ÉPREUVE D'INFORMATIQUE COMMUNE

Durée : 2 heures

L'usage de calculatrices est interdit. — Aucun document n'est autorisé.

La présentation, la lisibilité, l'orthographe, la qualité de la rédaction, la clarté et la précision des raisonnements entreront pour une part importante dans l'appréciation des copies. En particulier, les candidats devront apporter les commentaires suffisants à la compréhension de leurs programmes et veilleront à utiliser des noms de variables explicites. Si un candidat est amené à repérer ce qui peut lui sembler être une erreur d'énoncé, il le signalera sur sa copie et devra poursuivre sa composition en expliquant les raisons des initiatives qu'il a été amené à prendre.

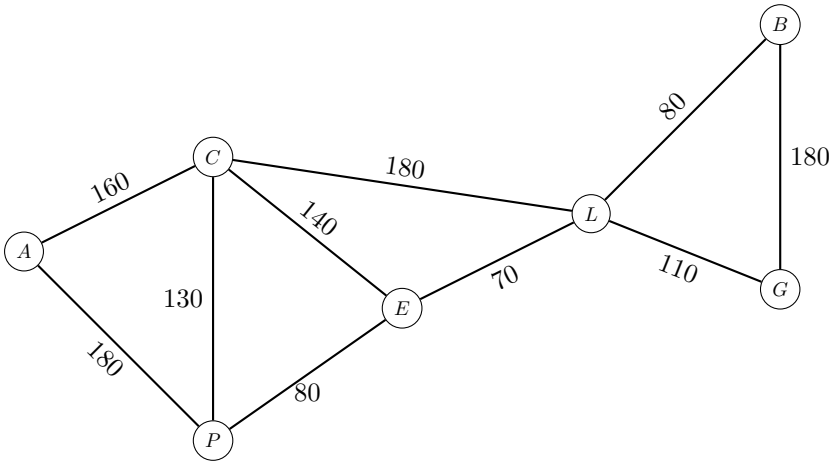
Le sujet est composé de trois exercices indépendants.

Gestion du temps — En admettant une durée de 5 minutes pour la lecture et l'assimilation du sujet, il est vivement conseillé de consacrer environ 30 minutes sur l'exercice 1, 55 minutes sur l'exercice 2 et 30 minutes sur l'exercice 3.

Exercice 1 – Parcours routier

Cet exercice sera traité sur le document réponse n° 1.

Le graphe ci-dessous indique les distances routières, exprimées en kilomètres, entre de grandes villes de la région Auvergne-Rhône-Alpes.



Légende :

A : Aurillac

B : Bourg-en-Bresse

C : Clermont-Ferrand

E : Saint-Étienne

G : Grenoble

L : Lyon

P : Le Puy-en-Velay

Question 1.1. *Un automobiliste désire se déplacer de Bourg-en-Bresse à Aurillac en suivant exclusivement les routes indiquées ci-dessus. Déterminer le trajet le plus court (et la distance associée) en utilisant l'algorithme de Dijkstra. En cas d'égalité entre deux étiquettes, on choisira le premier sommet dans l'ordre alphabétique.*

Question 1.2. *Afin de déterminer plus rapidement le meilleur parcours, l'automobiliste décide d'utiliser l'algorithme $A^\star$ en utilisant comme heuristique la fonction qui à chaque ville associe sa distance à Aurillac à vol d'oiseau. Le tableau ci-dessous résume ces distances.*

Ville	*A*	*B*	*C*	*E*	*G*	*L*	*P*
Distance (km)	*0*	*260*	*110*	*160*	*260*	*210*	*110*

1. *Donner le graphe obtenu en modifiant le graphe initial avec l'heuristique proposée.*
2. *Déterminer le plus court chemin (et la distance associée) à l'aide de l'algorithme $A^\star$. Que constate-t-on ?*

Exercice 2 – *Il y a de la tension dans l'air !*

Dans de nombreux logiciels ou appareils scientifiques, on dispose de mesures et de calculs automatiques. On se propose de programmer dans ce problème quelques uns de ces traitements. De façon générale, le traitement portera sur un ou deux signaux de tensions, et l'on aura donc au plus en argument des différents traitements trois listes :

- une liste pour les différents instants ;
- une liste pour chacune des deux tensions.

