Concours Individuel National d'Informatique

Sujet de finale 1999

Patapaint

1 Sujet

1 La petite histoire

Bienvenue sur LaHaag 42, station spatiale de retraitement de déchets toxiques en exploitation conjointe par plusieurs compagnies. LaHaag, c'est la fine fleur du retraitement, euh... décentralisé (entre Beltégeuse et Alpha du Centaure, on peut le dire !). Cuves anti-radiations à blindage de catégorie 5A, systèmes de décontamination à particules rapides, panneaux de capture des neutrinos émis pour recyclage en énergie libre... La crème. L'élite. Les tout meilleurs... Jusqu'à aujourd'hui.

Ce matin, l'ensemble de la partie habitée de la planète s'est trouvé réveillé par une sirène stridente, assourdissante, implacable. Dans chaque base, l'ensemble de la communauté s'est précipité en salle de briefing, et la nouvelle tombe, écrasante dans ses implications : la Grande Centrale s'est fissurée. Une réaction inattendue, hors de toute manipulation spécifique.

La Grande Centrale, c'est la cuve générale utilisée tour à tour par le personnel technique de chaque base pour décontaminer les déchets les plus dangereux. La construction d'une cuve capable d'arrêter net un tel niveau de radiation ne pouvait être financee en totalite par une seule base.

Le problème, c'est que des déchets un peu "violents" étaient entreposés dans la cuve. "Étaient ?" Oui, car la fissure a endommâgé la structure de la cuve, qui s'est ébréchée en de multiples endroits, notamment aux sas menant aux différentes bases. D'ici à quelques heures, toute habitation sera devenue plus invivable que la centrale de Chernobyl juste après l'incident qui l'a rendue célèbre. Il faut évacuer. Et vite.

Seulement voilà, pour évacuer, il faut appeler la maison-mère, histoire qu'ils viennent chercher les occupants de leur base. Or le matériel de communication, pourtant destiné expressément à ce type de base, ne fonctionne plus en raison du niveau déjà élevé de radiations à la surface, hors des bâtiments heureusement "protégés". Il ne reste plus que le dispositif VibraPaint, encore un équipement partagé par les différentes bases.

Le VibraPaint est une large surface hexagonale composée de nombreux triangles. On utilise des patatos (prononcez "patatôssé"), pour y former un motif. Les patatos sont des hamsters (très) mutants (eh oui, les premiers temps la planète n'était pas très clean) qui sont très utiles : ils se gonflent comme des éponges et bavent de partout (Ah ? et c'est utile, ça ?). On les aide en les fourrant de mousse (demandez pas comment...) et on les jette sur le VibraPaint.

Les nuances (infâmes) de couleur produites par le passage de patatos sur les dalles du VibraPaint sont transformées en vibrations qui composent un signal à très longue portée, capté par toutes les maisons mères. Chaque base dispose d'un motif d'urgence : une maison-mère recevant le motif de sa base envoie immédiatement une navette de rappatriement. Uniquement pour sa propre base.

Chaque base a dépêché le membre de son personnel le plus qualifié pour manipuler le VibraPaint. Plusieurs communautés. Un seul VibraPaint. Et chacun pour soi lorsqu'il s'agit de le manipuler. Le compte à rebours a commencé : sauvez les vôtres !

2 Structure du terrain (le VibraPaint)

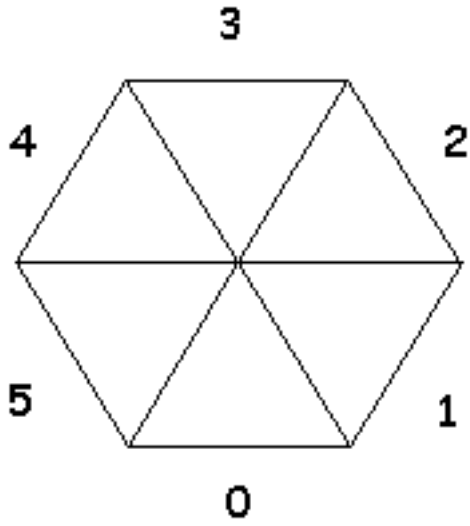
2.1 Vue aérienne

Le terrain est, vu de dessus, un hexagone régulier constitué de nombreuses dalles triangulaires équilatérales, chaque dalle étant une paroi élastique tendue entre trois piquets. Des triangles adjacents ont des piquets en commun. Suivant la position d'un piquet sur le terrain, il peut y avoir 2, 3 ou 6 dalles dont un sommet est fixé à ce piquet : 2 si le piquet se trouve sur un sommet de l'hexagone, 3 s'il se trouve sur une arrête, et 6 dans les autres cas.

