CONCOURS COMMUNS POLYTECHNIQUES

ÉPREUVE SPÉCIFIQUE - FILIÈRE MP

INFORMATIQUE

DURÉE : 3 heures

L'usage de toute machine est interdit.

PRÉAMBULE : Les trois parties qui composent ce sujet sont indépendantes et peuvent être traitées par les candidats dans un ordre quelconque.

Partie I : Logique et calcul des propositions

Dans la mythologie Grecque, l'accès aux Enfers est gardé par le Cerbère, un terrible loup à trois têtes. Celui-ci se trouve devant trois couloirs qui, soit permettent de rejoindre le monde des vivants, soit conduisent directement aux Enfers.

Lorsque Cerbère accueille un nouvel arrivant, il est contraint de lui dire la vérité. Par la suite, il peut mentir ou dire la vérité à sa guise mais il respecte toujours les règles qu'il s'est fixées.

Après avoir bu la coupe de ciguë, Socrate se retrouve face à Cerbère. Celui-ci, honoré de rencontrer le grand philosophe, veut lui offrir une chance d'éviter la damnation éternelle. Il lui dit alors : « Je vais t'indiquer un des couloirs qui mène au monde des vivants mais, pour mettre à l'épreuve ta grande sagesse, j'énoncerai trois indications logiques qui seront, soit toutes vraies, soit toutes fausses et tu en déduiras le couloir que tu devras suivre ».

Nous noterons I_1, I_2 et I_3 les propositions associées aux indications de la première, la deuxième et la troisième tête du Cerbère.

Question I.1 *Représenter l'énoncé de Cerbère sous la forme d'une formule du calcul des propositions dépendant de I_1, I_2 et I_3.*

La première tête dit ensuite : « Le premier couloir ainsi que le troisième mènent au monde des vivants »,

La deuxième tête dit : « Si le deuxième couloir mène au monde des vivants, alors le troisième n'y mène pas »,

La troisième tête conclut par : « Le premier couloir mène au monde des vivants par contre le deuxième n'y mène pas ».

Nous noterons C_1, C_2, C_3 les variables propositionnelles correspondant au fait que le premier, le deuxième, le troisième couloir mènent au monde des vivants.

Question I.2 *Exprimer I_1, I_2 et I_3 sous la forme de formules du calcul des propositions dépendant de C_1, C_2 et C_3.*

Question I.3 *En utilisant le calcul des propositions (résolution avec les tables de vérité), déterminer le couloir que Socrate doit suivre pour rejoindre le monde des vivants.*

Question I.4 *En admettant que Cerbère ait menti en donnant les trois indications, Socrate pouvait-il suivre d'autres couloirs ? Si oui, le ou lesquels ?*

Partie II : Algorithmique et programmation en CaML

Cette partie doit être traitée par les étudiants qui ont utilisé le langage CaML dans le cadre des enseignements d'informatique. Les fonctions écrites devront être récursives ou faire appel à des fonctions auxiliaires récursives. Elles ne devront pas utiliser d'instructions itératives (`for`, `while`, ...) ni de références.

L'explication du comportement d'une fonction doit au moins expliciter :
- L'objectif général,
- Le rôle des paramètres de la fonction,
- Le résultat renvoyé par la fonction,
- Le rôle des variables locales à la fonction,
- Le principe de l'algorithme,
- La terminaison du calcul quels que soient les valeurs des paramètres.

Les approximations introduites lors de la conversion d'images analogiques continues en images numériques discontinues transforment souvent une partie continue de l'image en un nuage de points disjoints. Nous allons étudier dans cette partie une technique exploitée en traitement d'images numériques pour régénérer les parties continues : la construction de l'enveloppe convexe d'un ensemble de points.

1 Représentation du problème

1.1 Notations

Nous nous plaçons dans un espace euclidien orienté de dimension 2 muni d'un repère $(O,\vec{i},\vec{j})$.

