Bazar à la boulangerie.

[Année 2014- 2015]

Barillot Thomas, Bridonneau Vincent, Dufaure Ludovic, Gonzalez Benjamin, Martin Jérémy, Maureau Clément et Sannier Yoann **du lycée Alfred Kastler de Talence**

Anninos Nathanael, , Bjakhov Inal, Dupuy Cassandre, Duret Guillaume, Guerin Elisabeth, Puissant Ines, Tandonnet Romane et Yassine Aïcha **du lycée Vaclav Havel de Bègles**

Chercheur : Adrien Boussicault, du LABRI (Bordeaux)
Enseignants : Guillaume Boix, Marie-Josée Denaes, Cathy Racadot et Corinne Ribrault.

Présentation du sujet

Aujourd'hui la boulangerie est pleine. Tous les matins, à l'ouverture de la boulangerie, il y a de nombreuses personnes qui attendent pour prendre du pain. Depuis une heure, les gens sont arrivés les uns après les autres et se sont agglutinés devant l'entrée.
Malheureusement, ils n'ont pas fait une unique file d'attente parfaitement ordonnée. Chaque personne, au moment d'arriver, a repéré, dans tout ce capharnaüm, un groupe de personnes et considère qu'elle doit passer après ce groupe. Ces relations peuvent être représentées par un graphique contenant des cercles reliés par des flèches. Chaque cercle représente une personne de la boulangerie. On met une flèche entre deux personnes si la personne à l'origine de la flèche a repéré que la personne pointée par la flèche devait passer avant elle. On appelle ce dessin un graphe.