On définira le côté d'un hexagone comme étant le nombre d'arrêtes de dalles constituant chaque arrête de l'hexagone. Un hexagone de côté 1 est donc un assemblage de 6 triangles équilatéraux, fixés sur 7 piquets : un central sur lequel sont fixés tous les triangles, et 6 piquets correspondant aux 6 sommets de l'hexagone, chacun de ces 6 piquets étant utilisé par 2 dalles. Un hexagone de côté 2 contiendra ainsi 24 dalles, et 19 piquets. A vous de trouver la formule si vous voulez savoir combien de dalles et de piquets contient un hexagone de côté n. Cherchez-la avant de lire la suite du sujet, cela vous aidera à suivre. Pour vérifier, sachez qu'un hexagone de côté 42 contient 5419 piquets et 10584 dalles.

Les hexagones manipulés dans le reste du sujet seront représentés de telle sorte que deux côtés opposés soient horizontaux. Les 4 autres côtés sont donc représentés avec un angle de 60 degrés par rapport à l'horizontale. Ainsi, chaque dalle possède une arrête représentée horizontalement.

Pour plus de commodité, on numérotera les côtés de l'hexagone de 0 à 5 : le côté horizontal du bas est le côté 0 et les autres suivent, en sens inverse des aiguilles d'une montre (sens trigonométrique).



2.2 Les piquets

Tout ce qui précède est, si vous avez bien suivi, ce qu'on voit en regardant le terrain du dessus. La réalité est un peu plus complexe : le tout n'est pas plat, loin de là ! En fait, chaque piquet est un vérin hydraulique, dont l'extrémité (là où sont fixés les sommets des dalles) peut se déplacer sur un axe vertical.

Lorsque les 3 piquets sur lesquels est fixée une dalle sont au même niveau, celle-ci est un triangle équilatéral. Mais dès qu'un des piquets (c'est-à-dire un vérin) est actionné, disons vers le haut, le triangle ne reste plus du tout équilatéral : la paroi élastique se tend, et les arrêtes correspondant au piquet modifié s'allongent.

Rassurez-vous, vous n'aurez pas à vous occuper des déformations des dalles, de la résistance de la paroi, etc. Tout ce qui vous préoccupera pendant les dizaines d'heures qui viennent est l'inclinaison de ces dalles !

2.3 Coordonnées

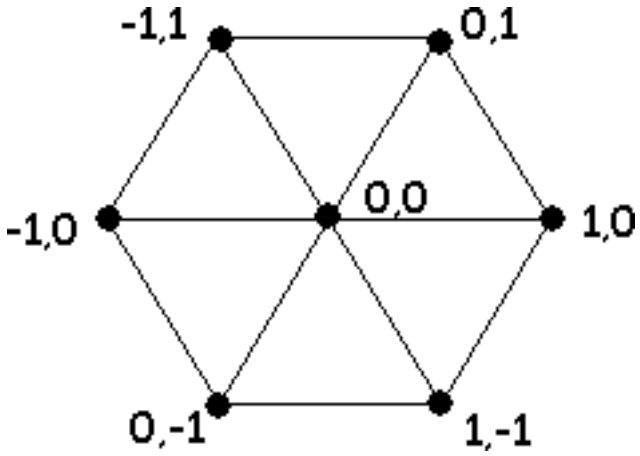
Pour manipuler ce terrain, on va définir un système de coordonnées, permettant de spécifier un piquet donné, ou une dalle donnée. Dans tout le reste du sujet, ce système de coordonnées qui sera utilisé. Vous devrez aussi utiliser ces coordonnées pour communiquer avec le serveur.

2.3.1 Coordonnées des piquets

On utilisera comme origine de notre repère le piquet au centre de l'hexagone. Ce piquet a donc les coordonnées (0,0) : abscisse = 0, ordonnée = 0, ou même x = 0, y = 0.

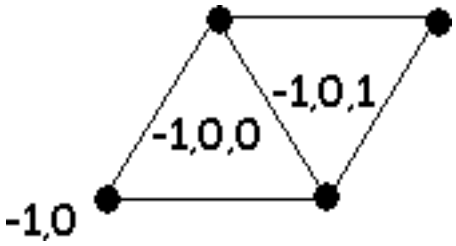
Les abscisses augmentent lorsqu'on va vers l'est (0 degrés sur le cercle trigo, 3 heures), et les ordonnées lorsqu'on va vers le nord-est (60 degrés / 1 heure).

