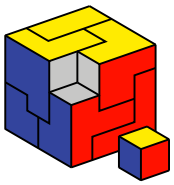
Prologin 2018



Concours national d'informatique

Épreuve écrite d'algorithmique

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 Les crêpes dont vous êtes le héros

En se levant ce matin, Joseph Marchand a eu une idée fantastique : il va créer un concours ! Pour ça, il a tout prévu : les organisateurs, les candidats... mais il a oublié de choisir une méthode de sélection pour envoyer les candidats en finale. En train de développer une méthode objective et équitable, il se rend compte que c'est en fait peine perdue [1] ! En effet, il remarque que les orgas sont très capricieux en matière de candidats : chaque orga a des chouchous (des candidats qu'il adore) et des anti-chouchous (des candidats qu'il déteste).

Si jamais aucun de ses chouchous ne va en finale et que tous ses anti-chouchous vont en finale, l'orga démissionne. Joseph a un problème : les vœux des organisateurs sont-ils compatibles ? Malheureusement pour lui, les effectifs de son association sont très réduits [3] et la moindre démission entraînerait l'annulation du concours, faute de suffisamment de personnes habilitées à faire des crêpes.

Grand amateur de crêpes, vous n'avez bien entendu pas envie que cela arrive. Joseph Marchand vous ayant remarqué pendant les sélections de Prologin, il a fait appel à vous pour l'aider à établir la liste des qualifiés en finale, si cela est possible.

Pour cela, il vous communique pour chaque orga sa liste de chouchous et sa liste d'anti-chouchous. Il y a N candidats, numérotés de 1 à N, et M orgas, numérotés de 1 à M.

3 Satisfiabilité

On va associer à chaque organisateur une formule booléenne. Par exemple, si la liste des chouchous de Joël est $\{2, 3, 5, 7\}$ et que sa liste d'anti-chouchous est $\{4, 9\}$, alors on lui associe la formule suivante ($\vee$ signifie « ou », et $\neg$ signifie « pas ») :

$$\phi_{\text{Joel}} = x_2 \vee x_3 \vee x_5 \vee x_7 \vee \neg x_4 \vee \neg x_9$$

Si Alexandre a comme liste de chouchous $\{4, 5\}$ et pas d'anti-chouchous, alors sa formule est $\phi_{\text{Alex}} = x_4 \vee x_5$. Ainsi, pour satisfaire Joël et Alexandre, il faut satisfaire la formule

$$(x_2 \vee x_3 \vee x_5 \vee x_7 \vee \neg x_4 \vee \neg x_9) \wedge (x_4 \vee x_5)$$

$\wedge$ signifie « et », c'est-à-dire que les deux formules doivent être satisfaites.
x_i vaut vrai si le candidat i est sélectionné, et faux s'il ne l'est pas..
$a_1 \vee a_2 \ldots \vee a_k$ vaut vrai si et seulement si au moins un des a_i vaut vrai.
$a_1 \wedge a_2 \ldots \wedge a_k$ vaut vrai si et seulement si tous les a_i valent vrai.
$\neg a$ vaut vrai si et seulement si a vaut faux, et inversement.
Il faut noter que $\neg$ est prioritaire sur $\wedge$ qui est prioritaire sur $\vee$. C'est à dire qu'on écrira $A \vee \neg B \wedge C$ pour $(A) \vee ((\neg B) \wedge (C))$.
On appellera formule ϕ la formule correspondant aux souhaits de tous les organisateurs :

$$\phi = \phi_{\text{orga 1}} \wedge \phi_{\text{orga 2}} \wedge \ldots \wedge \phi_{\text{orga M}}$$

Dans toute cette partie, toutes les formules dont on parle sont des formules correspondant aux souhaits des organisateurs.

On dit qu'une formule est satisfiable si on peut choisir un ensemble de finalistes tels qu'aucun organisateur ne démissionne. Cela revient à trouver une affectation pour chaque x_i (vrai si le candidat i va en finale, faux s'il n'y va pas) telle que la formule ϕ soit vraie.
Si vous n'avez pas compris ces explications après 10 minutes de réflexion, demandez des clarifications à un organisateur.