- (p,q) représente la droite passant par les points p et q,
- $[p,q]$ représente le segment d'extrémités p et q et $]p,q[= [p,q] \setminus \{p,q\}$,
- $\widehat{pqr}$ est l'angle orienté entre les vecteurs $\overrightarrow{qp}$ et $\overrightarrow{qr}$, également noté $\widehat{\overrightarrow{qp},\overrightarrow{qr}}$.

1.2 Rappels

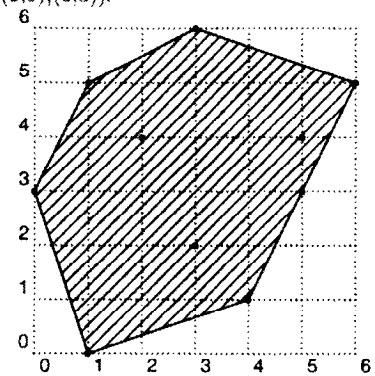
Déf. II.1 (Enveloppe convexe) *Soit $\mathcal{P}$ un ensemble de points; l'enveloppe convexe de $\mathcal{P}$, notée $\mathcal{E}_C(\mathcal{P})$, est le plus petit des ensembles convexes contenant $\mathcal{P}$. Dans le cas où $\mathcal{P}$ est un ensemble fini, ce que nous supposons dans la suite, nous admettrons que $\mathcal{E}_C(\mathcal{P})$ est un polygone convexe caractérisé par ses sommets : une suite $\mathcal{S} = (s_1,\ldots,s_n)$ composée de points de $\mathcal{P}$. Les points de $\mathcal{E}_C(\mathcal{P})$ sont alors des barycentres à coefficients positifs des points de $\mathcal{S}$, c'est-à-dire :*

$$p \in \mathcal{E}_C(\mathcal{P}) \Leftrightarrow \exists \{m_i\}_{i\in[1,n]}, m_i \geq 0, \sum_{i=1}^{n} m_i \neq 0, \sum_{i=1}^{n} m_i \overrightarrow{ps_i} = \overrightarrow{0}$$

Les points de la suite $\mathcal{S}$ sont ordonnés de manière à ce que :
- *les segments $[s_i,s_{i+1}]$ sont les arêtes du polygone convexe (avec $s_{n+1} = s_1$), ils forment une ligne polygonale : le contour du polygone,*

- *le contour du polygone est positif, c'est-à-dire orienté dans le sens direct.*

Exemple II.1 *Le schéma suivant représente en hachuré l'enveloppe convexe $\mathcal{E}_C(\mathcal{P})$ d'un ensemble de points $\mathcal{P} = \{(0,3),(1,0),(1,5),(2,4),(5,4),(3,2),(3,6),(4,1),(6,5),(5,3)\}$. Les sommets de $\mathcal{E}_C(\mathcal{P})$ sont $\mathcal{S} = ((1,0),(4,1),(6,5),(3,6),(1,5),(0,3))$.*



1.3 Codage en CaML

Un point est représenté par un couple d'entiers :

```
type point == int * int;;
```

Un ensemble, une liste, une suite de points sont représentés par une liste de points :

```
type suite = point list;;
```

Exemple II.2 *L'ensemble de points $\mathcal{P}$ de l'exemple II.1 est représenté par la liste :*

```
[ (0,3);(1,0);(1,5);(2,4);(5,4);(3,2);(3,6);(4,1);(6,5);(5,3) ]
```

L'enveloppe convexe de $\mathcal{P}$ est représentée par la liste :

```
[ (1,0);(4,1);(6,5);(3,6);(1,5);(0,3) ]
```

1.4 Produit croisé et fonctions trigonométriques

Les algorithmes que nous allons étudier reposent sur des propriétés trigonométriques. Le calcul informatique de fonctions trigonométriques est en général coûteux et source de nombreuses approximations. Pour éviter leur utilisation, nous allons exploiter le produit croisé qui permet la comparaison d'angles entre vecteurs sans calculer effectivement ces angles.