Pour simplifier les traitements, on fera les hypothèses suivantes :

- les signaux sont périodiques ;
- chaque liste de mesure correspond à plusieurs périodes.

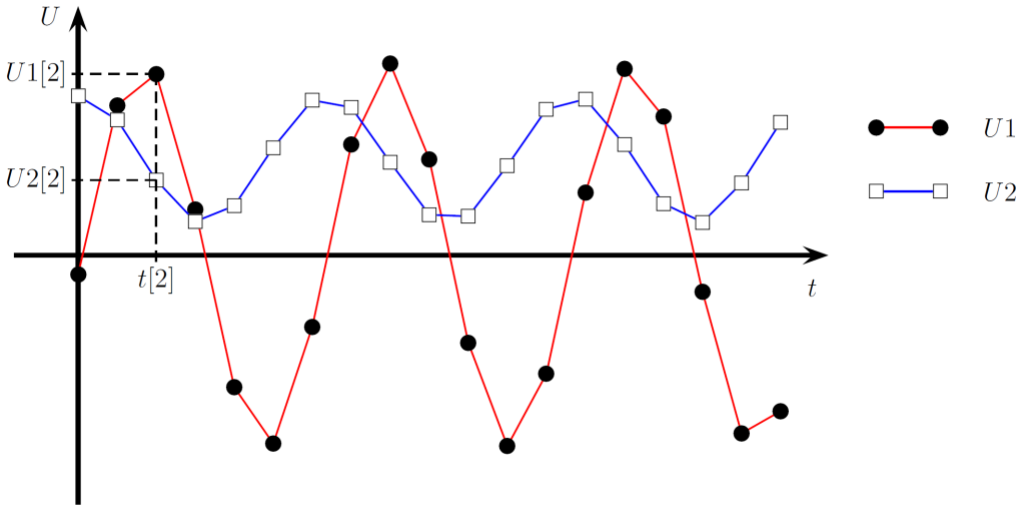


FIGURE 1 – Exemple de tensions mesurées.

Question 2.1. *Le premier traitement automatique est le calcul de la moyenne. Écrire une fonction* `moy` *qui prend en argument une liste de nombres* `U` *et qui renvoie la moyenne de cette liste.*

Question 2.2. *Le système entre en surtension dès que la tension dépasse* 220 V*, le but est donc de le détecter pour couper le courant. Implémenter une fonction* `surtension` *qui détecte si la liste* `U` *contient une valeur supérieure à un seuil, par défaut* 220 V*, et renvoie le booléen* `True` *dans ce cas, et* `False` *dans le cas contraire.*

Question 2.3. *Écrire une fonction* `max` *qui prend en argument une liste de nombres* `U` *et qui renvoie la valeur maximale de la liste.*

Question 2.4. *Implémenter de même une fonction* `min` *qui prend en argument une liste de nombres* `U` *et qui renvoie la valeur minimale de la liste en faisant appel à la fonction* `max`*.*

Question 2.5. *On considère maintenant l'amplitude crête-à-crête, c'est-à-dire la différence entre la valeur maximale et la valeur minimale de la tension. Écrire une fonction* `vpp` *qui prend en argument une liste de nombres* `U` *et qui renvoie l'amplitude crête-à-crête correspondante.*
Exemple : `vpp([4,2,1,2.5])` *doit renvoyer* `3`*.*

Question 2.6. *La plupart des logiciels propose des opérations mathématiques, comme par exemple la soustraction. Écrire une fonction* `soustraction` *qui prend en argument deux listes de nombres de même longueur* `U` *et* `V` *et qui renvoie la liste de même longueur contenant la soustraction terme à terme des deux listes (la première moins la deuxième).*
Exemple : `sous([1,2,2,0],[1,1,2,1])` *doit renvoyer* `[0,1,0,-1]`*.*

Sur le schéma ci-contre, les pointillés indiquent la valeur moyenne. Les numéros correspondent aux indices de chaque point. Afin de mesurer la période, on souhaite détecter les fronts montant, c'est-à-dire les instants où un signal franchit sa valeur moyenne en montant (parties en gras).

