**Université de Caen** — **Année 2011-2012**
**UFR de Sciences** — **2ème session - juin 2012**

**EI61T Approfondissement en algorithmique**
**durée 2h00**
Les notes de cours et TD sont autorisées.
Les parties sont indépendantes, correspondent aux différentes parties du cours et doivent être rendues sur des copies séparées.

Chaque candidat doit, au début de l'épreuve, porter son nom dans le coin de la copie qu'il cachera par collage après avoir été pointé. Il devra en outre porter son numéro de place sur chacune des copies, intercalaires, ou pièces annexées.

# 1 Algorithmique répartie (à rédiger sur une copie séparée)

## 1.1 Introduction

On se place dans un réseau de sites communiquant par messages. Les lignes de communication entre sites sont supposées bi-directionnelles et fiables (tout message émis à une extrémité est reçu par l'autre extrémité en un temps fini). Les sites n'ont pas d'identité au départ, et ne connaissent donc que les lignes de communication qui y sont reliées. Ces lignes sont, dans chaque site, numérotées de 1 au nombre de lignes reliées à ce site (qu'on notera `nblignes`, cette constante pouvant bien entendu prendre des valeurs différentes selon les sites).

Un des sites, l'initiateur, joue un rôle particulier.

## 1.2 Algorithme réparti à construire

On désire que l'initiateur lance un algorithme réparti dans le but de donner aux sites du réseau des identités 2 à 2 distinctes, reflétant leur position par rapport au parcours réalisé par cet algorithme. Plus précisément :

- On utilisera un parcours parallèle, avec diffusion contrôlée des messages émis par l'initiateur
- L'initiateur choisira pour sa propre identité le nombre 1.
- Tout site qui aura été numéroté avec une suite de caractères x numérotera ses voisins qui accepteront son message avec la chaîne de caractères obtenue par concaténation de x avec un tiret (-) puis le numéro de la ligne qui mène vers ce voisin.

**Question 1.1** *Écrire l'algorithme réparti respectant les spécifications ci-dessus, en ajoutant à un algorithme de parcours parallèle vu en cours ou TD ce qui est nécessaire pour fournir les identités à chaque site et leur prise en compte par ces derniers. L'identité de chaque site autre que l'initiateur devra être celle fournie par le premier message reçu du parcours parallèle.*

**Question 1.2** *En supposant que le réseau est connexe et fini, prouver que votre algorithme se termine en un temps fini et que tout site du réseau recevra un message, et aura une identité, en un temps fini.*

## 1.3 Propriétés de l'algorithme

On suppose que l'algorithme a été exécuté sur un réseau de topologie quelconque.

**Question 1.3**
- *qu'obtient-on comme structure sur le graphe du réseau (celle-ci pouvant être mémorisée totalement ou partiellement dans les variables de chaque site, selon l'écriture que vous avez choisie pour votre algorithme) ?*
- *que représentent les identités obtenues par chaque site relativement à cette structure ?*
- *les identités obtenues sont elles bien deux-à-deux distinctes ? Justifiez.*

## 1.4 Utilisations

**Question 1.4** *L'algorithme de la section 1.2 ayant été exécuté sur un réseau quelconque, décrivez comment :*
- *un site quelconque peut envoyer un message à l'initiateur ;*
- *un site peut, à partir de l'identité qu'il a acquise avec cet algorithme, trouver l'identité de celui dont il a reçu cette identité ;*
- *un site quelconque peut connaître le nombre de liaisons que devra parcourir un tel message pour parvenir à l'initiateur (donc en fait sa distance à ce dernier).*

## 2 Algorithmique des graphes & et pour la crypto (rédiger sur une copie séparée)

### – Exercice I –

On donne les durées suivantes de trajets :

| | |
|---|---|
| Nantes ↔ Paris-Montparnasse | 2 h |
| Nantes ↔ Lyon | 7 h |
| Paris-Montparnasse ↔ Paris-Lyon | 1 h (en autobus) |
| Paris-Lyon ↔ Grenoble | 4 h 30 |
| Paris-Lyon ↔ Lyon | 3 h 30 |
| Marseille ↔ Lyon | 3 h |
| Marseille ↔ Grenoble | 4 h 30 |
| Lyon ↔ Grenoble | 1 h |

Ces données peuvent être représentées par un graphe non orienté $G$ auquel on fera référence dans les questions qui suivent.

**Q1**. Représenter le graphe $G$.

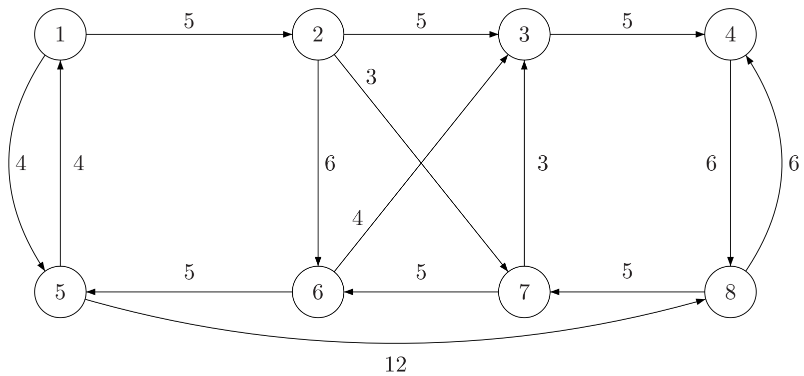
**Q2**. Écrire la matrice d'adjacence associée au graphe $G$.

### – Exercice II –

Appliquez l'algorithme de Dijkstra pour déterminer les plus courtes distances des sommets du graphe suivant au sommet 1. On donnera la trace d'exécution de l'algorithme. On utilisera notamment deux tableaux $\pi$ et $d$. À la fin de l'exécution le tableau $\pi$ contiendra pour chaque sommet son prédécesseur dans un plus court chemin du sommet 1 à ce sommet. Le tableau $d$ contiendra la longueur des plus courts chemins du sommet 1 à chaque sommet : $\forall i \in \{1 \ldots 8\}$,

$\pi[i]$ := prédécesseur du sommet $i$ dans un plus court chemin du sommet 1 au sommet $i$,
$d[i]$ := longueur d'un plus court chemin du sommet 1 au sommet $i$.

On précisera comment ces deux tableaux sont initialisés et comment leurs contenus évoluent à chaque itération de l'algorithme.




### – Exercice III –

**Q1**. Rappeler le principe de l'exponentiation rapide, et la complexité du calcul de l'opération $a^b \mod n$.

**Q2**. Calculer par cette méthode, en donnant la trace d'exécution de l'algorithme, $5^9 \mod 11$.