Déf. II.2 (Produit croisé) *Le produit croisé de trois points p, q et r est égal au produit mixte des vecteurs $\overrightarrow{qp}$ et $\overrightarrow{qr}$, c'est-à-dire : $[p,q,r] = Det(\overrightarrow{qp},\overrightarrow{qr})$.*

Question II.1 *Soient p, q et r trois points distincts; montrer que :*

1. $[p,q,r] > 0 \Leftrightarrow \widehat{pqr} \in]0,\pi[$,

2. $[p,q,r] < 0 \Leftrightarrow \widehat{pqr} \in]\pi,2\pi[$,
3. $[p,q,r] = 0 \Leftrightarrow p$, q *et* r *sont alignés,*

Question II.2 *Écrire en CaML une fonction* `croise` *de type* `point -> point -> point -> int` *telle que l'appel* `(croise p q r)` *renvoie la valeur de* [p,q,r].

1.5 Séparation du plan par une droite

Déf. II.3 *Soient p et q, deux points distincts; l'ensemble des points à gauche de la droite* (p,q) *est* $\{r \mid \widehat{pqr} \in]\pi,2\pi[\}$, *l'ensemble des points à droite de la droite* (p,q) *est* $\{r \mid \widehat{pqr} \in]0,\pi[\}$.

Question II.3 *Écrire en CaML une fonction* `a_gauche` *de type* `point -> point -> suite -> suite` *telle que l'appel* `(a_gauche p q e)` *renvoie une liste composée de tous les points de l'ensemble e qui :*

- *n'appartiennent pas à la droite* (p,q),
- *se trouvent à gauche de la droite* (p,q).

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

Question II.4 *Expliquer la fonction* `a_gauche` *proposée pour la question précédente.*

Question II.5 *Calculer une estimation de la complexité de la fonction* `a_gauche` *en fonction de la taille de l'ensemble e. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Question II.6 *Modifier la fonction précédente pour définir en CaML une fonction* `separation` *de type* `point -> point -> suite -> (suite * suite * suite)` *telle que l'appel* `(separation p q e)` *renvoie un triplet de listes* `( g, d, a)` *composées de tous les points de l'ensemble e qui :*

- *sont distincts de p et q,*
- *se trouvent à gauche de la droite* (p,q) *pour la liste g,*
- *se trouvent à droite de la droite* (p,q) *pour la liste d,*
- *appartiennent à la droite* (p,q) *pour la liste a.*

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

2 Relation d'ordre polaire

La technique de construction d'enveloppes convexes étudiée dans les parties suivantes repose sur l'utilisation d'une relation d'ordre polaire pour trier les sommets de $\mathcal{P}$.

Déf. II.4 (Relation d'ordre polaire) *Soit O un point; la relation* $\leq_O$ *sur un ensemble* $\mathcal{E}$ *est définie par :* $p \leq_O q \Leftrightarrow (\widehat{pOq} \in]0,\pi[) \vee ((\widehat{pOq} = 0) \wedge (\|\overrightarrow{Op}\| \geq \|\overrightarrow{Oq}\|))$.

Question II.7 *Écrire en CaML une fonction* `ordre_polaire` *de type* `point -> point -> point -> bool` *telle que l'appel* `(ordre_polaire o p q)` *renvoie la valeur vraie si et seulement si* $p \leq_o q$.

3 Décomposition en sous-problèmes disjoints

Pour simplifier la construction de relations d'ordre total à partir de relations d'ordre polaire, nous allons décomposer le problème en deux sous-problèmes disjoints séparés par la droite reliant le point le plus bas à gauche de $\mathcal{P}$ et le point le plus haut à droite de $\mathcal{P}$.