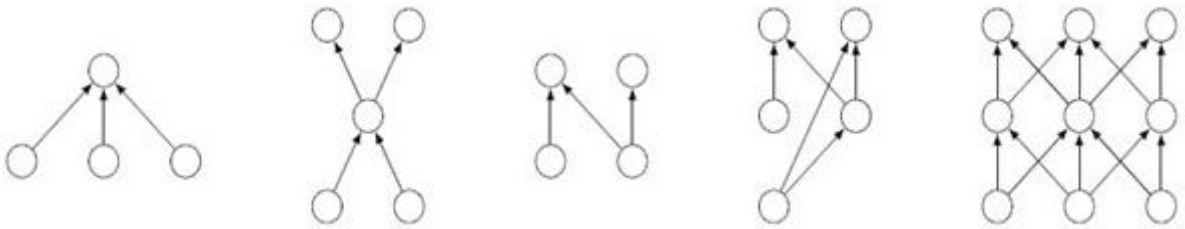


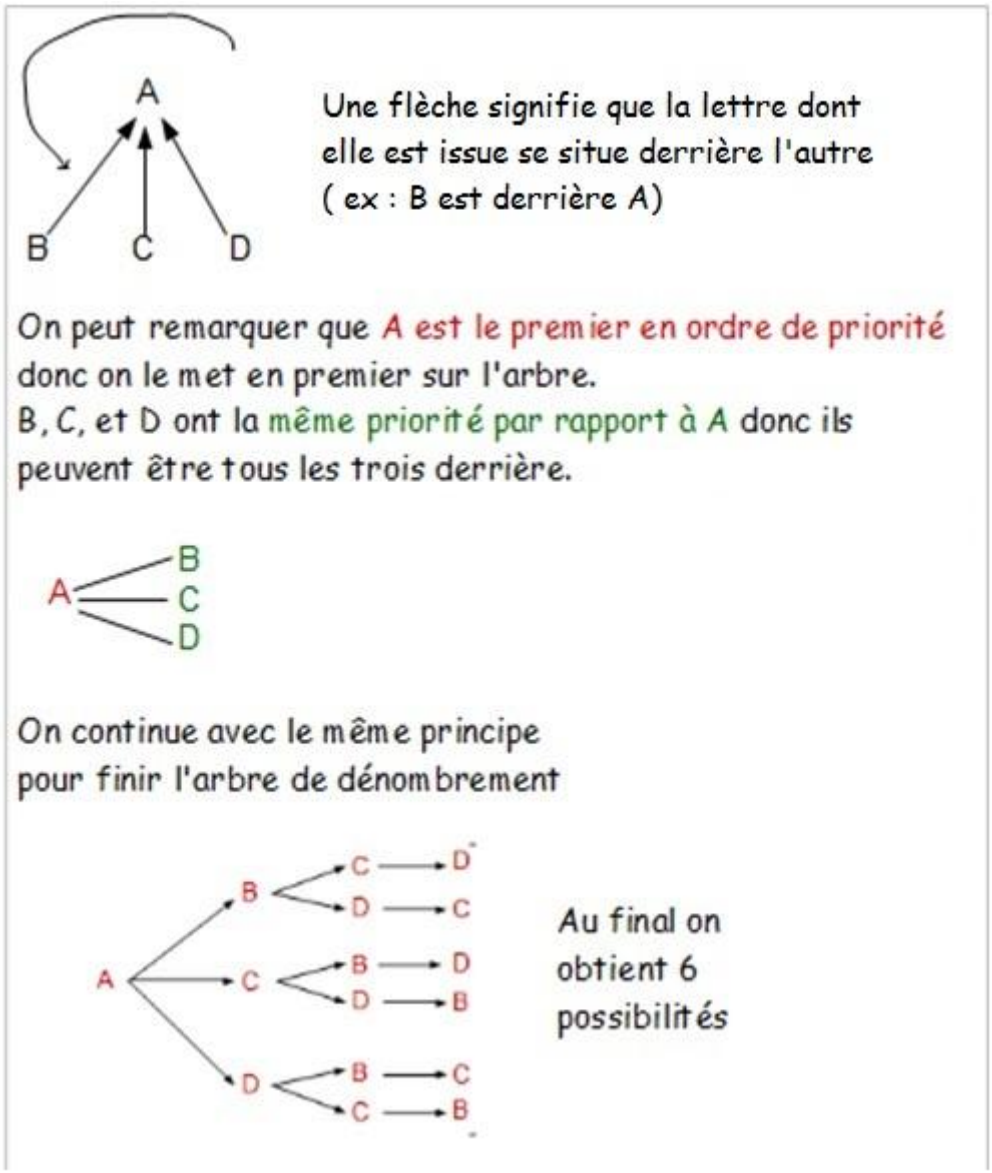
Figure 1 – Cinq exemples de files d'attente mal ordonnées

Par exemple, la figure 1 montre cinq exemples de files d'attentes mal ordonnées. Ainsi, le premier travail consiste à ordonner les personnes en respectant les contraintes imposées par les gens.

.

