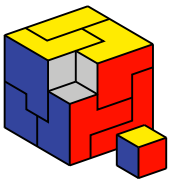
Prologin 2022



Concours national d'informatique

Épreuve écrite d'algorithmique

Jouons un peu

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.
- 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.

2 Se tourner les pouces

Quand les organisateurs de Prologin n'ont rien à faire, ils ne se tournent pas les pouces. Ils jouent à un jeu surnommé "le jeu là, tu sais avec les mains", il n'a pas vraiment nom.

Voilà en quoi il consiste :

Le jeu se joue à n ($n \geqslant 2$) joueurs. Chaque joueur possède en début de partie k[1] mains n'ayant qu'un doigt levé (le pouce par convention) représentant le chiffre 1.

Chaque joueur peut lors de son tour additionner la valeur d'une de ses mains à une autre main (de n'importe quel joueur). La main qui effectue l'action conserve sa valeur.

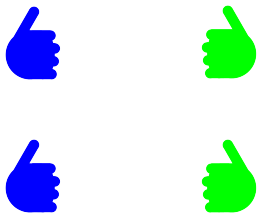
Une main est perdue quand elle atteint une valeur de 5, ses 5 doigts sont levés. Pour gagner il faut être la dernière personne avec au moins une main.

Bien sûr, il peut arriver que la valeur d'une main dépasse 5 (par exemple si une main de valeur 3 ajoute à une autre qui a la valeur 4). Les mains des organisateurs de Prologin étant des versions biologiques de base acquises à la naissance, elles ne se limitent qu'à 5 doigts, il faudra alors revenir à une valeur entre 1 et 5 inclus. Ainsi, on soustrait 5 à la valeur de la main après que l'autre valeur ait été ajoutée, on relève donc ce nombre de doigts (par exemple $(4 + 3) - 5 = 2$ doigts).

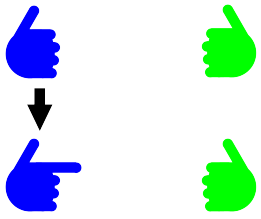
Les parties se font sur plusieurs tours où les joueurs jouent toujours dans le même ordre.

Prenons un début de partie rapide entre Marie Lussier (en bleu) et Baptiste Marchand (en vert) où nous mettrons les valeurs des mains concernées entre parenthèses.

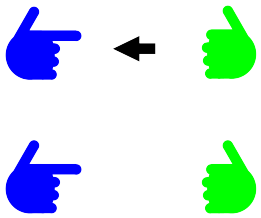
Marie et Baptiste commencent chacun avec 2 mains de 1 doigt :



Marie prend sa première main (1) et l'ajoute à sa seconde ($1 + 1 = 2$) :

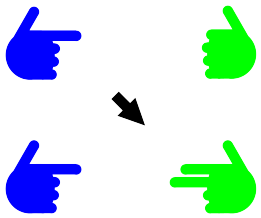


Baptiste ajoute sa première main (1) à la première main de Marie ($1 + 1 = 2$) :

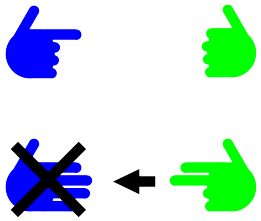


[1] Les joueurs venant de toute la galaxie, le nombre de mains est donc variable.

Marie réplique avec sa première main (2) sur la seconde de Baptiste ($2 + 1 = 3$) :

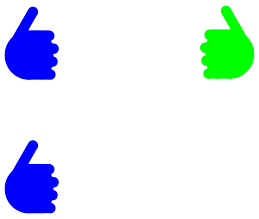


Baptiste prend sa seconde main (3) et l'ajoute à la seconde main de Marie (3+2 = 5), celle-ci est donc perdue :



Arrêtons l'exemple ici bien que la partie ne soit pas terminée.

Question 1 (1 point)

> Déroulez une partie avec la configuration suivante. Vous devez faire gagner le joueur de gauche en un minimum de coups. C'est lui qui commence la partie. Uniquement dans cette question, le joueur de droite ne pourra s'empêcher de faire disparaitre une de vos mains s'il peut le faire dans le tour.
>
> 
>
> Si vous ne savez pas comment représenter la partie, écrivez de la manière suivante :
>
> 1 1
> 1 0

Question 2 (1 point)

> Reprenons la partie précédente. Supposons que le joueur de droite commence, donnez un exemple de partie où le joueur de droite gagne.