Déf. II.5 (Point en bas à gauche, en haut à droite) *Soient p et q deux points; p est en bas à gauche de q (et q est alors en haut à droite de p) si et seulement si* $(y_p < y_q) \vee ((y_p = y_q) \wedge (x_p < x_q))$. *Cette notion est étendue à des ensembles de points en prenant le minimum et le maximum de l'ensemble selon la relation d'ordre précédente.*

Question II.8 *Écrire en CaML une fonction* `extreme` *de type* `point -> point -> (point * point)` *telle que l'appel* `(extreme p1 p2)` *renvoie le couple* `(p1,p2)` *si p1 est en bas à gauche de p2 et le couple* `(p2,p1)` *si p2 est en bas à gauche de p1.*

Question II.9 *Écrire en CaML une fonction* `extremes` *de type* `suite -> (point * point)` *telle que l'appel* `(extremes e)`, *sur un ensemble non vide de points e, renvoie un couple de points* `(b,h)` *de l'ensemble e. b et h doivent être les points en bas à gauche et en haut à droite par rapport aux points contenus dans e. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.10 *Expliquer la fonction* `extremes` *proposée pour la question précédente.*

Question II.11 *Soit* $\mathcal{P}$ *un ensemble de points, soit* $B \in \mathcal{P}$ *le point en bas à gauche de* $\mathcal{P}$, *soit* $H \in \mathcal{P}$ *le point en haut à droite de* $\mathcal{P}$; *montrer que B et H sont des sommets de l'enveloppe convexe* $\mathcal{E}_C(\mathcal{P})$.

Théorème II.1 *Soit* $\mathcal{P}$ *un ensemble de points; soient :*

- $\mathcal{P}_\mathcal{D}$ *les points à droite de la droite* (B,H) *n'appartenant pas à* (B,H),
- $\mathcal{P}_\mathcal{G}$ *les points à gauche de la droite* (B,H) *n'appartenant pas à* (B,H),
- $\mathcal{P}_{BH}$ *les points de* $]B,H[$,

alors $\{B\} \cup \mathcal{P}_\mathcal{D} \cup \{H\} \cup \mathcal{P}_\mathcal{G} \cup \mathcal{P}_{BH}$ *forme une partition disjointe de* $\mathcal{P}$ *et* $\mathcal{E}_C(\mathcal{P}) = \mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D}) \cup \mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{G})$.

Exemple II.3 *L'ensemble de points* $\mathcal{P}$ *de l'exemple II.1 est partitionné en :*

- $B = (1,0)$
- $H = (3,6)$
- $\mathcal{P}_\mathcal{D} = \{(5,4),(3,2),(4,1),(5,3),(6,5)\}$
- $\mathcal{P}_\mathcal{G} = \{(0,3),(1,5),(2,4)\}$
- $\mathcal{P}_{BH} = \emptyset$

Question II.12 *Montrer que la relation d'ordre polaire* $\leq_B$ *est une relation d'ordre total sur* $\mathcal{P}_\mathcal{D} \cup \mathcal{P}_{BH} \cup \{H\}$.

4 Algorithme de Jarvis : Génération des sommets

L'algorithme étudié dans cette partie est dû à Jarvis. Il est dérivé de la construction de l'ensemble des arêtes composant le contour de l'enveloppe convexe. Il repose sur le fait que les sommets d'un polygone convexe se trouvent soit tous à droite, soit tous à gauche de chaque arête du polygone.

Théorème II.2 *Un polygone $\mathcal{S}$ de sommets $(s_1,\ldots,s_n)$ est convexe si et seulement si, pour toute arête $[s_i,s_{i+1}]$, les autres sommets de $\mathcal{S}$ sont, soit tous à gauche (contour positif), soit tous à droite (contour négatif) par rapport à la droite (s_i,s_{i+1}).*

L'algorithme initial proposé par Jarvis consiste à engendrer tous les segments possibles connectant deux points de l'ensemble, puis à éliminer les segments qui ne sont pas des arêtes. Cet algorithme est trop coûteux pour être utilisé pratiquement. Nous allons étudier une variante permettant d'éviter la construction de tous les segments potentiels.