Question 1 (1 point)

Donnez un exemple de formule représentant les chouchous et anti-chouchous des orga satisfiable et un exemple de formule non satisfiable.

Question 2 (2 points)

Lisez attentivement la partie 1 et donnez une struture de données pour représenter les formules des chouchous et anti-chouchous des orga et les affectations des variables. Réfléchissez bien avant de répondre : vous utiliserez cette structure dans les questions suivantes.

1. à ne pas confondre avec le pain perdu [2]
2. « qui ne l'est pas vraiment » (si vous avez la référence, dites-le lors de votre entretien ;-) »
3. mais pas assez réduits pour que les préférences des orgas ne posent pas problème

Question 3 (2 points)

Écrivez une fonction qui prend en entrée une formule et une affectation des x_i et qui renvoie la valeur de cette formule (vrai ou faux).

Question 4 (3 points)

Écrivez une fonction qui, pour une formule donnée, renvoie (ou affiche) une affectation qui satisfait la formule, ou « NON SATISFIABLE » si aucune affectation ne satisfait la formule.

Question 5 (2 points)

Donnez la complexité (en fonction de N et de M) de votre algorithme pour la Question 3.

Question 6 (1 point)

Pour cette question, il y a autant de candidats que d'orgas [4]. Joseph Marchand dispose d'un ordinateur avec un processeur à 1 Ghz sur lequel il peut exécuter votre programme pendant 0,01 seconde [5]. Pour quel N au maximum votre programme termine-t-il sur l'ordinateur de Joseph dans le temps imparti (donnez un ordre de grandeur) ?

4 2-Satisfiabilité

C'est une catastrophe ! Joseph a exécuté votre programme, et il n'a pas réussi à terminer dans le temps imparti ! Dans la situation actuelle, il lui est donc impossible de trouver en temps raisonnable [6] une liste de finalistes qui les satisfait tous.

Comme il est aussi un grand crêpophile, il a décidé de prendre des mesures drastiques : chaque organisateur a l'obligation d'avoir exactement deux préférences (donc soit deux chouchous, soit deux anti-chouchous, soit un de chaque). Pas plus, pas moins [7]. Joseph vous rappelle pour vous annoncer la bonne nouvelle : il y aura peut-être une finale ! Mais cela dépend encore de vous... (et des orgas : si leurs préférences sont incompatibles, on ne pourra rien faire)

Dans toute cette partie, toutes les formules dont on parle sont des formules correspondant aux souhaits des organisateurs, chaque souhait étant de taille 2.

Joseph espère que maintenant qu'il a interdit aux orgas d'être trop capricieux, on va pouvoir trouver un algorithme plus rapide ! Vous avez justement une idée : utiliser l'opérateur $\Rightarrow$ (« implique »), dont voici la table de vérité :

p	q	$p \Rightarrow q$
vrai	vrai	vrai
vrai	faux	faux
faux	vrai	vrai
faux	faux	vrai

Cela veut dire que la formule $A \Rightarrow B$ n'est pas satisfaite si et seulement si A vaut vrai et B vaut faux.

Vous décidez que vous transformerez la formule ϕ à l'aide du symbole $\Rightarrow$.

Question 7 (2 points)

Montrez que $a \vee b$ a les mêmes valeurs de vérité que $(\neg a \Rightarrow b) \wedge (\neg b \Rightarrow a)$.