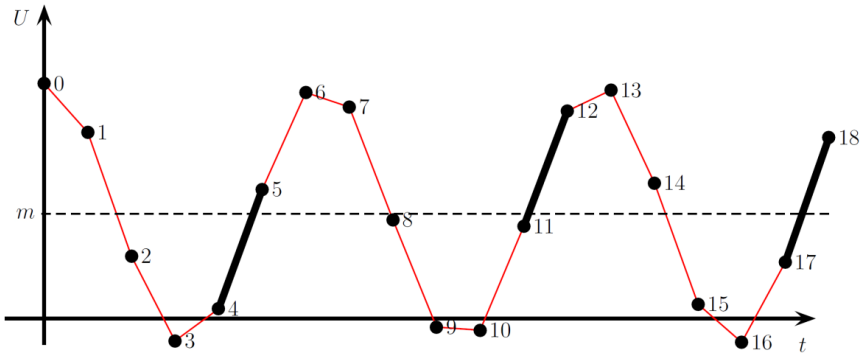


FIGURE 2 – Détection des fronts montants.

Question 2.7. *Si on note U la liste des valeurs de tension et m sa valeur moyenne, et $i \in [\![0, n-2]\!]$ un indice de la liste U. Exprimer les conditions sur U_i et U_{i+1} correspondant à la détection d'un front montant. En déduire une fonction* `fronts` *qui prend comme argument une liste de nombre* `U` *et renvoyant la liste des indices (à gauche) vérifiant cette condition. Exemple : appliquée au tableau associé à la figure 2, la liste renvoyée par la fonction* `fronts` *devrait être* `[4,11,17]`.

Question 2.8. *Implémenter une fonction* `periodes` *qui prend en argument deux listes* `T` *et* `U` *de même longueur représentant respectivement les instants et les valeurs de tension associées, et qui renvoie le nombre de périodes identifiées et la moyenne des périodes identifiées.*

On rappelle que la valeur efficace d'une tension est définie par

$$U_{\text{eff}} = \sqrt{\langle u^2(t) \rangle} = \sqrt{\frac{1}{T} \int_0^T u^2(t)\, \mathrm{d}t}$$

Pour calculer une valeur approchée de l'intégrale $\displaystyle\int_a^b f(x)\, \mathrm{d}x$ d'une fonction f dont on connaît la valeur en n points équirépartis dans l'intervalle $[a; b]$, $\forall k \in [\![0; n-1]\!]$, $x_k = a + k\left(\dfrac{b-a}{n}\right)$, la méthode des trapèzes s'écrit :

$$\mathcal{A}(n) = \frac{b-a}{2n} \sum_{k=0}^{n-1} \left(f(x_k) + f(x_{k+1})\right)$$

Question 2.9. *Écrire une fonction* `trapezes` *qui prend en argument deux listes* `T` *et* `V` *de même longueur représentant respectivement les instants et les valeurs associées, et qui renvoie une approximation de l'intégrale correspondante avec la méthode des trapèzes.*

Question 2.10. *Écrire une fonction* `efficace` *qui prend comme arguments deux listes* `T` *et* `U` *de même longueur représentant respectivement les instants et les valeurs de tension associées, et qui renvoie une estimation de la valeur efficace sur le plus grand nombre de périodes possible.*

Exercice 3 – Tri de chaînes de caractères

On s'intéresse au problème de tri de chaînes de caractères par ordre alphabétique, par exemple afin de former un dictionnaire. Pour ce faire, on considère des chaînes de caractères constituées uniquement des 26 lettres latines minuscules de `a` à `z`. On suppose en outre disposer d'une fonction `rang` qui à chaque lettre associe un entier entre 0 et 25 correspondant à sa position dans l'alphabet. Par exemple, `rang('a')`, `rang('e')` et `rang('z')` renvoient respectivement 0, 4 et 25.

Question 3.1. *Définir une fonction* `lettres` *qui prend comme argument une liste de chaînes de caractères* `L` *et qui renvoie l'entier correspondant au nombre de lettres de la chaîne la plus longue.*

On suppose dans un premier temps que les listes de chaînes de caractères considérées sont constituées de chaînes de caractères de même longueur p, où p est en entier fixé.