Cette variante est basée sur la même propriété des arêtes. Elle consiste à choisir un premier sommet, puis à construire l'enveloppe en parcourant le contour positif.

L'ensemble $\mathcal{P}$ est partitionné selon le théorème II.1.

Nous allons étudier le traitement de la partie droite $\mathcal{P}_\mathcal{D}$ de $\mathcal{P}$ en partant du point en bas à gauche B. Le traitement de la partie gauche $\mathcal{P}_\mathcal{G}$ sera symétrique en partant du point en haut à droite H.

Notons $(s_1,\ldots,s_n)$ les sommets de $\mathcal{E}_\mathcal{C}(\{B,H\}\cup\mathcal{P}_\mathcal{D})$. Nous avons montré dans la question II.11 que $s_1 = B$ et $s_n = H$. Nous allons construire $s_2,\ldots,s_{n-1}$ selon la relation suivante : s_{i+1} est le minimum de $\mathcal{P}_\mathcal{D}\setminus\{s_2,\ldots,s_i\}$ selon $\leq_{s_i}$. Montrons dans une première étape la correction de cette technique :

Question II.13 *Soit p le minimum de $\mathcal{P}_\mathcal{D}\cup\{H\}$ selon $\leq_B$; montrer que p est un sommet de $\mathcal{E}_\mathcal{C}(\{B,H\}\cup\mathcal{P}_\mathcal{D})$.*

Question II.14 *Soit $1 < i < n$, soient $(s_1,\ldots,s_i)$ les i premiers sommets de $\mathcal{E}_\mathcal{C}(\{B,H\}\cup\mathcal{P}_\mathcal{D})$, soit p le minimum de $(\mathcal{P}_\mathcal{D}\cup\{H\})\setminus\{s_1,\ldots,s_i\}$ selon $\leq_{s_i}$; montrer que p est un sommet de $\mathcal{E}_\mathcal{C}(\{B,H\}\cup\mathcal{P}_\mathcal{D})$.*

Question II.15 *Déduire des questions précédentes que $(s_1,\ldots,s_n)$ est la suite des sommets de $\mathcal{E}_\mathcal{C}(\{B,H\}\cup\mathcal{P}_\mathcal{D})$.*

Question II.16 *Écrire en CaML une fonction `min_polaire` de type `point -> suite -> (point * suite)` telle que l'appel `(min_polaire o e)`, sur un ensemble non vide de points `e`, renvoie un couple `(p, r)` composé du point `p`, minimum dans l'ensemble `e` selon l'ordre $\leq_o$ et du reste `r` de l'ensemble `e` privé de `p`. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.17 *Expliquer la fonction `min_polaire` proposée pour la question précédente.*

Question II.18 *Calculer une estimation de la complexité de la fonction `min_polaire` en fonction de la taille de la liste `e`. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Question II.19 *Écrire en CaML une fonction `demi_jarvis` de type `point -> point -> suite -> suite` telle que l'appel `(demi_jarvis b h e)` avec `e` un ensemble de points situés à droite de la droite (b,h), renvoie la suite $(s_1,\ldots,s_n)$ composée de points pris dans $\mathtt{e}\cup\{\mathtt{b},\mathtt{h}\}$ avec :*

- $s_1 = \mathtt{b}$,
- $s_n = \mathtt{h}$,
- s_{i+1} *est le minimum de* $(\mathtt{e}\cup\{\mathtt{h}\})\setminus\{s_1,\ldots,s_i\}$ *selon* $\leq_{s_i}$ *(avec $i < n$).*

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

Question II.20 *Expliquer la fonction `demi_jarvis` proposée pour la question précédente.*

