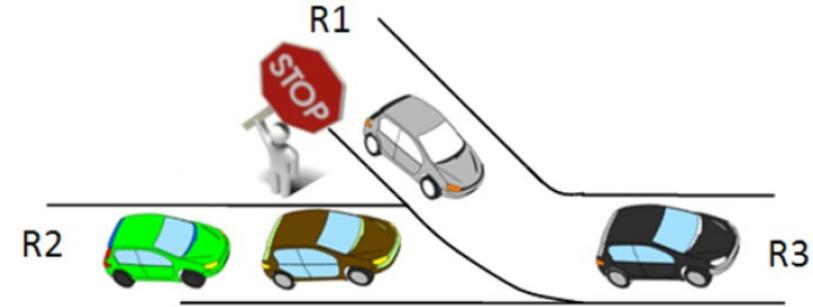
Liste chaînée



Pile



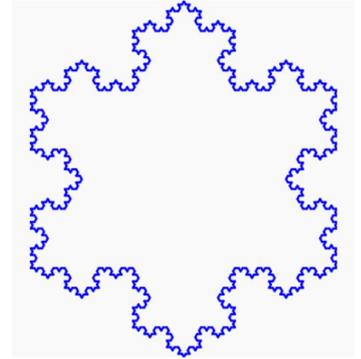
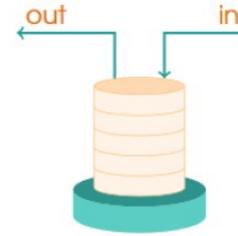
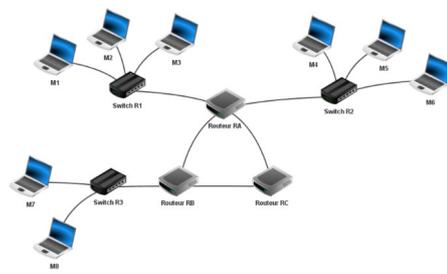
Arbre