Pour un hexagone de côté 1, les coordonnées des piquets sont donc, pour les piquets des sommets, dans le sens inverse des aiguilles d'une montre, en partant du piquet le plus a l'est : $(1,0),(0,1),(-1,1),(-1,0),(0,-1),(1,-1)$. (Le premier chiffre correspond à l'abscisse, ou x, et le deuxième chiffre correspond à l'ordonnée, ou y.)



2.3.2 Coordonnées des dalles

Pour repérer les dalles, on utilisera trois coordonnées, x, y, et t. Prenons un piquet de coordonnées (x,y). Soit un losange, formé de deux dalles, orienté Nord-Est à partir du piquet. Les coordonnées des deux dalles formant le losange sont alors (x,y,0) pour celle qui touche le piquet d'origine, et (x,y,1) pour l'autre :



3 Actions sur le terrain

La seule chose que vous puissiez contrôler sur le terrain est la hauteur des différents piquets. Celle-ci est nulle au début de la partie, pour tous les piquets (le terrain est plat), et peut ensuite varier entre -42 et 42.

Le principe est simple : vous pouvez augmenter ou diminuer la pression d'huile dans les vérins. Vous avez donc accès à une commande pour diminuer, ou augmenter d'un certain nombre d'unités la hauteur d'un piquet donné (voir La librairie).

Le seul problème est que vos adversaires aussi peuvent régler la hauteur de ces piquets, puisqu'ils ont accès aux mêmes commandes que vous !

Imaginons que vous soyiez deux à vouloir régler la hauteur d'un piquet donné, ayant une hauteur de 0. Vous voulez la diminuer de 30 unités, et votre adversaire veut l'augmenter de 26 unités. Au final, la hauteur du piquet n'aura diminué que de 4, et sera donc de -4.

Cependant, rien ne vous empêche, si vous tenez absolument à ce que le piquet baisse d'au moins 30, à demander à ce qu'il baisse de beaucoup plus. En l'occurence, en donnant une commande de descente du piquet de 56 unités, si votre adversaire commande toujours une montée de 26, le piquet atteindra finalement une hauteur de -30 unités.

Bien sûr, il y a une limite au nombre d'unités que vous pouvez utiliser pour faire descendre, ou monter les piquets. Cette limite porte sur le nombre total d'unités utilisées par tour : elle est égale au nombre de piquets du terrain.

De plus, en aucun cas, le piquet ne pourra descendre ou monter au-delà de -42 , respectivement 42 unités. Si vous donnez un ordre de descente de 400 et votre adversaire un ordre de montée de 26, le piquet s'arrêtera à -42. Un ordre de montée de 1 au tour suivant (voir Déroulement de la partie) suffira à le faire remonter à une hauteur de -41. Notez que la limitation à -42 ou 42 n'est appliquée qu'une fois le total des ordres des différents joueurs effectué pour ce piquet.

Pour simplifier, on n'utilisera qu'une seule commande, à laquelle on passera le déplacement, celui-ci étant négatif pour une descente, et positif pour une montée.

4 Vous et les patatos

4.1 Pour eux, vous êtes un lâcheur

Vous êtes l'expert envoyé par votre base pour manipuler le VibraPaint. Vous vous trouvez à l'extérieur du terrain, sur le bord de l'un des 6 côtés de l'hexagone. Tout ce que vous pouvez faire, en plus de contrôler la hauteur du terrain, consiste à remplir des patatos et à les lâcher sur le terrain, devant vous.

Vous pouvez placer les patatos sur toute dalle dont l'une des arêtes longe le côté qui vous est attribué sur l'hexagone. Evidemment, vous ne pouvez y placer de patatos que s'il n'y en a pas déjà sur la dalle.

De votre point de vue, vous vous situez toujours du côté de numéro 0. Le serveur s'occupe de faire toutes les conversions de coordonnées pour donner l'illusion à tous les joueurs qu'ils se trouvent de ce côté.

4.2 Déplacement des patatos

Une fois un patatos posé sur le terrain, vous n'avez plus de contrôle dessus, excepté par l'intermédiaire de la modification du terrain.

En effet, les patatos se déplacent sur le terrain en sautant d'une dalle à l'autre, en fonction de l'inclinaison de celle sur laquelle ils se trouvent. A chaque tour, un patatos peut soit rester sur sa dalle, soit essayer de sauter sur l'une des trois dalles adjacentes. Les règles de déplacement sont les suivantes :

- Si l'un des 3 piquets a une hauteur strictement supérieure à celle des deux autres piquets, le patatos va tenter de sauter dans la direction opposée au sommet correspondant (donc il saute dans le sens de la pente de la dalle). Sinon, il ne bouge pas.
- Si plusieurs patatos tentent de sauter sur la même dalle, ils rebondissent les uns contre les autres, et retombent sur leurs positions de départ respectives.
- Si un patatos A tente de sauter sur une dalle où se trouve un patatos B soit immobile, soit qui n'a pas réussi à sauter où il voulait, le patatos A rebondit sur B, et retourne à sa position initiale.
- Si deux patatos tentent d'échanger leur position, ils rebondissent l'un sur l'autre, et retournent donc à leurs positions initiales.