Question 3 (3 points)

> A Décrivez une structure de données pour représenter une main.
>
> B Puis une pour représenter un joueur avec autant de mains qu'on le souhaite.
>
> C Enfin une dernière pour représenter une partie avec plusieurs joueurs.

Question 4 (1 point)

> Écrivez un algorithme permettant de s'assurer que la valeur d'une main reste dans l'intervalle $[1; 5]$. Il prend en entrée une main et met à jour sa valeur.

Question 5 (3 points)

Comptons un peu !

> A En combien de coups minimum une partie entre 2 personnes de deux mains peut-elle finir ? Expliquer.
>
> B Depuis une situation quelconque, de combien de manières peut-on obtenir une main de valeur 2 après une addition d'une main à celle concernée Écrivez-les.

Question 6 (2 points)

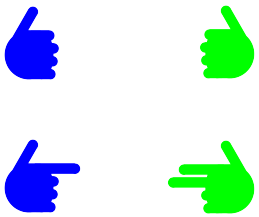
> Au cours d'une partie, Marie et Baptiste se retrouvent avec une main chacun, est-il possible d'arriver à une situation où les valeurs bouclent ?

Question 7 (2 points)

> Réalisez un algorithme permettant de connaitre toutes les combinaisons différentes de mains possibles au prochain tour. Cet algorithme prend une partie p (comme définie à la question 3.C), le nombre de joueurs n et un nombre de mains k sachant que c'est au joueur j de jouer et renvoie une liste de parties.

Question 8 (1 point)

> Combien existe-t-il de combinaisons différentes[2] de la partie ci-dessous après 2 ajouts ? C'est au joueur de gauche de commencer.



Question 9 (2 points)

> Réalisez un algorithme pour connaitre les combinaisons possibles d'une partie p après n actions, sachant que c'est le joueur j qui commence et qu'il y a n joueurs avec k mains.

Question 10 (3 points)

> Proposez un algorithme ou une technique afin de gagner et de devenir le maitre du "jeu là, tu sais avec les mains"[3].

[2] (1, 2) et (2, 1) étant considérées comme la même combinaison.
[3] Ce qui ne vous empêchera pas de perdre au Jeu.

3 Barrer le T

Après avoir joué tous les matchs possibles qui se finissent (ça a été long), nos deux organisateurs laissent leurs mains de côté et se lancent dans un jeu plus tactique sur une grille infinie.

Le principe du jeu est le suivant : sur une grille, on place un segment originel de longueur n, chaque joueur va ensuite, à tour de rôle, placer un segment dit fils. On dit d'un segment qu'il est fils d'un autre segment s'il est directement en contact avec cet autre segment. Le segment père est celui sur lequel est attaché les segments fils.

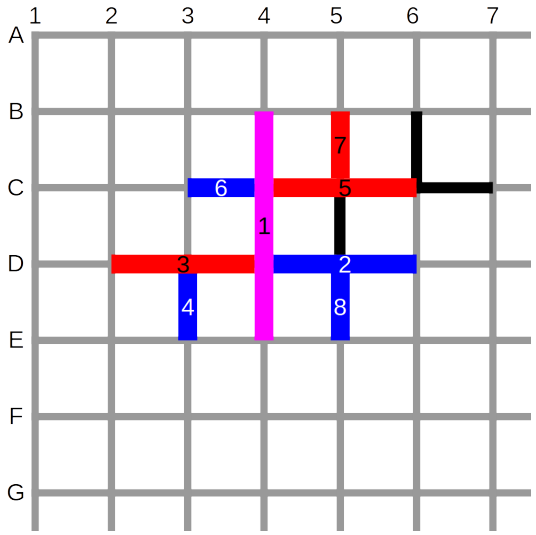
Quelques règles régissent la pose de segment :

- Les segments fils doivent avoir une longueur strictement inférieure à celle du segment père.
- Les segments fils ne peuvent pas être attachés à une extrémité de leur père.
- Un segment fils ne peut avoir qu'un père (il ne peut donc être posé au contact de plusieurs segments).
- Les segments fils sont posés perpendiculairement au segment père.