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

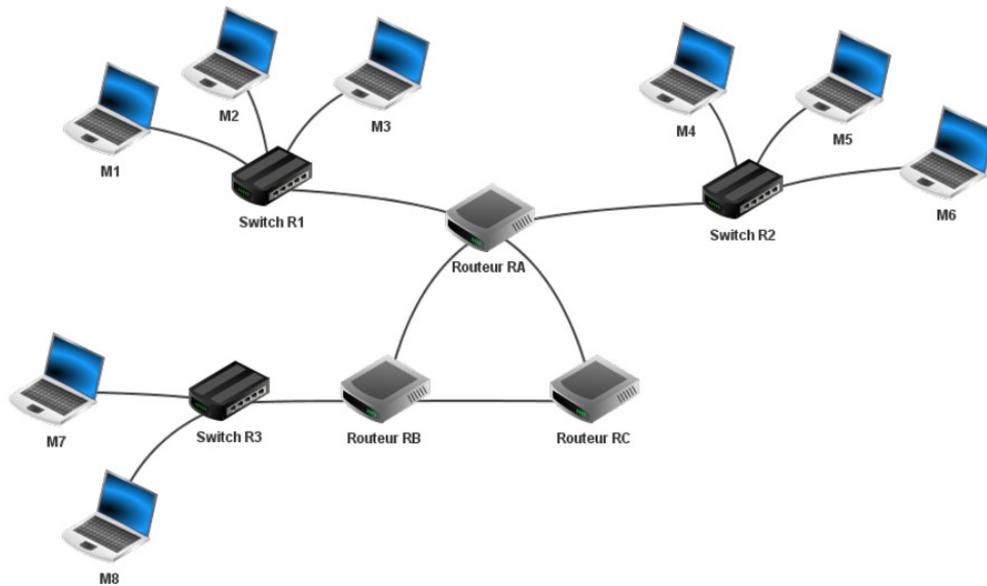


```
dash@dash:~$ top
top 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
Tasks: 288 total, 1 in state, 189 in wait, 8 sleep, 8 zombie
VmStall: 30.4 ut, 5.2 st, 6.8 ni, 82.9 id, 2.1 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem : 387428 total, 58024 free, 239228 in use, 304648 swap free
KiB Dch : 2097140 rxmb, 2090260 txmb, 746 rxib, 4795984 txmb Mem
PID  UTML  PP  NI  VIRT  RES  SHR  S  CPU  MEM  TTYSTA  CMD
1381  DaviD  20  0  911180 14480 74000  S  21.0 1.0  5:17.90  Cloudfann
809     root  20  0  902960 94928 47224  S  11.6 1.2  4:52.80  nmap
1245  DaviD  20  0  2453760 112384 78032  S  5.3 1.4  2:38.84  nglayer
1188  DaviD  20  0  506880 27568 24884  S  4.6 0.4  0:46.48  gnome-scre
1624  DaviD  20  0  983952 463248 137072  S  4.3 3.3  6:17.45  Chrome
1836  DaviD  20  0  843424 32448 28848  S  4.0 0.4  0:24.72  nvidia-lan
7424  DaviD  20  0  729584 181156 71624  S  3.8 1.3  6:42.59  Chrome
1120  DaviD  20  0  225888 37728 12064  S  2.8 0.2  0:12.20  gnome-stre
5177  DaviD  20  0  185584 24664 15704  S  1.7 0.4  3:44.46  straz
1880  DaviD  20  0  45560 4228 3528  R  1.9 0.1  0:15.45  top
9756  DaviD  20  0  155360 9488 7488  S  1.8 1.2  0:41.35  nmap
0     root  20  0  0 0 0  I  1.3 0.0  0:17.74  you subcl
1174  DaviD  20  0  168840 3228 4956  S  1.3 0.1  0:46.78  nvidia-lan
1287  DaviD  20  0  755822 99820 38188  S  1.2 1.2  0:10.30  Cloudfann-vt
2736  DaviD  20  0  304120 1888 2768  S  1.2 0.2  0:46.48  gnome-Schre
4882  root  20  0  0 0 0  I  1.2 0.0  0:41.83  hmc/hmc/vst
2740  root  20  0  0 0 0  I  1.0 0.0  0:46.88  hmc/hmc/vst
```



3- Réseaux informatiques

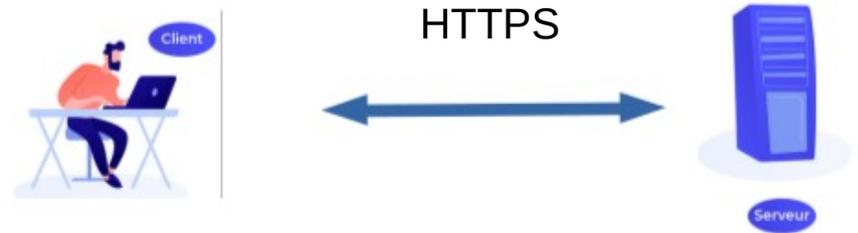
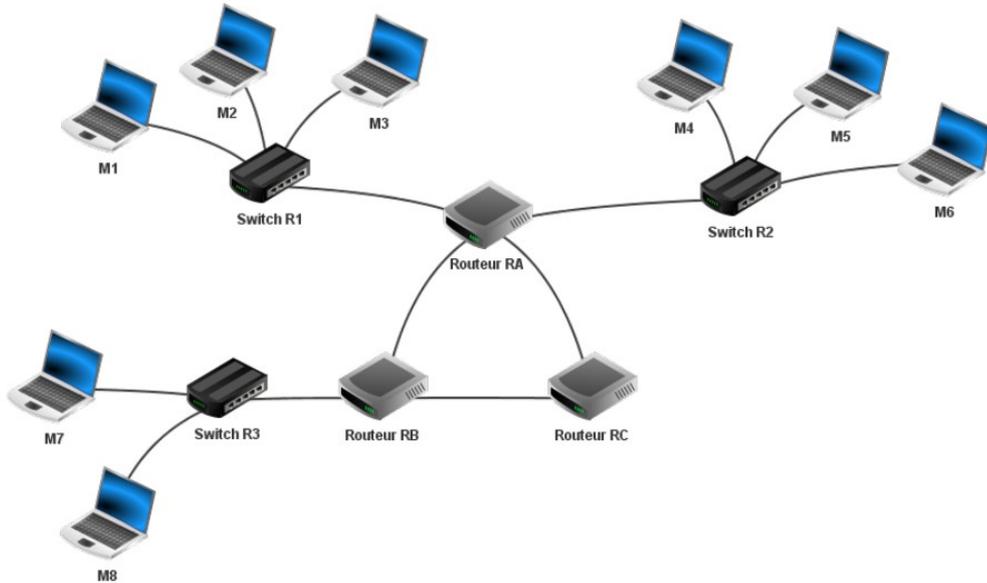
Comment *guider* les paquets dans un réseau ?