Dans tous les cas, on ne peut avoir deux patatos sur la même dalle. Toute tentative de déplacement qui n'entre pas dans l'un de ces 4 cas est effectuée. Donc si un patatos tente de sortir du terrain, il est éliminé !

4.3 Remplissage des patatos

Avant de lâcher des patatos sur le terrain, vous devez les remplir. Un patatos est rempli d'une sorte de mousse, dont vous avez une réserve importante,puis de fluide coloré, sous la forme d'un mélange de trois couleurs de base : rouge, vert, et bleu. Ce fluide permet aux patatos d'émettre de la lumière de n'importe quelle couleur. Vous disposez également de réserves importantes (mais pas inépuisables) de fluides des trois sortes. Les limites standard sont les suivantes :

- Pour les patatos : côté^2/2. Pour un VibraPaint de côté 8, 32 patatos (par expert)
- Mousse et couleurs de peinture : aucune (implicite en fonction de celle des patatos)

Vous pouvez remplir des patatos avec une quantité quelconque de ces quatre ingrédients de base. La seule restriction est que le volume total du patatos ne doit pas excéder 1536 unités.

Lors du bourrage d'un patatos, les ingrédients sont ajoutés dans l'ordre suivant : mousse, rouge, vert, bleu. Lorsque le patatos est plein (c'est-à-dire que vous atteignez le volume maximal), les ingrédients que vous tentez d'ajouter sont perdus. Si vous tentez de mettre, pour un ingrédient donné, une quantité supérieure à celle qui vous reste dans votre réserve, tout le contenu de la réserve concernée est vidé, et le volume ajouté au patatos est égal à la quantité ainsi prélevée.

Vous pouvez lâcher un patatos par tour, pas plus. Vous pouvez aussi choisir de ne pas en lâcher.

4.4 Transferts de fluide

Une fois lâché sur le terrain, un patatos a tendance à laisser du fluide partout lorsqu'il est plein, ou bien à aspirer du fluide, lorsqu'il est vide. Lorsqu'un patatos se trouve sur une dalle, un transfert de fluide se fait, à chaque tour, entre le patatos et la dalle. Ce transfert modifie leurs composantes de couleur. La quantité de fluide transféré dépend du volume de fluide présent sur la dalle, du volume de fluide contenu par le patatos, et de la capacité totale du patatos (c'est-à-dire le volume maximal d'un patatos, moins la quantité de mousse qu'il contient).

Le volume transféré à chaque tour est calculé de la manière suivante :

```
patatos.capacité = 1536 - patatos.mousse
patatos.fluide = patatos.rouge + patatos.vert + patatos.bleu
dalle.fluide = dalle.rouge + dalle.vert + dalle.bleu
si dalle.fluide <= patatos.capacité / 2
transfert = MIN(10, patatos.capacité / 2 - dalle.fluide, patatos.fluide)
sinon transfert = - MIN(10, dalle.fluide - patatos.capacité / 2, dalle.fluide, patatos.capacité - patatos.fluide)
fin si
```

Le transfert correspond à la quantité de fluide qui passe du patatos vers la dalle. Si ce transfert est négatif, le fluide passe de la dalle dans le patatos. Le fluide transféré contient les mêmes proportions de rouge, vert, et bleu, que le fluide contenu dans la source.

On a tendance à avoir un équilibre qui s'établit entre la moitié de la capacité en fluide du patatos, et le volume de fluide de la dalle. Tant que cet équilibre n'est pas établi, le patatos dépose/aspire tout ce qu'il peut jusqu'à rétablir l'équilibre, et dans la limite de 10 unités de fluide par tour.

Vous remarquerez qu'un patatos qui n'a pas été bourré avec de la mousse ne peut jamais aspirer de fluide, et que la quantité de fluide présent sur une dalle ne peut pas dépasser 768 unités.

4.5 Ce que vous voyez

Vous pouvez à tout moment regarder le terrain. Vous pouvez voir :

- Pour chaque piquet :
  - sa hauteur
- Pour chaque dalle :
  - sa couleur

- la présence ou non d'un patatos
- la couleur de cet éventuel patatos

Les composantes rouge, verte et bleue de la couleur d'une dalle correspondent aux volumes de fluides des trois sortes présents sur cette dalle, lorsque ceux-ci ne dépassent pas 255, ou 255 sinon.