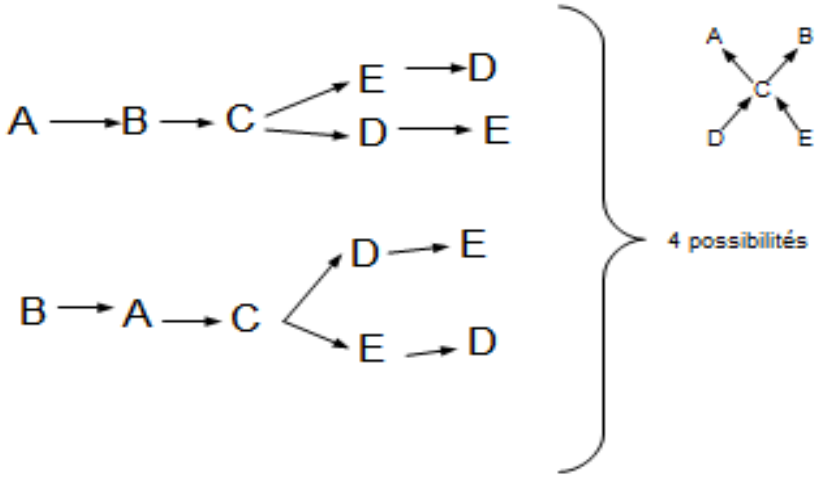
Première partie

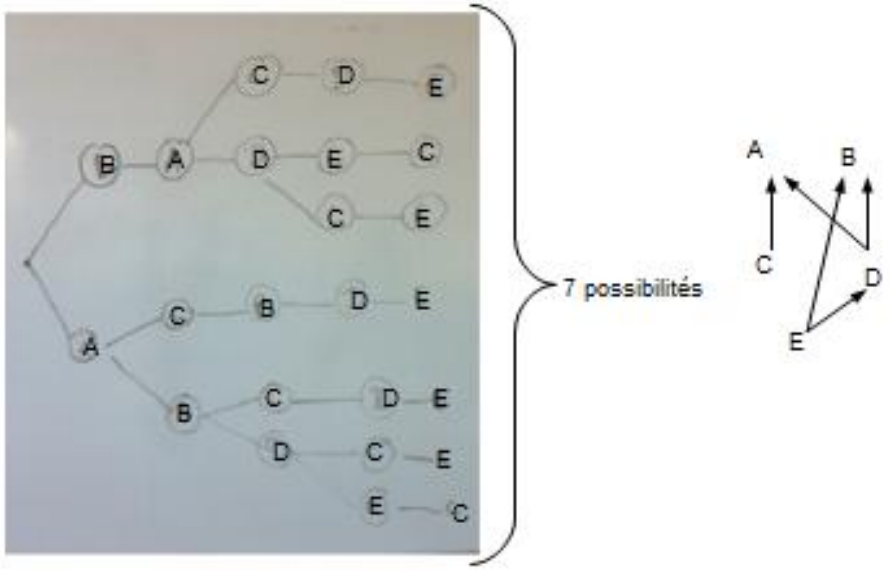
Notre groupe a tout d'abord essayé de trouver le nombre de possibilités dans les files données. Pour cela nous avons construit des arbres de dénombrement :
On obtient ainsi toutes les possibilités pour le premier graphe proposé,

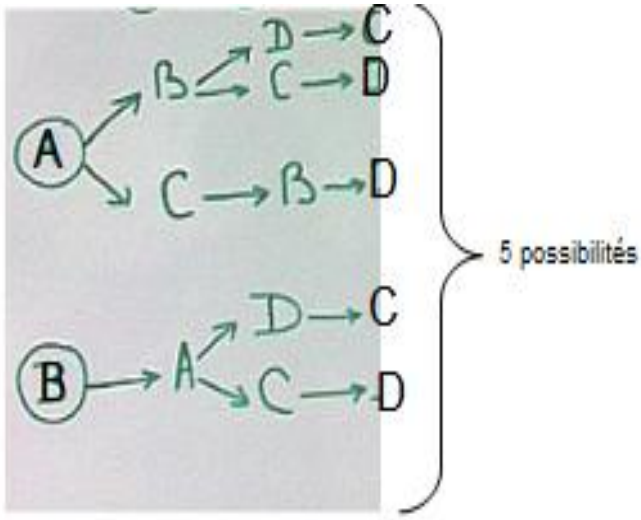


Nous avons procédé de ma même façon pour dénombrer toutes les possibilités pour les trois graphes suivants,

Arbre de dénombrement :