3- Réseaux informatiques

Comment *guider* les paquets dans un réseau ?

Comment *sécuriser* l'échange d'informations ?

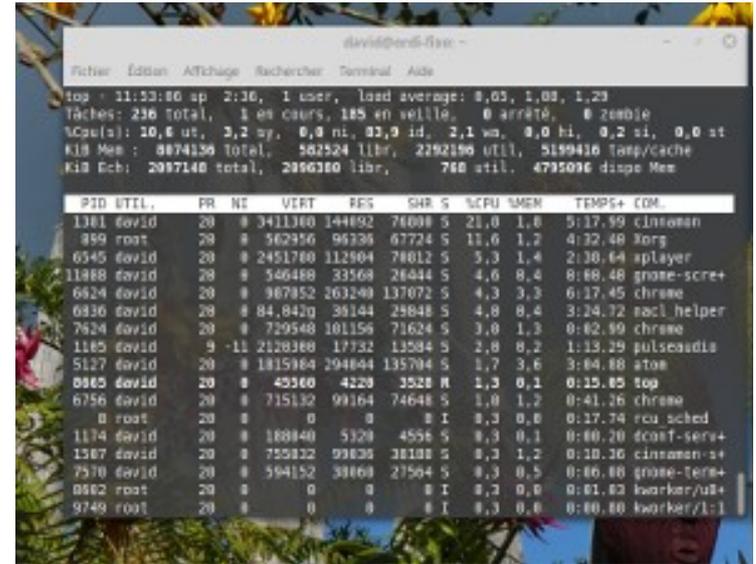


4- Architecture matérielle et systèmes d'exploitation



Des ordinateurs...

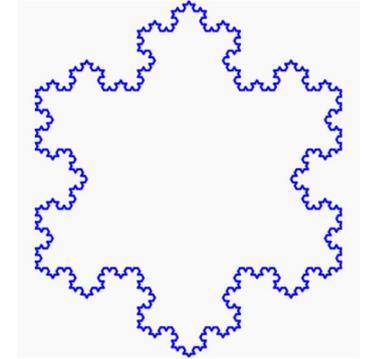
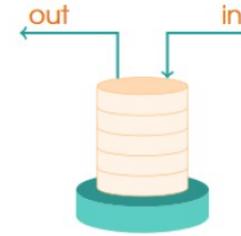
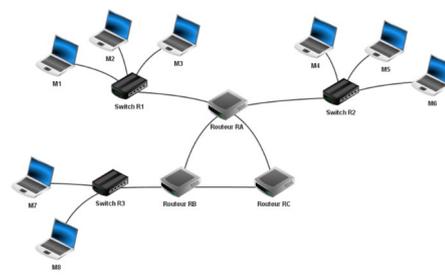
Gestion des ressources par un OS



