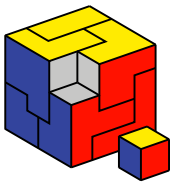
Prologin 2018



Concours national d'informatique

Épreuve écrite d'algorithmique

Une histoire de pizzas

1 Préambule

Bienvenue à **Prologin**. Ce sujet est l'épreuve écrite d'algorithmique et constitue la première des trois parties de votre épreuve régionale. Sa durée est de 3 heures. Par la suite, vous passerez un entretien (20 minutes) et une épreuve de programmation sur machine (4 heures).

Conseils

- Lisez bien tout le sujet avant de commencer.
- **Soignez la présentation** de votre copie.
- Les exercices ne sont pas en ordre croissant de difficulté ! Lisez **vraiment** bien tout le sujet.
- N'hésitez pas à poser des questions.
- Si vous avez fini en avance, relisez bien, ou préparez votre présentation pour l'entretien.
- N'oubliez pas de passer une bonne journée.

Remarques

- Le barème est donné à titre indicatif uniquement.
- Indiquez lisiblement vos nom et prénom, la ville où vous passez l'épreuve et la date en haut de votre copie.
- Tous les langages sont autorisés, veuillez néanmoins préciser celui que vous utilisez.
- Ce sont des humains qui lisent vos copies : laissez une marge, aérez votre code, ajoutez des commentaires (**seulement** lorsqu'ils sont nécessaires) et évitez au maximum les fautes d'orthographe, sinon ça va barder.
- Le barème récompense les algorithmes les plus efficaces : écrivez des fonctions qui trouvent la solution le plus rapidement possible.
- Si vous trouvez le sujet trop simple, relisez-le, réfléchissez bien, puis dites-le-nous, nous pouvons ajouter des questions plus difficiles.
- Les questions bonus sont à traiter seulement une fois que le sujet entier a été essayé.

2 Concours pour pizzaphile

Vous voilà arrivé à la demi-finale du concours Prologin, félicitations [1] ! Non seulement, vous allez devoir résoudre ce stupide sujet d'algorithmique bien trop dur [2], mais aussi choisir votre pizza pour midi.

3 Préparatif

Les organisateurs de la demi-finale vous fournissent une liste de pizzas disponibles à la meilleure pizzeria de tout Lausanne [3]. On part du principe que la pizzeria peut préparer autant de pizzas d'un type de pizzas disponible qu'elle le veut. Par exemple [4], vous avez ces différents choix :

— Marguerita
— Quattro stagioni
— Merguez
— Funghi
— Napolitaine

Pour simplifier le sujet, on suppose que chaque type de pizza est représenté par un entier unique. Nécessairement, il y a toujours un candidat qui désire avoir une pizza Hawaïenne (beurk !) avec un supplément de jambon de Parme, de roquette, des copeaux de Parmesan et de la truffe. Votre première mission est d'aider les organisateurs [5] à vérifier si les choix des candidats sont valides.

Question 1 (2 points)

Écrire une fonction qui, donné un choix de pizzas et une liste de pizzas disponibles, renvoie vrai si le choix est licite.

> On définit P comme le nombre de pizzas différentes disponibles.

Question 2 (1 point)

Donner la complexité de votre algorithme pour la Question 1, c'est-à-dire le nombre d'opérations effectuées par l'algorithme en fonction de P.

Question 3 (3 points)

On suppose maintenant que la liste des pizzas disponibles est triée. Réécrire une version plus efficace de l'algorithme de la Question 1.

Question 4 (1 point)

Donner la complexité (en fonction de P) de votre algorithme pour la Question 3.

4 Alea jacta est [6]

C'est bon, tous les candidats ont choisi leur pizza. Malheureusement pour les organisateurs, les candidats ont tous choisi une pizza différente [7]. À partir de maintenant, il faut rajouter la complexité de l'algorithme proposé pour chaque question. De plus la note sera attribuée en fonction de l'efficacité de l'algorithme proposé.

Question 5 (2 points)

Donné une liste de pizzas choisies, vérifier si chaque candidat a bien pris une pizza différente.

> On suppose maintenant qu'il y a N candidats (donc N pizzas différentes). De plus, les pizzas sont numérotées de 1 à N.

1. Avouez-le, vous êtes juste venu pour les pizzas
2. L'auteur du sujet déteste les gens qui... les gens en général en fait
3. Comprendre : la pizzeria la plus proche du centre d'examen
4. Note aux organisateurs : actualiser cette liste le jour de l'épreuve
5. Être le chouchou des organisateurs pour se qualifier en finale
6. *Les dés sont tombés*, parfois aussi librement traduit comme *les pizzas ont été choisies*
7. Les candidats Prologin sont barbants

Les pizzas sont enfin arrivées. Cependant, un candidat [8] a décidé de changer sa pizza. Heureusement, les pizzas sont finalement réparties telles que chaque personne obtient exactement une pizza : la personne i prend en effet la pizza qui a été initialement destinée à la personne p_i.

Question 6 (4 points)

Donné l'affection initiale des candidats vers les pizzas (un tableau où la i-ième case représente la pizza du candidat i), et le tableau p_i (où la i-ème case représente l'indice du candidat p_i duquel le candidat i obtient sa pizza), trouver l'affection finale des pizzas.

Question 7 (4 points)

Le candidat d'avant a changé son avis, il désire maintenant garder sa pizza initiale. Malheureusement, vous avez déjà jeté la configuration initiale des pizzas. Vous avez cependant l'affection des pizzas après que le candidat a changé son avis et le tableau p_i. Retrouver l'affection initiale des pizzas.