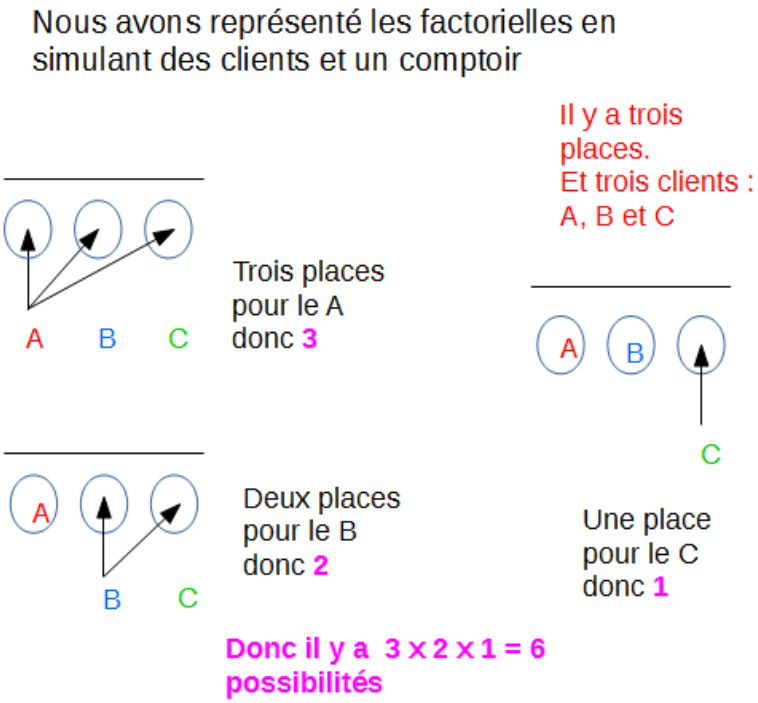




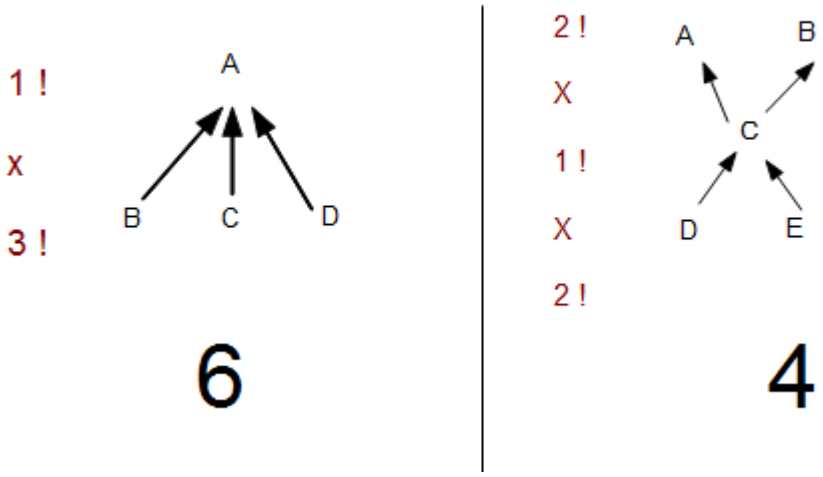


On a ensuite cherché une méthode de calcul avec laquelle on pourrait dénombrer le nombre de possibilités.

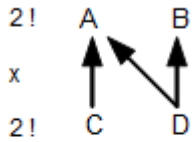
On s'est aperçu que cela faisait intervenir les factorielles.
N! est le produit de tous les entiers de 1 jusqu'à N.
Ex: 1! = 1
2! = 2x1
3! = 3x2x1
4! =4x3x2x1
5! = 5x4x3x…..
Pour trouver le nombre de possibilités, on compte le nombre de points sur chaque ligne, supposons que ce nombre soit n. Ensuite on prend les factorielles de tous les nombres n trouvés à chaque ligne et on les multiplie entre eux. (1)



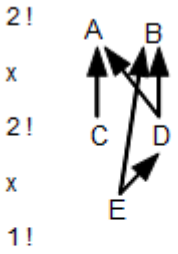
Et on constate que cela fonctionne pour les deux cas suivants :



Mais cela ne fonctionne pas dans tous les cas, Par exemple :

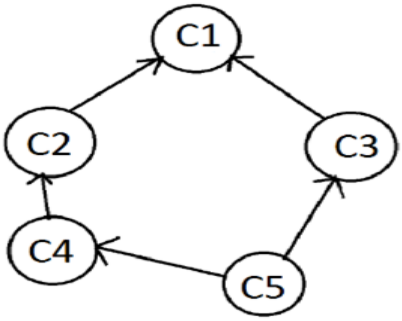


4 différent de 5



4 différent de 7

Nous avons essayé d'écrire un algorithme pour automatiser le travail de dénombrement et nous avons réfléchi à partir de ce graphe.



	C1	C2	C3	C4	C5	Total
C1	0	0	0	0	0	0
C2	1	0	0	0	0	1
C3	1	0	0	0	0	1
C4	1	1	0	0	0	2
C5	1	1	1	1	0	4

Nous avons rempli un tableau de cette façon :

Si le client i est derrière le client j, on met 1 dans la cellule (i;j),
sinon on met 0.

A partir de ces donnés nous avons écrit l'algorithme suivant mais nous n'avons pas réussi à le programmer.

Entrée :

- Saisir tous les couples (i ; j)

- Traitement :
- Pour i allant de 1 à 5

  si $\sum_{j=1}^{5}(i;j)=0$

  créer liste avec Ci en premier
- Réitérer le même calcul après avoir enlevé

  la colonne et la ligne de Ci

Sortie :

- Afficher toutes les listes

(2)

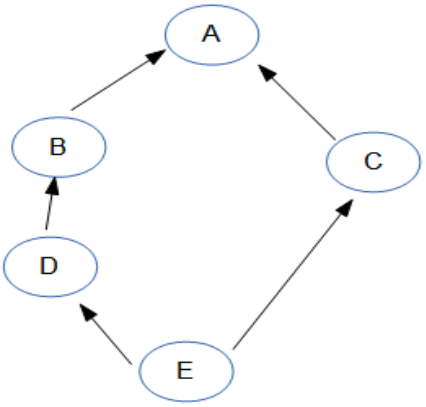
Seconde partie

La deuxième problématique posée était de trouver **un ordre de passage** des clients pour aider le boulanger.