On dit qu'une partie est finie lorsque l'on ne peut plus poser de segment.

Le ou la gagnant(e) est celui ou celle qui a posé le dernier segment.

Exemple d'une partie finie où les segments sont numérotés dans l'ordre de pose :



Le segment 4 est le segment fils du numéro 3. Le segment 2 est le segment père du numéro 8.

Notons également qu'ici le joueur bleu a gagné, étant le dernier à avoir posé un segment (le 8).

Entre autres, dans cet exemple nous ne pouvons pas placer de segment entre $C5$ et $D5$ car il serait relié des deux côtés. Les segments ($C6$, $C7$) et ($B6$, $C6$) ne sont également pas possibles car ils sont au bout du segment numéro 5. Enfin, le segment 3 ne peut pas faire 3 cases de long car sa taille ne serait pas strictement inférieure au segment 1.

Question 11 (1 point)

> Proposez une structure de données permettant de représenter un segment.

Question 12 (2 points)

> Marie propose la victoire par longueur : le ou la gagnant(e) est celui ou celle qui a la longueur totale des segments la plus élevée. Dans l'exemple ci-dessus il y aurait une égalité car les deux joueurs ont 5 cases de longueur totale chacun (le joueur rouge et le joueur bleu).
>
> On cherche à gagner par longueur, proposez une technique simple.

Question 13 (1 point)

> Pour la victoire par longueur, existe-t-il une situation où mettre un segment de longueur non maximale est une bonne idée ?

Question 14 (2 points)

> Baptiste trouve que la victoire par longueur n'est pas une victoire équitable, mais Marie n'est pas d'accord. C'est à vous de régler ce débat :
>
> A La probabilité de gagner est-elle la même en commençant ou non la partie ? Expliquez.
>
> B Qu'en est-il de la victoire par tour ? Expliquez, encore.

Question 15 (1 point)

> Les feuilles de papier n'étant pas infiniment grandes, les organisateurs se demandent quelle est la hauteur et la largeur maximales que peut prendre une partie pour un segment originel donné.
>
> Proposez un encadrement ou une valeur exacte en fonction de la taille du premier segment.

On souhaite maintenant retrouver dans quel ordre aurait été joué une partie.

Question 16 (1 point)

> Rédigez un algorithme qui retourne tous les segments ayant pu être posés lors du dernier coup ne sachant pas quel joueur à posé le dernier segment. La partie n'est pas forcément finie.

Question 17 (2 points)

> Proposez un algorithme qui retourne toutes les suites de coups permettant d'arriver à la partie donné sachant que nous savons quel joueur a posé chaque segment.

4 mot mot mot t'eusses

Vos organisateurs préférés ont fini de faire des lignes partout (mais vraiment partout). Ils décident donc d'allumer la télé et de regarder leur émission favorite : Motus. Ils veulent donc faire quelques parties.

Rappelons un peu les règles. L'objectif est de retrouver un mot. La longueur et la première lettre sont données.

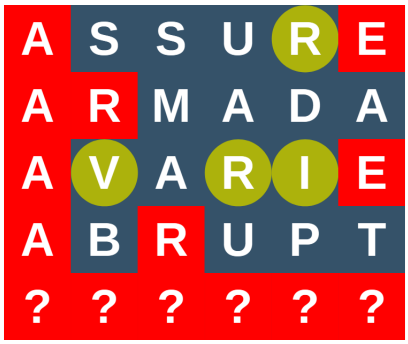
Lorsqu'une lettre du mot proposé est bien placée, elle est identifiée par un carré rouge et lorsqu'elle est dans le mot à découvrir mais mal placée, on l'identifie avec un rond jaune. Les lettres sur fond bleu ne sont pas dans le mot. Il ne faut que proposer des mots qui existent dans le dictionnaire. Un mot est trouvé quand toutes les lettres sont dans des carrés rouges.

Bien sûr ce n'est pas si simple, il y a un nombre limité d'essais.

Voici un exemple de partie où le mot à trouver est "Prologin" (les mots sont listés dans l'ordre de proposition) :

Question 18 (1 point)

> À vous de jouer, trouvez le mot recherché :



Question 19 (2 points)

> Proposez une structure de données afin de représenter les informations que vous avez sur le mot recherché au cours de la partie.