... aux systèmes sur puce

Programme

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

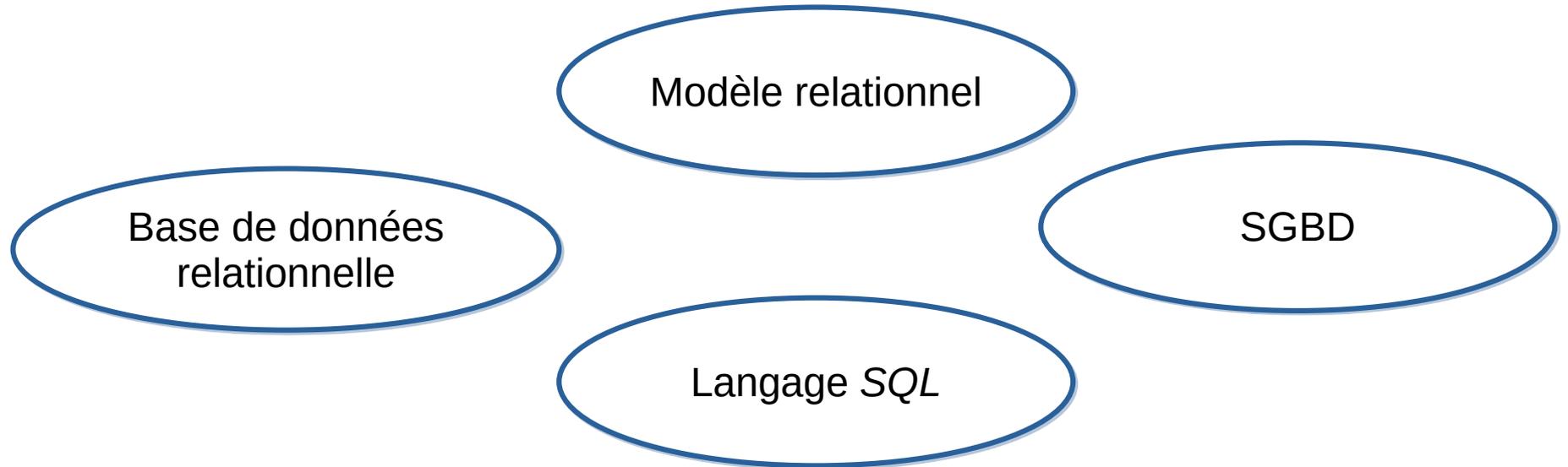


```
top 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
taskset: 200 users, 1 on core, 189 on walltime, 8 on CPU, 8 on mem
VCPU(s): 30, 4 ut, 5, 2 by, 6, 8 nl, 82, 9 id, 2, 15 wa, 8, 0 hi, 8, 2 si, 0, 0 st
KIB Mem - 187428 Total, 58124 Mem, 229228 Htt, 339848 Temp/Active
KIB Dch - 2097140 total, 2090260 lib, 746 util, 4795984 disp Mem
```