Première façon de traiter cette problématique.

On nomme chaque client par une lettre ; ensuite on note les liens entre les personnes dans un tableau.

	A	B	C	D	E
A	0	0	0	0	0
B	1	0	0	0	0
C	1	0	0	0	0
D	0	1	0	0	0
E	0	0	1	1	0



(3)

À partir de ce tableau on peut calculer les autres tableaux qui donnent le nombre de chemins de longueur 2 ; 3 ; 4 ; … à l'aide de matrices.

Matrice des chemins de longueur 1

	A	B	C	D	E
A	0	0	0	0	0
B	1	0	0	0	0
C	1	0	0	0	0
D	0	1	0	0	0
E	0	0	1	1	0

Matrice des chemins de longueur 2

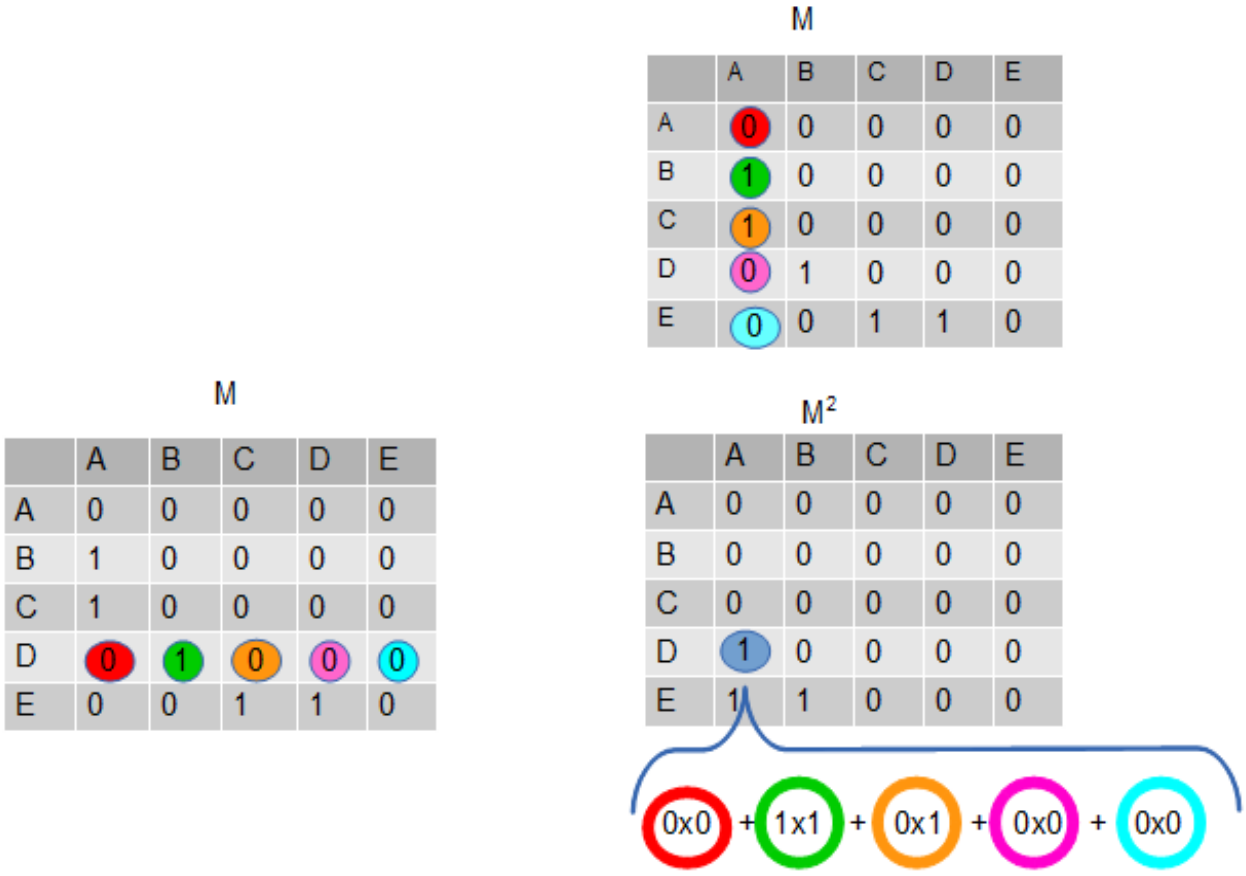
	A	B	C	D	E
A	0	0	0	0	0
B	0	0	0	0	0
C	0	0	0	0	0
D	1	0	0	0	0
E	1	1	0	0	0