Question 20 (1 point)

> À l'aide des deux mots ci-dessous, dites quel est le meilleur mot à proposer parmi : "Diner", "Fixer", "Forge", "Crêpes" et "Fusée". Justifiez.

Question 21 (1 point)

> A partir d'un ensemble de mots (le dictionnaire) et des informations sur le mot recherché (rappelez-vous de la question 20), écrivez un algorithme qui renvoie la liste des mots potentiellement correctes.

Question 22 (3 points)

> A partir d'un dictionnaire contenant tous les mots possibles, de la première lettre et de la longueur du mot à trouver, proposez un algorithme permettant de trouver ce fameux mot en un minimum de coups.

5 Organisateurs organisés

5.1 De père en fils

Il se trouve que le jeu de la seconde partie de ce sujet est un réel succès. Plusieurs tables du local sont donc réservées aux duels de segments. Ethan est désigné responsable de l'organisation.

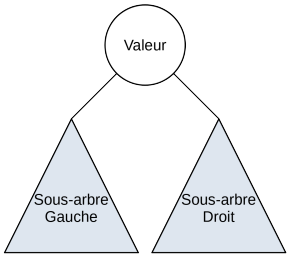
Comme toute personne organisée, il commence par trier les crayons du plus petit au plus grand, sachant que tous les crayons ont des tailles différentes[4].

Pour les ranger il choisit d'utiliser un arbre binaire de recherche (ABR).

Le principe est simple, chaque nœud (ici un crayon) a 2 fils : le fils gauche et le fils droit. Les fils sont également des nœuds.

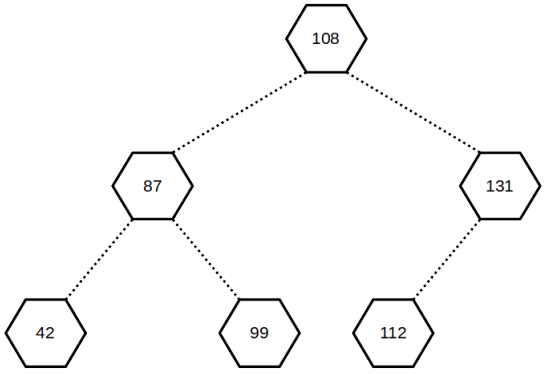
Le fils gauche est un nœud avec une valeur strictement inférieure à son père, et le fils droit un nœud d'une valeur strictement supérieure. Ces fils eux aussi peuvent avoir des enfants. Le fils gauche, ses enfants, ses petits enfants, etc. sont appelés : sous-arbre gauche. C'est également le cas pour le fils droit et ses descendants nommés le sous-arbre droit.

Voici un nœud :



L'arbre se construit à partir d'un premier élément : le nœud racine. À chaque ajout d'un crayon, on le compare à un nœud (en premier le nœud racine) pour savoir s'il est plus petit ou plus grand. Ensuite on le pose comme fils s'il n'y en a pas à cette place, mais s'il y en a déjà un, on réitère l'étape précédente en prenant le dit fils comme nœud comparant.

Voici un exemple d'ABR où les crayons ont été ajoutés dans cet ordre 108, 87, 131, 99, 42, 112 :



Question 23 (1 point)

[4]Et si ce n'était pas le cas, il s'est arrangé pour que ça le soit.

> Faites l'ABR des longueurs de crayons dans l'ordre suivant : 135, 87, 119, 143, 100, 72 et 121.

Question 24 (1 point)

> Décrivez une structure de données pour représenter les nœuds.

Question 25 (2 points)

> Énoncez un algorithme qui recherche l'existence ou non d'un crayon dans un arbre. Cet algorithme prend la longueur du crayon recherché et un arbre en entrée.

Question 26 (1 point)

> Écrivez un algorithme permettant d'insérer un crayon à sa place dans un arbre donné. L'algorithme prend en paramètre un crayon et le nœud racine de l'arbre. Si la longueur est déjà dans l'arbre, il ne faut rien faire.

Question 27 (2 points)

> Proposez un algorithme permettant de supprimer un nœud s'il existe, sinon il ne doit rien faire. Ses paramètres sont la longueur du crayon à supprimer et un arbre.