PID	UTL	PR	NI	UID	PPID	SEC	USR	S	SHR	MEM	TEMP	COM	
1381	devl0	20	0	3411380	14482	74089	5	21	0	1,0	5,17	50	Clustern
809	root	20	0	342950	94328	87724	5	11	0	1,2	4,32	80	borg
1245	devl0	20	0	2453740	112384	78012	5	3,3	1,4	2,38	84	ngltyper	
1188	devl0	20	0	356826	27568	24884	5	6,5	8,4	0,86	48	game-scrn	
6824	devl0	20	0	987932	263248	137072	5	4,3	3,3	6,17	45	chrome	
1826	devl0	20	0	843428	26344	28848	5	4,8	8,4	0,24	72	msl-helpr	
7424	devl0	20	0	729588	181158	71624	5	3,8	1,3	8,62	59	chrome	
1120	devl0	20	0	1422888	37728	12064	5	4,8	8,2	0,22	20	python3.6	
5177	devl0	20	0	1815884	246464	13704	5	1,7	3,4	3,44	48	atom	
1840	devl0	20	0	10510	4228	32824	8	1,9	0,1	8,15	45	top	
9756	devl0	20	0	153740	94328	74848	5	1,8	1,2	8,41	38	atom	
0	root	20	0	0	0	0	0	0	0	0	0	0	you-subst
1174	devl0	20	0	188848	3228	4268	5	0,3	0,1	8,16	28	atom-scrn	
1287	devl0	20	0	715822	99820	38188	5	0,3	1,2	8,18	30	Clustern-vt	
2736	devl0	20	0	304122	3168	2768	5	0,3	0,2	8,16	48	game-Schne	
4882	root	20	0	0	0	0	0	0	0	0	0	0	banker/vst
3740	root	20	0	0	0	0	0	0	0	0	0	0	bank/vst