(3bis)

Matrice des chemins de longueur 3

	A	B	C	D	E
A	0	0	0	0	0
B	0	0	0	0	0
C	0	0	0	0	0
D	0	0	0	0	0
E	1	0	0	0	0

Pour calculer un élément du tableau représentant les chemins de longueur 2 on prend le tableau des chemins de longueur 1 pour en déduire le tableau des chemins de longueur 2 comme montré dans le schéma ci-dessous :



En additionnant tous les nombres contenus dans chaque ligne de chaque matrice on obtient le tableau suivant nous donnant les ordres de priorité maximale de chaque client, et ainsi en les rangeant par ordre croissant on obtient un ordre de passage possible.

(4)

	M1	M2	M3	Ordre de priorité maximale
A	0	0	0	0
B	1	0	0	1
C	1	0	0	1
D	1	1	0	2
E	2	2	1	5

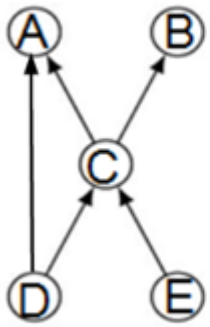
Le boulanger a donc dans ce cas précis deux ordres de passage possibles :

A B C D E ou A C B D E

Seconde façon de traiter cette problématique.

I) Tableau des profondeurs :

Profondeur de D vers A : longueur du plus long chemin allant de D vers A



	A	B	C	D	E	PRIORITE (Total)
A		0	0	0	0	0
B	0		0	0	0	0
C	1	1		0	0	2
D	2	2	1		0	5
E	2	2	1	0		5

Conjecture : En ordonnant les clients par ordre croissant de priorité on obtient une file d'attente totalement ordonnée qui respecte toutes les contraintes

Une proposition de file totalement ordonnée : A B C E D

II) Simplification d'un graphe par suppression des flèches inutiles

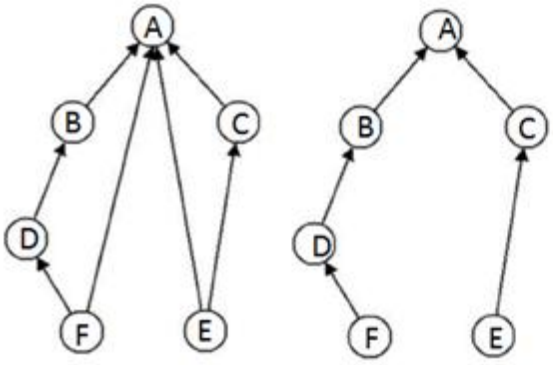


Tableau des profondeurs :

	A	B	C	D	E	F	PRIORITE (Total)
A	0	0	0	0	0	0	0
B	1	0	0	0	0	0	1
C	1	0	0	0	0	0	1
D	2	1	0	0	0	0	3
E	2	0	1	0	0	0	3
F	3	2	0	1	0	0	6

Une solution possible **: A B C E D F**

Une autre solution possible : A B D F C E

Priorités: 0 1 3 6 1 3

Cette solution est tout à fait satisfaisante pourtant elle ne respecte pas l'ordre croissant des priorités.

Cela signifie que notre conjecture est une condition suffisante pour trouver une solution mais que ce n'est pas une condition nécessaire.

III) Situation ne pouvant pas avoir de solution.

Dans le cas d'une boucle, comme présenté ci-dessous, il est impossible de faire passer quelqu'un en premier sans contredire une flèche.