5.2 Il y a du bouleau

Maintenant que tous les organisateurs ont vu Ethan ranger les crayons par ordre de taille, il faut qu'il mette en place un calendrier pour la répartition des tables de jeux.

Afin d'organiser les réservations, il utilise l'algorithme des nœuds chapeaux.

Cet algorithme utilise un arbre binaire[5] où chaque feuille[6] représente une période. Cet algorithme permet de partager des ressources, en l'occurrence les tables de duels.

On appelle un nœud chapeau d'une réservation, un nœud qui vérifie deux conditions :

- Toutes ses feuilles descendantes[7] font partie de la réservation.
- Ce nœud est le nœud racine, ou bien le père a une feuille descendante ne faisant pas partie de la réservation.

Chaque nœud a une valeur représentant la quantité de ressources réservées par lui-même ou ses descendants.

Soit q la quantité de ressources d'un nœud. La quantité de ressources d'un nœud est donc :

$q(noeud) = qf(noeud) + qn(noeud)$

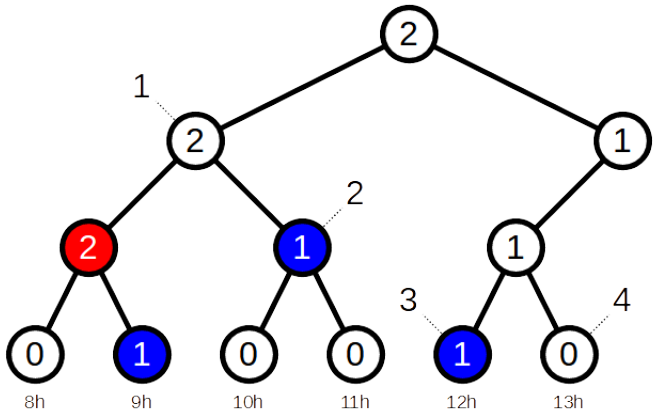
Soit qf le maximum des quantités de ressources réservées des enfants :

$qf(noeud) = max(q(\text{fils gauche}), q(\text{fils droit}))$

Soit qn la somme des quantités réservées par les réservations ayant ce nœud comme nœud chapeau.

Voici un exemple où il n'y a que deux tables à réserver et deux réservations, une de 8h00 à 9H59 et l'autre de 9h00 à 12h59 :

[5] c'est à dire un arbre dont les nœuds ont 2 fils maximum
[6] Une feuille est un nœud qui n'a pas de fils.
[7] filles ou petites filles ou petites petites filles...



Le nœud rouge est le nœud chapeau de la première réservation et les bleus ceux de la seconde réservation.

Le nœud rouge a une valeur de $2 = max(0, 1) + 1$.

Le nœud "1" est un nœud quelconque qui a la valeur 2 car le maximum de ses fils est 2.

Les nœuds "2" et "3" sont des nœuds chapeaux de la seconde réservation.

Les nœuds "3" et "4" sont des feuilles.

Question 28 (1 point)

> Proposez une structure de données afin de représenter un arbre.

Question 29 (1 point)

> Ethan est embêté car il ne sait pas quels sont les nœuds chapeaux pour une réservation à faire. Proposez un algorithme qui renvoie tous les nœuds chapeaux dans un intervalle de temps donné. Il prend le nœud racine de l'arbre et le début et la fin de cet intervalle.

Question 30 (2 points)

> Maintenant le responsable des duels souhaite savoir combien il reste de réservations à prendre.
>
> Écrivez un algorithme qui recherche le nombre de tables disponibles lors d'une période. Il prend en entrée le nœud racine de l'arbre r, le nombre total n de tables ainsi que le début et la fin de la période.

Question 31 (2 points)

> Faites une fonction permettant de réserver une période si elle est disponible. Elle prendra un arbre, le début et la fin de la période à réserver et le nombre de tables à réserver.

Question 32 (2 points)

> Il se trouve que ce jeu reste populaire plus longtemps que prévu, il faut donc décaler le calendrier afin de pouvoir réserver pour les prochains jours.
> Proposez un algorithme permettant d'enlever les nœuds passés pour en proposer plus tard.

5.3 Arbre intervallaire

Après quelques parties, Ethan se rend compte que découper les périodes en heures n'est assez précis. Effectivement une partie avec un segment originel de longueur 3 est beaucoup plus courte qu'une autre avec un segment

de longueur 12. Il décide donc d'utiliser des intervalles précis à la minute ainsi qu'un arbre intervallaire centré.