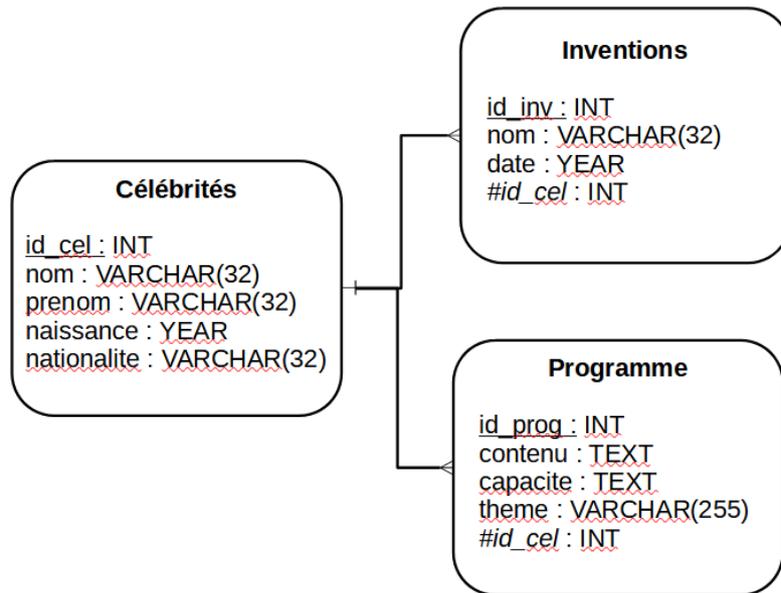


5- Bases de données

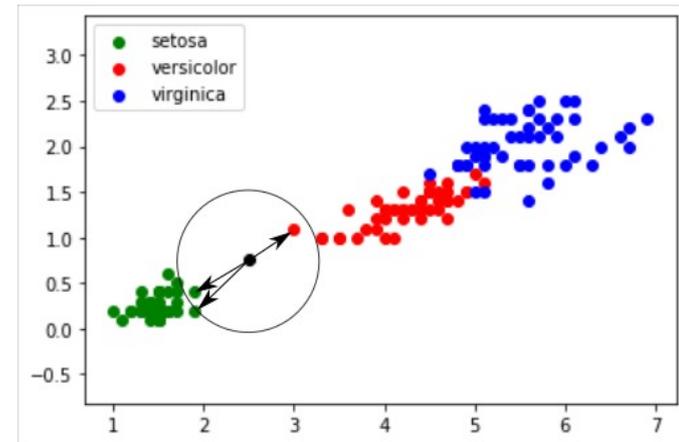


Projet bases de données & IA

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

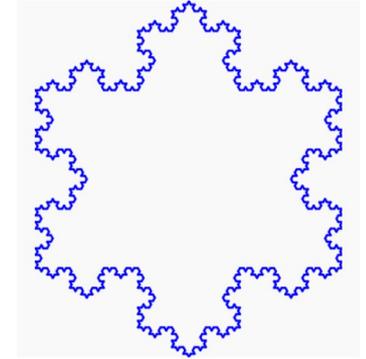
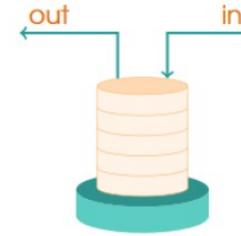
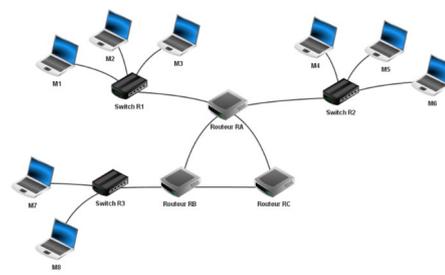


Implémentation d'un **algorithme d'IA** (k plus proches voisins, arbre de décision) pour faire des **prédictions**



Programme

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



```
dash@dash:~$ top
top - 11:33:00 up 2:16, 1 user, load average: 0.03, 0.08, 1.29
Tasks: 280 total, 1 in state, 189 in wait, 8 sleep, 8 zombie
VmStall: 10.4 ut, 5.2 st, 6.8 ni, 82.9 id, 2.1 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem : 387428 total, 58024 free, 239228 in use, 308648 swap free
KiB Dch : 2097140 rxmb, 2090260 txmb, 746 rxit, 4795966 dsmg Mem

PID  UTML  PP  NI  VIRT  RES  SHR  S  MEM%  USER  TTYSTA  CMD
1381  dvrld  20  0  3411360 144802 74088  S  21.0 1.0  5:17:50  clistman
809    root   20  0  302560 94928  8724  S  11.6 1.2  4:52:40  nmap
1245  dvrld  20  0  2453760 112384 78032  S  5.3 1.4  2:38:44  nglayer
1188  dvrld  20  0  506880 27568 24884  S  0.6 0.4  0:06:48  gnome-scre
1624  dvrld  20  0  987952 263248 137072  S  4.3 3.3  6:17:45  chrome
1826  dvrld  20  0  843424 26144 28848  S  0.8 0.4  2:24:12  ncolfilter
7824  dvrld  20  0  729584 181158 71624  S  3.8 1.3  6:42:59  chrome
1120  dvrld  20  0  222880 37728 12064  S  2.8 0.2  3:12:29  polkitd@1
5177  dvrld  20  0  181584 24664 13784  S  1.7 3.4  3:44:46  straz
1860  dvrld  20  0  45560 4228 3528  R  1.9 0.1  6:15:45  top
9756  dvrld  20  0  153760 94928 74088  S  11.6 1.2  4:51:38  nmap
0      root   20  0  0 0 0  S  0.3 0.0  6:17:74  you subcl
1174  dvrld  20  0  168840 3228 4936  S  0.3 1.2  6:06:28  ncolfilter
1267  dvrld  20  0  715824 93820 38188  S  4.3 1.2  0:10:30  clistman-1
2736  dvrld  20  0  304172 3168 2768  S  0.3 0.2  0:00:48  gnome-scre
1882  root   20  0  0 0 0  S  0.3 0.0  6:41:43  hmc/hmc/141
2740  root   20  0  0 0 0  S  0.3 0.0  6:18:48  hmc/hmc/141
```



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

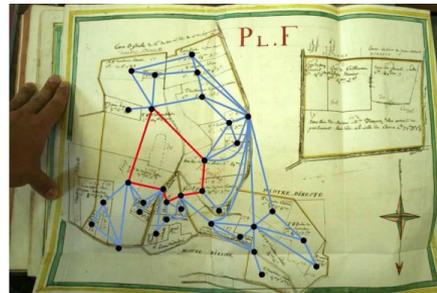
Comment trouver l'algorithme *le plus efficace* pour résoudre un problème ?

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 ?