Les composantes des dalles sont stockées avec une bonne précision, mais la couleur que vous voyez n'est que la valeur entière de ces composantes (lorsqu'elles ne dépassent pas 255).

Le calcul est le même pour la couleur d'un patatos.

4.6 Votre objectif

Votre but est qu'à la fin de la partie, le terrain ressemble au maximum au motif de votre signal. Le joueur dont le motif est le plus ressemblant gagne la partie.

La distance entre le motif d'un joueur et l'apparence du terrain est calculée selon le principe suivant :

total = 0
pour chaque dalle faire
total = total + (dalle.rouge - motif.rouge)^2 + (dalle.vert - motif.vert)^2 + (dalle.bleu - motif.bleu)^2
fin pour
distance = racine_carree(total)

où dalle.couleur représente la valeur de la composante couleur de la dalle considérée, et motif.couleur représente la valeur de la composante couleur de la case du motif considérée. Les patatos présents sur le terrain n'influencent donc aucunement ce total.

On calcule à partir de cette distance un taux de ressemblance :

Taux = 1 - (distance / distance_max)

Où distance_max est la distance la plus grande que l'on puisse obtenir pour le motif considéré.

Un taux de 1 signifie que le terrain est identique au motif. Un taux de 0 signifie que le terrain est une des images moins ressemblantes au motif que l'on puisse imaginer. Le gagnant est donc le joueur pour lequel ce taux est le plus proche de 1.

4.7 Déroulement d'une partie

Vous pouvez envoyer vos commandes dans n'importe quel ordre, mais le serveur les gère toujours de la manière suivante :

pour Tour allant de 1 à 1000 faire
Récupérer les commandes des différents joueurs
Effectuer les transferts de fluide
Faire sauter les patatos
Règler la hauteur des piquets
Lâcher les patatos (un au maximum par joueur)
fin pour
Déterminer le gagnant

5 Ce que vous devez faire

5.1 Votre code

Vous ne devez pas écrire le corps du programme, car il est déjà contenu dans la librairie qui vous est fournie. Vous devez simplement écrire trois fonctions, qui sont appelées par la librairie :

- Une fonction d'initialisation, appelée au lancement du programme : InitGame.
- La fonction appelée à chaque tour : playTurn.
- Une fonction qui vous permet de donner le nom de votre champion, appelée au lancement du programme : getClientName. Elle prend en paramètre une chaîne de caractères (le nom du champion doit être le même que celui qu'on vous demandera dans le fichier de configuration du métaserveur).

5.2 Le code que nous fournissons

Pour obtenir des informations sur le terrain, vous disposez de fonctions pour connaître :

- La taille de l'hexagone (son côté) : getTilesPerSide
- La couleur d'une case du motif : getTargetTileColor
- La couleur d'une dalle du terrain : getTileColor
- La présence ou non d'un patatos sur une dalle du terrain : isPatatosThere
- La couleur du patatos présent sur une dalle du terrain : getPatatosColor
- La hauteur d'un piquet : getPegHeight
- Ce qui vous reste dans vos réserves de peinture et de mousse : getPaint, getFoam.

Pour donner vos ordres, vous disposez des fonctions suivantes :

- Faire monter/descendre un piquet : movePeg
- Lâcher un patatos : dropPatatos

5.3 Temps de réponse

Vous disposez à chaque tour de 100ms pour faire toutes vos opérations, et sortir de la fonction playTurn. Cependant, si vous dépassez ce temps, tout n'est pas perdu : vous disposez d'une réserve de temps de 30 secondes pour toute la partie. Lorsque vous dépassez 100ms lors d'un tour, le temps supplémentaire utilisé est puisé dans cette réserve. Lorsque cette réserve est épuisée, votre client n'est plus jamais appelé, mais n'est pas éliminé pour autant : son score sera comptabilisé à la fin dans tous les cas. Toutefois, votre temps de réponse ne peut en aucun cas excéder 2 secondes par tour.

Pour avoir des informations sur le temps qui vous reste à jouer, vous disposez de deux fonctions, pour obtenir :

- Le temps que vous avez déjà utilisé pour le tour (en millisecondes) : getElapsedTime
- Le temps qu'il vous reste dans votre réserve (en millisecondes) : getExtraTime

6 La librairie

Dans les deux paragraphes suivants, vous trouverez les prototypes C/C++ et Pascal des différentes routines fournies par la librairie, et que vous devez également fournir. Les fichiers utilisés pour Java et CamL viendront très prochainement. Dans l'intervalle, vous pouvez bien sûr commencer à mettre au point quelques algorithmes avec les outils Java/CamL standard.