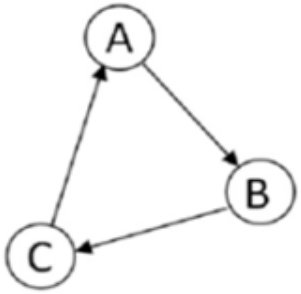


Tableau des profondeurs :

	A	B	C	PRIORITE (Total)
A	3	1	2	6
B	2	3	1	6
C	1	2	3	6

On remarque que les clients ont une liaison vers eux-mêmes de la longueur correspondante au nombre de clients dans la boucle. Cela se voit dans le tableau car des valeurs non nulles (des 3) sont apparues sur la diagonale.

IV) Réalisation d'un algorithme :

Nous avons ensuite élaboré un algorithme qui résout le problème tout seul.
Dans l'algorithme on a traité séparément les chemins directs (chemins de longueur 1 soit l'équivalent d'une flèche) et les chemins indirects (chemins de longueur supérieure à 1).

Chemin direct :

Si le chemin n'est pas déjà de longueur >1
. Le chemin est de longueur 1
Fin si

Chemin indirects :

Pour i de 1 à (nombre de clients)
. Pour h de 1 à (nombre de clients)
. . Si le chemin du client i vers le client h est direct alors
. . . Pour j de 1 à (nombre de clients)
. . . . Si le chemin du client h vers le client j est différent de 0 et que sa longueur +1
. . . . est supérieure à la longueur du chemin du client i vers le client j alors
. . . . . La longueur du chemin du client i vers le client j est égale à la
. . . . . longueur du chemin du client h vers le client j+1
. . . . Fin si
. . . Fin pour
. . Fin si
. Fin pour
Fin pour

Extraits du programme réalisé en Visual Basic

Chemin direct :

```
If tableau(Origine, Destination)< 1 Then
    tableau(Origine, Destination) = 1
End if
```

Chemins indirects :

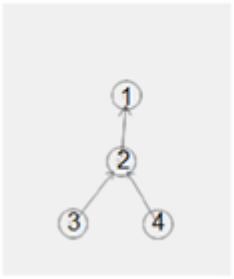
```
For i = 1 To Client_N
  For h = 1 To Client_N
    If tableau(i, h) = 1 Then
      For j = 1 To Client_N
        If tableau(h, j) <> 0 And tableau(h, j) + 1 > tableau(i, j) Then
          tableau(i, j) = tableau(h, j) + 1
        End If
      Next
    End If
  Next
Next
```

(5)

Exemple de fonctionnement du programme :

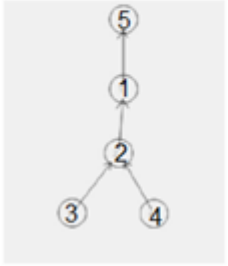
Il y a 4 clients à la boulangerie (situation initiale) et arrive un cinquième client très important qui se positionne devant tout le monde : création du rond "5" puis création d'une flèche de 1 vers 5.

Situation initiale



	1	2	3	4
1				
2	1			
3	2	1		
4	2	1		

Nouveau client
Nouveau lien



	1	2	3	4	5
1					1
2	1				2
3	2	1			3
4	2	1			3
5					

Pour le chemin direct l'algorithme met un 1 en (1,5)

Pour tous les chemins indirects ....

Notes d'édition

(1) La première figure n'a pas de lien avec la phrase précédente, elle donne une explication sur le calcul de 3 !. C'est la deuxième figure qui illustre les explications commençant par « Pour trouver le nombre de possibilités... »

(2)Dans le programme, ce que signifie somme (i;j) n'est pas très clair. Si le tableau précédent s'appelle A, la case à l'intersection de la ligne i et de la colonne j est notée Aij et on somme les Aij quand j varie entre 1 et 5. C'est dommage que l'algorithme ne soit pas commenté ou expliqué, on aimerait comprendre comment les élèves l'ont trouvé

(3) A côté du premier tableau, on peut rappeler que, comme précédemment, si i est derrière j on met 1 dans la case à l'intersection de la ligne i et de la colonne j.

(3bis) A côté du tableau du bas, ici on met 1 dans la case à l'intersection de la ligne i et de la colonne j si i est deux places derrière j

(4) Pour illustrer le schéma, on peut donner l'explication suivante : on peut multiplier les matrices représentant les chemins de longueur 1 pour obtenir le tableau représentant les chemins de longueur 2 comme expliqué sur le schéma

(5) L'édition regrette le manque d'explications et de commentaires pour les programmes, en particulier on se demande comment ils permettent de résoudre le problème.