Question II.21 *Calculer une estimation de la complexité de la fonction `demi_jarvis` en fonction de la taille de l'ensemble `e` et du nombre de sommets de l'enveloppe convexe n. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Question II.22 *Écrire en CaML une fonction `jarvis` de type `suite -> suite` telle que l'appel `(jarvis e)`, sur un ensemble de points `e` contenant au moins trois points, renvoie la suite des points composant l'enveloppe convexe des points contenus dans l'ensemble `e`. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.23 *Expliquer la fonction `jarvis` proposée pour la question précédente.*

Question II.24 *Calculer une estimation de la complexité de la fonction `jarvis` en fonction de la taille de l'ensemble `e`. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Partie II : Algorithmique et programmation en PASCAL

Cette partie doit être traitée par les étudiants qui ont utilisé le langage PASCAL dans le cadre des enseignements d'informatique. Les fonctions écrites devront être récursives ou faire appel à des fonctions auxiliaires récursives. Elles ne devront pas utiliser d'instructions itératives (`for`, `while`, `repeat`, ...).

L'explication du comportement d'une fonction doit au moins expliciter :

- L'objectif général,
- Le rôle des paramètres de la fonction,
- Le résultat renvoyé par la fonction,
- Le rôle des variables locales à la fonction,
- Le principe de l'algorithme,
- La terminaison du calcul quels que soient les valeurs des paramètres.

Les approximations introduites lors de la conversion d'images analogiques continues en images numériques discontinues transforment souvent une partie continue de l'image en un nuage de points disjoints. Nous allons étudier dans cette partie une technique exploitée en traitement d'images numériques pour régénérer les parties continues : la construction de l'enveloppe convexe d'un ensemble de points.

1 Représentation du problème

1.1 Notations

Nous nous plaçons dans un espace euclidien orienté de dimension 2 muni d'un repère $(O,\vec{i},\vec{j})$.

- (p,q) représente la droite passant par les points p et q,
- $[p,q]$ représente le segment d'extrémités p et q et $]p,q[= [p,q] \setminus \{p,q\}$,
- $\widehat{pqr}$ est l'angle orienté entre les vecteurs $\overrightarrow{qp}$ et $\overrightarrow{qr}$, également noté $\widehat{\overrightarrow{qp},\overrightarrow{qr}}$.

1.2 Rappels

Déf. II.1 (Enveloppe convexe) *Soit $\mathcal{P}$ un ensemble de points; l'enveloppe convexe de $\mathcal{P}$, notée $\mathcal{E}_C(\mathcal{P})$, est le plus petit des ensembles convexes contenant $\mathcal{P}$. Dans le cas où $\mathcal{P}$ est un ensemble fini, ce que nous supposons dans la suite, nous admettrons que $\mathcal{E}_C(\mathcal{P})$ est un polygone convexe caractérisé par ses sommets : une suite $\mathcal{S} = (s_1,\ldots,s_n)$ composée de points de $\mathcal{P}$. Les points de $\mathcal{E}_C(\mathcal{P})$ sont alors des barycentres à coefficients positifs des points de $\mathcal{S}$, c'est-à-dire :*

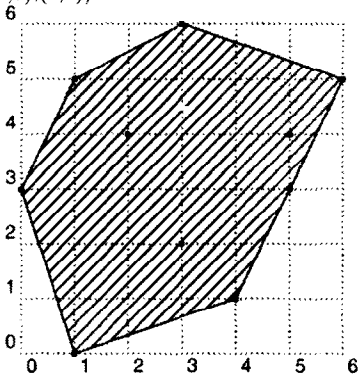
$$p \in \mathcal{E}_C(\mathcal{P}) \Leftrightarrow \exists \{m_i\}_{i\in[1,n]}, m_i \geq 0, \sum_{i=1}^{n} m_i \neq 0, \sum_{i=1}^{n} m_i \overrightarrow{ps_i} = \vec{0}$$

Les points de la suite $\mathcal{S}$ sont ordonnés de manière à ce que :