6.1 Chez le serveur

Vous disposez de plusieurs fonctions auprès du serveur, accessibles grâce aux librairies client. Ces librairies existent pour différents langages, aussi, pour chaque fonction, vous verrez les prototypes pour chaque langage. Dans tous les cas, un certain nombre de fichiers vous est fourni, qui contiennent entre autres toutes les déclarations (voir plus loin). Dans ce qui suit, nous avons supprimé à des fins de clarté les directives diverses de convention d'appel, d'import, d'export, etc.

- extern void dropPatatos(/*in*/ int x, int y, int t, int density, int r, int g, int b);
  procedure DropPatatos(X, Y, T, Density, R, G, B : LongInt);

Lance un patatos sur le VibraPaint, aux coordonnées de dalle spécifiées par (X, Y, T). Density représente la quantité de mousse ; R, G et B sont les composantes en Rouge, Vert, Bleu de sa charge colorée.

- extern int getElapsedTime(void) ;
  function GetElapsedTime : LongInt ;

  Renvoie le temps écoulé depuis le début du tour, en millisecondes.

- extern int getExtraTime(void) ;
  function GetExtraTime : LongInt ;

  Renvoie le temps "extra" restant pour tous les tours à venir, y compris le tour courant. En millisecondes. Vous commencez avec une charge "extra" de 30 secondes.

- extern int getFoam(void) ;
  function GetFoam : LongInt ;

  Renvoie le niveau de la réserve de mousse.

- extern void getPaint(/*out*/ int* r, int* g, int* b) ;
  procedure GetPaint(var R, G, B : LongInt) ;

  Renvoie les niveaux de la réserve de peinture.

- extern void getPatatosColor(/*in*/ int x, int y, int t, /*out*/ int* r, int* g, int* b) ;
  procedure GetPatatosColor(X, Y, T : LongInt ; var R, G, B : LongInt) ;

  Renvoie les composantes colorées du patatos sur la dalle de coordonnées (X, Y, T).

- extern int getPegHeight(/*in*/ int x, int y) ;
  function GetPegHeight(X, Y : LongInt) : LongInt ;

  Renvoie la hauteur (entre -42 et 42) du piquet de coordonnées (X, Y).

- extern void getTargetTileColor(/*in*/ int x, int y, int t, /*out*/ int* r, int* g, int* b) ;
  procedure GetTargetTileColor(X, Y, T : LongInt ; var R, G, B : LongInt) ;

  Renvoie les composantes colorées, dans le motif que vous devez former, de la dalle de coordonnées (X, Y, T).

- extern void getTileColor(/*in*/ int x, int y, int t, /*out*/ int* r, int* g, int* b) ;
  procedure GetTileColor(X, Y, T : LongInt ; var R, G, B : LongInt) ;

  Même chose, mais pour le motif que vous êtes en train de former.

- extern int getTilesPerSide(void) ;
  function GetTilesPerSide : LongInt ;

  Renvoie la longueur de côté (en nombre d'arêtes de dalles) du VibraPaint.

- extern int isPatatosThere(/*in*/ int x, int y, int t) ;
  function IsPatatosThere(X, Y, Z : LongInt) : LongInt ;

  Teste si un patatos est présent sur la dalle en (X, Y, T). Renvoie 1 si c'est le cas, 0 sinon.

- extern void movePeg(/*in*/ int x, int y, int delta) ;
  procedure MovePeg(X, Y, Delta : LongInt) ;

  Change la hauteur du piquet de coordonnées (X, Y) en lui appliquant l'offset Delta. N'oubliez pas que la

hauteur est bornée.

- extern int startGame(int argc, char const * argv) ;
  function StartGame(ArgC : LongInt ; var ArgV : T3CStrings) : LongInt ;

  Appelée par le corps principal du programme. Ce corps vous est fourni, aussi vous n'appelez jamais cette fonction depuis votre code. Les arguments passés décrivent la ligne de commande de votre propre programme client, dont la librairie a besoin.

6.2 Chez vous

Quant à votre programme client, il doit implémenter les fonctions suivantes :

- extern char const *getClientName(void) ;
  function GetClientName : CString ;

  Doit renvoyer le nom de votre champion. Ce nom doit être identique à celui figurant entre crochets dans votre fichier /prolo.conf. Voir la documentation de prolo.conf pour les détails.

- extern void initGame(void) ;
  procedure InitGame ;

  Appelé au démarrage de votre client. Vous permet d'initialiser différentes données à vous, en récupérant par exemple le motif à obtenir, les quantités présentes dans les réserves, l'état actuel du VibraPaint (d'autres experts ont peut-être déjà commencé à le manipuler), etc.