Question 8 (3 points)

Le candidat d'avant a encore changé son avis [9]. Les pizzas étant maintenant presque froides, vous décidez d'aider les organisateurs [10], et répartissez les pizzas. Écrire un algorithme qui calcule une affection au hasard des pizzas. On suppose que votre algorithme a accès à une fonction qui, donné k, retourne un entier au hasard entre 1 et k avec une distribution uniforme (c'est-à-dire chaque nombre a la même probabilité d'être tiré). Votre algorithme doit retourner chaque affection de pizzas avec la même probabilité.

Question 9 (3 points)

Vous venez de voir que la fonction qui tire un nombre au hasard entre 1 et k ne fonctionne pas comme voulu. Ne voulant pas adapter votre algorithme précédent, vous décidez de créer la fonction qui tire un entier entre 1 et k vous-même. Heureusement, vous avez accès à une fonction qui retourne soit 0 ou 1 avec probabilité égale. Attention, votre fonction doit avoir la même probabilité de tirer chaque nombre.

5 Correction

Malheur ! La majorité des candidats n'a pas écouté lors de la correction à midi. Les organisateurs ont donc décidé de rajouter une partie à l'épreuve écrite [11]. Dans cette partie, donné une liste d'entiers uniques, on appelle le problème de **3-Sum** le problème de déterminer s'il y a trois éléments dans la liste (prendre le même élément plusieurs fois est autorisé) tels que la somme est égale à 0.

Question 10 (1 point)

Quel est le résultat de **3-Sum** avec l'entrée $5, 2, -2, 4, 7$? Quel est le résultat pour l'instance $-3, 7, 4, 10$?

Question 11 (3 points)

Trouver un algorithme trivial pour résoudre **3-Sum** (la complexité temporelle de l'algorithme ne sera pas pris en compte lors de la correction).

Question 12 (5 points)

Trouver un algorithme qui, donné une liste de nombres et un nombre cible, retourne vrai si il y a deux éléments (prendre le même nombre deux fois est autorisé) dont la somme est égale au nombre cible.

Question 13 (4 points)

À l'aide de la question précédente, trouver un algorithme rapide pour résoudre **3-Sum**.

8. Dont on a malheureusement pas le droit de révéler son identité
9. Des légendes racontent qu'il a été retrouvé au fond du lac Léman ou du Rhône après l'épreuve régionale
10. Vous voulez *vraiment* être le chouchou
11. Les organisateurs sont sadiques

Question 14 (3 points)

Supposons que vous avez à disposition un algorithme qui résout le problème de **3-Sum**°. Donné une liste d'entiers uniques, **3-Sum**° renvoie vrai si et seulement il existe 3 éléments distincts (cette fois, prendre le même élément plusieurs fois n'est pas autorisé) tels que la somme est égale à 0. Résoudre **3-Sum** à l'aide de **3-Sum**°.

Question 15 (6 points)

Supposons que vous avez à disposition un algorithme qui, donné un ensemble de points retourne vrai si 3 points sont colinéaires. Résoudre **3-Sum**° à l'aide de l'algorithme pour vérifier la colinéarité.

Question 16 (5 points)

Que peut-on conclure de la Question 15 à propos de la complexité (c'est-à-dire la complexité temporelle minimale) du problème de la colinéarité ?

6 Bonus

Question bonus 1 (-1 point)

Si $\mathbf{N} = 1$ et $\mathbf{P} = 2$, alors est-ce que $\mathbf{P} = \mathbf{NP}$?

Question bonus 2 (5 points)

Donné une liste d'entiers de taille n et un entier $k \geq 3$, trouver un algorithme qui détermine en $O(n^{k-1})$ [12] si il y a k éléments (les répétitions sont autorisées) dont la somme est égale à 0.

Question bonus 3 (10 points)

Composer un morceau musical dans le style des Nocturnes de Chopin [13].

Question bonus 4 (10 points)

Donné un ensemble de n points A, et un ensemble de m points B, trouver un algorithme qui détermine pour chaque point p dans B si il y a trois points dans A tel que le triangle formé par ces points contient p.

Question bonus 5 (15 points)

Donné un ensemble A de n points dans le plan qui 1) ne contient pas l'origine $(0, 0)$ et 2) il n'existe aucune droite qui contient au moins 3 points de $A \cup (0, 0)$, trouver un algorithme rapide qui compte le nombre de triangles (dont les sommets sont dans l'ensemble A) qui contiennent l'origine.

Question bonus 6 (42 points)

Montrer que **3-Sum** peut être résolu en $o(n^2)$, où n désigne le nombre d'éléments en entrée. Un super-bonus [14] est envisageable si vous montrez que **3-Sum** est faisable en ou n'est pas faisable en $O(n^{2-\epsilon})$ pour un $\epsilon > 0$.

12. C'est-à-dire qu'il existe une constante $c > 0$ telle que pour chaque n assez large, l'algorithme prend au maximum cn^{k-1} opérations de base
13. N'hésitez pas à nous demander du papier supplémentaire
14. À condition de donner votre preuve à l'auteur du sujet sans la montrer à autrui

Le sujet comporte 4 pages (sans compter la page de garde), 16 questions, et 6 questions bonus. Les questions normales sont notées sur 50 points, et les questions bonus rapportent au total 81 points, plus 1 point de présentation.