La minute où commence l'intervalle est nommé le point de début. Le point de fin est la minute (incluse) où se termine l'intervalle.

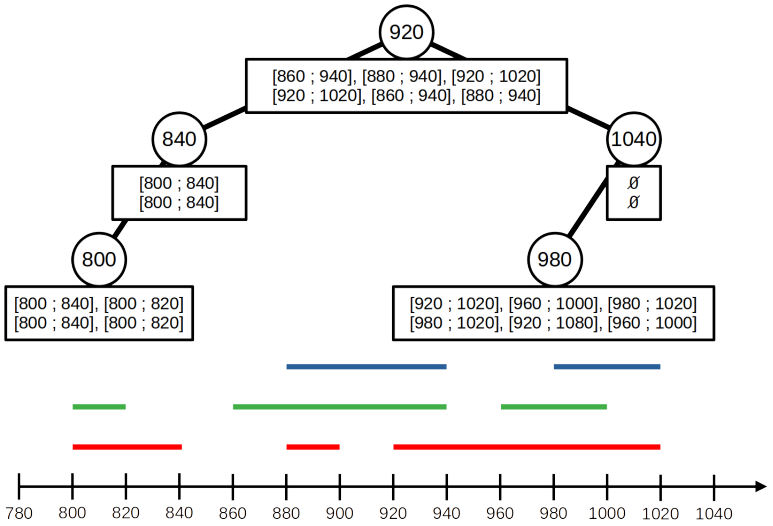
Un nœud se définit avec :

- La valeur du point central, c'est-à-dire la minute que représente ce nœud.
- Un fils gauche (étant un arbre intervallaire).
- Un fils droit (étant également un arbre intervallaire).
- Les intervalles incluant ce point, triés par ordre croissant du point de début.
- Les intervalles incluant ce point, triés par ordre décroissant du point de fin.

Le choix des points centraux est arbitraire.

L'intérêt des arbres intervallaires et de pouvoir rapidement trouver des intersections de points et d'intervalles. Il faut donc s'assurer de garder l'arbre optimisé, c'est-à-dire avec le minimum de nœuds.

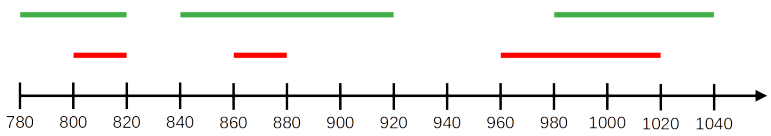
Voici un exemple d'un arbre intervallaire non optimisé avec les intervalles de réservation de 3 tables représentés en dessous (dans les rectangles se trouve les deux listes) :



L'arbre ci-dessus n'est pas optimisé car le nœud 1040 est inutile, n'étant inclus dans aucun intervalle.

Question 33 (1 point)

Pour s'entrainer, Ethan commence avec 2 tables, faites un arbre intervallaire de ces intervalles :



Question 34 (1 point)

Proposez des structures de données afin de représenter l'arbre et les intervalles.

Question 35 (1 point)

> Il faut qu'Ethan trouve les intervalles qui passent par un point donné, aidez-le avec un algorithme.

Question 36 (2 points)

> Faites un algorithme qui renvoie toutes les intersections entre les intervalles d'un arbre et un intervalle donné.

Question 37 (1 point)

> Ethan remarque que certaines personnes souhaitent annuler leurs réservations.
>
> Faites un algorithme qui permet de supprimer un intervalle donné d'un arbre.

Question 38 (2 points)

> Idotno vient d'annuler sa réservation, n'est plus inclus dans aucun intervalle.
>
> Écrivez un algorithme qui permet de supprimer un nœud donné.

6 Il y en a un peu plus

Question bonus 39 (1 point)

> Donnez des noms aux deux premiers jeux.

Question bonus 40 (1 point)

> Dessine-moi un thon mou.

Question bonus 41 (3 points)

> Écrivez une saynète entre Joseph et Baptiste Marchand.

Question bonus 42 (2 points)

> Énoncez les règles du jeu du cul de chouette (en ignorant les règles à l'Aquitaine).