- extern void playTurn(void) ;
  procedure PlayTurn ;

  Appelé pour chaque tour. N'oubliez pas que vous avez un temps limite pour le tour. Il n'existe pas de compteur de tour au niveau du serveur, ce serait une véritable perte de temps : si vous en voulez un, utilisez un compteur interne et incrémentez-le à chaque appel de votre routine. C'est au sein de cette routine que vous utiliserez l'essentiel des fonctions de la librairie.

7 Annexes

7.1 Docs du métaserveur

7.1.1 Qu'est-ce que le métaserveur ?

C'est un programme vous permettant de lancer votre (vos) client(s) ainsi que les différents serveurs (serveur effectif et serveur graphique). Vous pouvez même lancer les clients des autres ! Pour cela, il suffit de confectionner un petit fichier prolo.conf à la racine de votre compte (/).

7.1.2 Où trouver les docs ?

Sur papier, déjà. Nous vous fournissons les docs joliment formattées au format man Unix. Qui plus est, ces docs sont accessibles directement via l'outil man sur tous les PCs des salles machines. Les docs papier vous expliquent comment y accéder. Les docs portent sur les points suivants :

- pintro : Commande vous expliquant tout cela à nouveau, en plus détaillé.
- prun : Lance les clients et les serveurs.
- pprint : Affiche la liste des clients actuellement recensés dans le système.

– prolo.conf : Détail du format à respecter pour votre fichier de configuration.

C'est cool toute cette doc non ? ! En attendant, si pas de prolo.conf : Summon(POUFAPU).