Question 3.2. *Définir une fonction* `comptage` *qui prend comme arguments une liste de n chaînes de caractères* `L` *de longueur p et un nombre entier* `i` *compris entre* 0 *et* $p-1$*, et qui renvoie en temps linéaire, c'est-à-dire en* $O(n)$*, une liste* `T` *contenant exactement 26 éléments, de sorte que pour tout indice* $k \in [\![0, 25]\!]$*,* `T[k]` *soit égale à la liste des chaînes de caractères de* `L` *dont le caractère d'indice i est celui de rang k. Par exemple :*

```
>>> comptage(['rza', 'spm', 'sts', 'dqp', 'voi', 'gca', 'rtu'], 1)
[[], [], ['gca'], [], [], [], [], [], [], [], [], [], [], ['voi'],
['spm'], ['dqp'], [], [], ['sts', 'rtu'], [], [], [], [], [], ['rza']]
```

Question 3.3. *Écrire une fonction* `fusion` *qui prend en entrée une liste de listes de chaînes de caractères* `T` *et qui renvoie la liste obtenue en concaténant les listes* `T[0]`*,* `T[1]`*,... sans utiliser la concaténation avec « + ». Par exemple, si* `T` *est la liste obtenue dans la question précédente, alors :*

```
>>> fusion(T)
['gca', 'voi', 'spm', 'dqp', 'sts', 'rtu', 'rza']
```

On considère le code suivant :

```
A=['bcd', 'dab', 'ccc', 'ecf', 'add', 'ada', 'daa']
B=fusion(comptage(A,2))
C=fusion(comptage(B,1))
D=fusion(comptage(C,0))
```

Question 3.4. *Écrire les listes* `B`*,* `C` *et* `D` *obtenues. Que constatez-vous concernant la liste* `D` *?*

Question 3.5. *En déduire une fonction* `trip` *qui prend comme argument une liste* `L` *de chaînes de caractères de longueur p et qui renvoie une image triée par ordre alphabétique de la liste* `L`*.*

Afin de traiter de se rapprocher du « vrai » problème de tri alphabétique de mots, on considère maintenant que les listes de chaînes de caractères peuvent être constituées de chaînes de caractères de longueurs différentes.

Question 3.6. *Définir une fonction* `dico` *qui prend comme argument une liste* `L` *de chaînes de caractères de longueurs différentes et qui renvoie une image triée par ordre alphabétique de la liste* `L`*.*

— Fin de l'énoncé —

NOM : _____

Document réponse n° 1

(Exercice 1)

Question 1.1.

	A	B	C	E	G	L	P
Initialisation							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							

.....

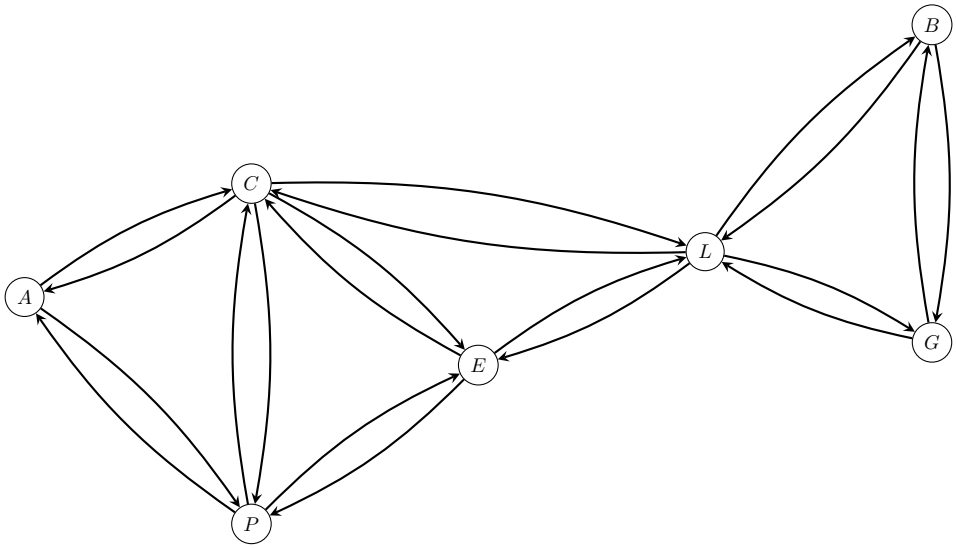
.....

.....

.....

.....

Question 1.2.



	A	B	C	E	G	L	P
Initialisation							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							
Étude de							