Question 8 (3 points)

On décide ainsi de réécrire notre formule en remplaçant tous nos $a \vee b$ par des $(\neg a \Rightarrow b) \wedge (\neg b \Rightarrow a)$. Proposez une structure de données pour la nouvelle formule telle qu'on puisse facilement et rapidement avoir accès pour chaque littéral [8] x à tous les littéraux qu'il implique

4. ils en ont de la chance, les candidats...
5. il a des problèmes pour payer sa facture d'électricité
6. par rapport à sa facture d'électricité, c'est-à-dire moins de 0,01 seconde
7. notez que, et cela n'a pas d'importance, des petits malins trichent en proposant deux fois le même chouchou ou anti-chouchou
8. on vous avait maintenu dans l'ignorance, mais le temps est venu de vous instruire : un littéral est ou bien une variable, ou bien la négation d'une variable (soit un x_i, soit un $\neg x_i$ dans notre cas)

Question 9 (2 points)

Donnez une fonction pour passer de votre structure précédente à celle de la Question 2. Donnez la complexité en nombre d'opérations de votre fonction (en fonction de N et M).
Dans la suite, on se sert de votre nouvelle structure de données.

Question 10 (2 points)

Montrez que si $a \Rightarrow \neg a$ et $\neg a \Rightarrow a$ apparaissent dans ϕ, alors ϕ ne peut pas être satisfaite. Pour rappel, ϕ est la formule concaténant tous les voeux des orgas.

Question 11 (1 point)

Montrez que si on a une chaîne $a \Rightarrow a_1 \Rightarrow a_2 \Rightarrow \ldots \Rightarrow a_u \Rightarrow \neg a \Rightarrow b_1 \Rightarrow b_2 \Rightarrow \ldots \Rightarrow b_v \Rightarrow a$ dans ϕ alors ϕ ne peut pas être satisfaite.

Question 12 (9 points)

Donnez un algorithme pour tester si tel est le cas. Donnez sa complexité. Notez que plus l'algorithme que vous écrirez sera rapide, plus on vous accordera de points.

On va montrer la réciproque : que si une formule n'a aucune chaine contradictoire (chaine de la forme $a \Rightarrow a_1 \Rightarrow a_2 \Rightarrow \ldots \Rightarrow a_u \Rightarrow \neg a \Rightarrow b_1 \Rightarrow b_2 \Rightarrow \ldots \Rightarrow b_v \Rightarrow a$), alors elle est bien satisfiable. On dit qu'on a trouvé une **condition necessaire et suffisante**[9].

Question 13 (6 points)

Donner un algorithme qui prend en entrée une formule sans chaîne contradictoire, et qui renvoie en sortie une affectation de cette formule qui la satisfasse. On pourra s'apercevoir quand lorsqu'une boucle contient 2 littéraux, alors ces deux littéraux doivent avoir la même affectation.

Question 14 (6 points)

Donner la complexité de l'algorithme précédent. Prouver qu'il renvoie bien une affectation qui satisfasse la formule.

5 Bonus

Ces questions ne sont à faire que si vous avez essayé tout le sujet.

Question 15 (3 points)

Montrez que tout formule logique comportant des $\neg$, des $\vee$ et des $\wedge$ peut être mise sous la forme de souhaits d'orgas, comme la formule ϕ. Note : les formules sous cette forme sont dites en « forme normale conjonctive ».

Question 16 (3 points)

Donnez un algorithme permettant de vérifier qu'une formule est bien parenthésée.

Question 17 (3 points)

Écrivez une fonction qui lit une formule bien parenthésée en forme normale conjonctive (donnée par une chaîne de caractères) et retourne la structure de données définie en Question 2. Exemple de formule devant être lue correctement :

```
(x1) ET (x2 ou NON(x3)) ET (NON(x4)) ET (x2 OU NON(x8))
```

Question 18 (5 points)

Maintenant, on ne garantit plus que les formules sont en forme normale conjonctive (mais elles sont toujours bien parenthésées). Donnez une structure de données adaptée à la lecture de ces formules et écrire une fonction qui les lit. Exemple de formule devant être parsée correctement :

```
(x1) ET ( (x2) ET ( (NON(x4) => (x3 => x7)) OU (x2 ET NON(x8)) ) )
```

9. Un peu de math n'a jamais fait de mal à personne...

Question 19 (42424242 points)

Adaptez votre algorithme de la Question 13 pour le cas où les clauses sont toutes de taille 3. On demande un algorithme linéaire en M et en N.