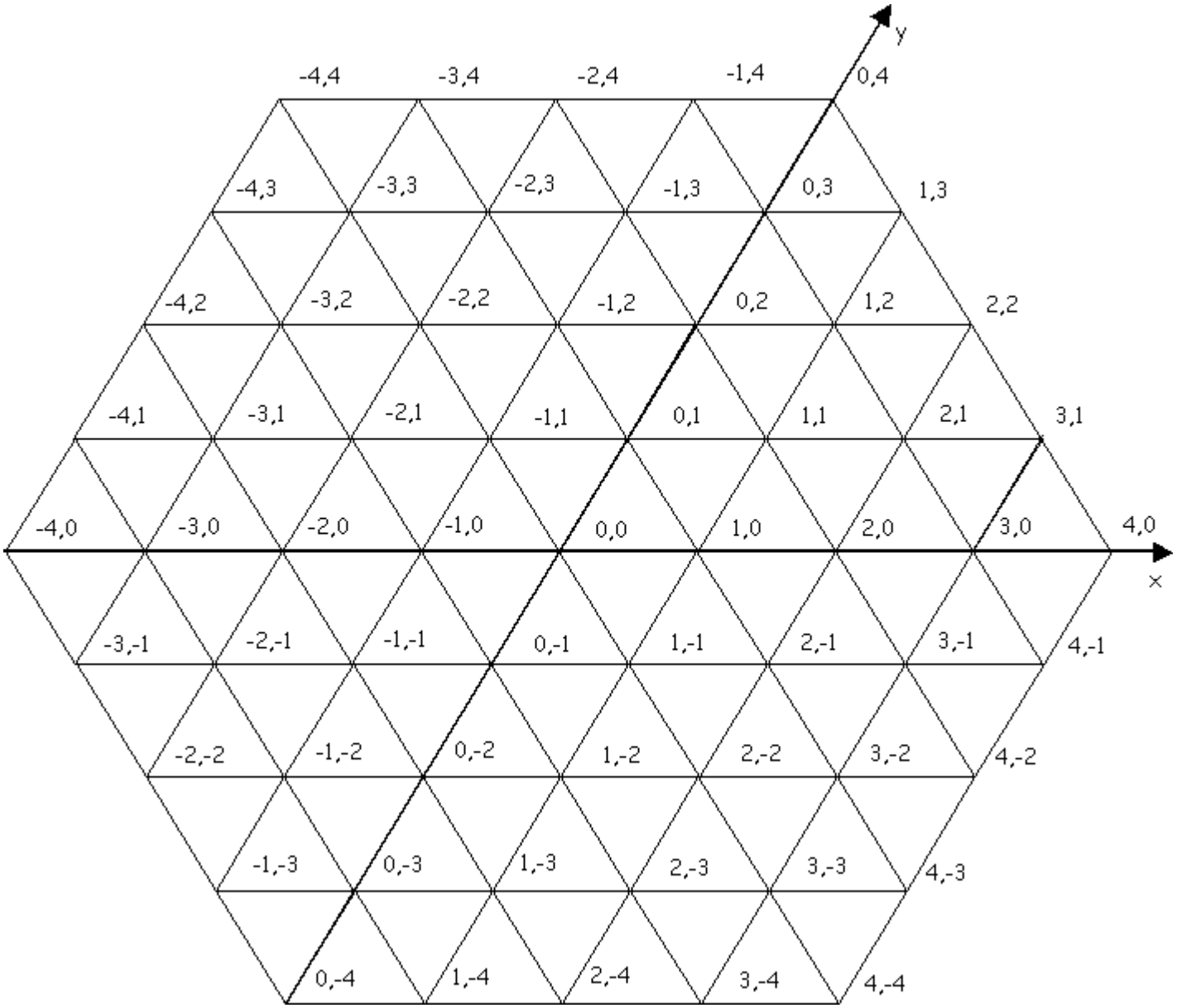
7.2 Déroulement de l'évaluation

- Nous sélectionnerons les candidats dépassant un certain score (qui est en pourcentage) en utilisant le VibraPaint tous seuls. Le nombre de candidats retenus est un multiple de 3.
- Les candidats ainsi retenus font ensuite l'objet de plusieurs passes d'un algorithme de tri de type Bubble Sort, dont le principe est très simple :
  - On part d'une liste d'ordre "aléatoire" de candidats.
  - On les fait effectuer les matchs par groupes de trois, pris linéairement dans la liste.
  - Pour chaque groupe, les champions sont replacés dans l'ordre de leurs résultats
  - Pour chaque groupe, on inverse le dernier (plus bas score) avec le premier du groupe suivant, ce qui fournit une nouvelle version de la liste globale.
  - On relance le processus depuis l'étape 2, jusqu'à obtenir un équilibre convenable.
- Dans cette phase, tous les matchs sont donc faits avec 3 joueurs. Cependant, les organisateurs se réservent le droit de tester votre champion dans toutes sortes d'autres conditions, afin de vérifier qu'il peut s'y adapter.
- Pouf! Et voilà, on a le classement de TOUS les candidats. Les 10 meilleurs sont sélectionnés pour soutenance Lundi matin. A l'issue de ces soutenances, le classement définitif des 10 meilleurs a lieu.
- En cas de quasi égalité entre le 10ème et les suivants, ils seront départagés par lecture de leur code en se basant sur les sources et votre doc. En effet, vous devez nous fournir une doc claire présentant les algorithmes et les principes de fonctionnement de votre champion. On ira regarder, pas la peine de pipeauter! Le document doit s'appeler kesako.txt et se trouver à la racine de votre compte. Pas de fichier? Pas classé(e)!

8 Liste des fichiers disponibles

Vous trouverez les fichiers suivants dans le répertoire prolo :

lib/libclient.a La librairie client générale (C/C++, Pascal)
lib/libclient.so La même, mais utilisable en Java
src/c/IClient.h Les prototypes des fonctions Client et Serveur (cf. section 6)
src/c/ClientFun.c Un squelette de code pour vos fonctions client (cf. section 6.2)
src/c/Client.c Un squelette de code pour votre programme principal
src/c/Makefile Le makefile de base, tout bô tout prêt à utiliser
src/cpp/IClient.h Les prototypes des fonctions Client et Serveur (cf. section 6)
src/cpp/ClientFun.cpp Un squelette de code pour vos fonctions client (cf. section 6.2)
src/cpp/Client.cpp Un squelette de code pour votre programme principal
src/cpp/Makefile Le makefile de base, tout bô tout prêt à utiliser
src/pas/prologin.pas Les prototypes des fonctions Serveur
src/pas/ClientFun.pas Les prototypes et le squelette des fonctions Client
src/pas/Client.pas Un squelette de code pour votre programme principal
src/pas/Makefile Le makefile version Pascal


y
x
-4,4
-3,4
-2,4
-1,4
0,4
-4,3
-3,3
-2,3
-1,3
0,3
1,3
-4,2
-3,2
-2,2
-1,2
0,2
1,2
2,2
-4,1
-3,1
-2,1
-1,1
0,1
1,1
2,1
3,1
-4,0
-3,0
-2,0
-1,0
0,0
1,0
2,0
3,0
4,0
-3,-1
-2,-1
-1,-1
0,-1
1,-1
2,-1
3,-1
4,-1
-2,-2
-1,-2
0,-2
1,-2
2,-2
3,-2
4,-2
-1,-3
0,-3
1,-3
2,-3
3,-3
4,-3
0,-4
1,-4
2,-4
3,-4
4,-4