- *les segments $[s_i,s_{i+1}]$ sont les arêtes du polygone convexe (avec $s_{n+1} = s_1$), ils forment une ligne polygonale : le contour du polygone,*

- *le contour du polygone est positif, c'est-à-dire orienté dans le sens direct.*

Exemple II.1 *Le schéma suivant représente en hachuré l'enveloppe convexe $\mathcal{E}_C(\mathcal{P})$ d'un ensemble de points $\mathcal{P} = \{(0,3),(1,0),(1,5),(2,4),(5,4),(3,2),(3,6),(4,1),(6,5),(5,3)\}$. Les sommets de $\mathcal{E}_C(\mathcal{P})$ sont $\mathcal{S} = ((1,0),(4,1),(6,5),(3,6),(1,5),(0,3))$.*



1.3 Codage en PASCAL

Un point est représenté par le type de base `POINT`.

Un ensemble, une liste, une suite de points sont représentés par le type de base `SUITE`.

Nous supposons prédéfinies les constantes et les fonctions suivantes dont le calcul se termine quelles que soient les valeurs de leurs paramètres. Elles pourront éventuellement être utilisées dans les réponses aux questions :

- `FUNCTION P_Creer(a,o:INTEGER):POINT;` renvoie un point dont l'abscisse est `a` et l'ordonnée `o`,
- `FUNCTION P_Abs(p:POINT):INTEGER;` renvoie l'abscisse de `p`,
- `FUNCTION P_Ord(p:POINT):INTEGER;` renvoie l'ordonnée de `p`,
- `NIL` représente la suite vide de points,
- `FUNCTION S_Ajouter(p:POINT;s:SUITE):SUITE;` renvoie une suite de points composée d'un premier point `p` et du reste de la suite contenu dans `s`,
- `FUNCTION S_Premier(s:SUITE):POINT;` renvoie le premier point de la suite `s`,
- `FUNCTION S_Reste(s:SUITE):SUITE;` renvoie le reste de la suite `s` privée de son premier point,
- `FUNCTION S_Juxtaposer(s1,s2:SUITE):SUITE;` renvoie la suite composée de la suite `s1` suivie de la suite `s2`.

Exemple II.2 *L'ensemble de points $\mathcal{P}$ de l'exemple II.1 est représenté par la liste :*

```
S_Ajouter( P_Creer(0,3), S_Ajouter( P_Creer(1,0),
S_Ajouter( P_Creer(1,5), S_Ajouter( P_Creer(2,4),
S_Ajouter( P_Creer(5,4), S_Ajouter( P_Creer(3,2),
S_Ajouter( P_Creer(3,6), S_Ajouter( P_Creer(4,1),
S_Ajouter( P_Creer(6,5), S_Ajouter( P_Creer(5,3), NIL)))))))))) ]
```

L'enveloppe convexe de $\mathcal{P}$ est représentée par la liste :

```
S_Ajouter( P_Creer(1,0), S_Ajouter( P_Creer(4,1),
S_Ajouter( P_Creer(6,5), S_Ajouter( P_Creer(3,6),
S_Ajouter( P_Creer(1,5), S_Ajouter( P_Creer(0,3), NIL))))))
```

1.4 Produit croisé et fonctions trigonométriques

Les algorithmes que nous allons étudier reposent sur des propriétés trigonométriques. Le calcul informatique de fonctions trigonométriques est en général coûteux et source de nombreuses approximations. Pour éviter leur utilisation, nous allons exploiter le produit croisé qui permet la comparaison d'angles entre vecteurs sans calculer effectivement ces angles.

Déf. II.2 (Produit croisé) *Le produit croisé de trois points p, q et r est égal au produit mixte des vecteurs $\overrightarrow{qp}$ et $\overrightarrow{qr}$, c'est-à-dire : $[p,q,r] = Det(\overrightarrow{qp},\overrightarrow{qr})$.*

Question II.1 *Soient p, q et r trois points distincts; montrer que :*

1. $[p,q,r] > 0 \Leftrightarrow \widehat{pqr} \in]0,\pi[$,
2. $[p,q,r] < 0 \Leftrightarrow \widehat{pqr} \in]\pi,2\pi[$,
3. $[p,q,r] = 0 \Leftrightarrow p$, q *et* r *sont alignés,*

Question II.2 *Écrire en PASCAL une fonction* `croise(p,q,r:POINT):INTEGER;` *telle que l'appel* `croise(p,q,r)` *renvoie la valeur de* [p,q,r].

1.5 Séparation du plan par une droite

Déf. II.3 *Soient p et q, deux points distincts; l'ensemble des points à gauche de la droite (p,q) est $\{r \mid \widehat{pqr} \in]\pi,2\pi[\}$, l'ensemble des points à droite de la droite (p,q) est $\{r \mid \widehat{pqr} \in]0,\pi[\}$.*

Question II.3 *Écrire en PASCAL une fonction* `a_gauche(p,q:POINT;e:SUITE):SUITE` *telle que l'appel* `a_gauche(p,q,e)` *renvoie une liste composée de tous les points de l'ensemble* `e` *qui :*

- *n'appartiennent pas à la droite* (p,q),
- *se trouvent à gauche de la droite* (p,q).

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

Question II.4 *Expliquer la fonction* `a_gauche` *proposée pour la question précédente.*

Question II.5 *Calculer une estimation de la complexité de la fonction* `a_gauche` *en fonction de la taille de l'ensemble* `e`. *Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Un triplet de suites de points est représenté par le type de base `TRIPLET`.

Nous supposons prédéfinies les constantes et les fonctions suivantes dont le calcul se termine quelles que soient les valeurs de leurs paramètres. Elles pourront éventuellement être utilisées dans les réponses aux questions :

- `FUNCTION T_Creer(s1,s2,s3:SUITE):TRIPLET;` renvoie un triplet dont les éléments sont `s1`, `s2` et `s3`,
- `FUNCTION T_Un(t:TRIPLET):SUITE;` renvoie le premier élément de `t`,
- `FUNCTION T_Deux(t:TRIPLET):SUITE;` renvoie le deuxième élément de `t`,
- `FUNCTION T_Trois(t:TRIPLET):SUITE;` renvoie le troisième élément de `t`.

Question II.6 *Modifier la fonction précédente pour définir en PASCAL une fonction* `separation(p,q:POINT;e:SUITE):TRIPLET;` *telle que l'appel* `separation(p,q,e)` *renvoie un triplet* `T_Creer(g,d,a)` *de listes composées de tous les points de l'ensemble* `e` *qui :*

- *sont distincts de* `p` *et* `q`,
- *se trouvent à gauche de la droite* (p,q) *pour la liste* `g`,
- *se trouvent à droite de la droite* (p,q) *pour la liste* `d`,
- *appartiennent à la droite* (p,q) *pour la liste* `a`.

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

2 Relation d'ordre polaire

La technique de construction d'enveloppes convexes étudiée dans les parties suivantes repose sur l'utilisation d'une relation d'ordre polaire pour trier les sommets de $\mathcal{P}$.

Déf. II.4 (Relation d'ordre polaire) *Soit O un point; la relation $\leq_O$ sur un ensemble $\mathcal{E}$ est définie par : $p \leq_O q \Leftrightarrow (\widehat{pOq} \in]0,\pi[) \vee ((\widehat{pOq} = 0) \wedge (\|\overrightarrow{Op}\| \geq \|\overrightarrow{Oq}\|))$.*

Question II.7 *Écrire en PASCAL une fonction* `ordre_polaire(o,p,q:POINT):BOOLEAN;` *telle que l'appel* `ordre_polaire(o,p,q)` *renvoie la valeur vraie si $p \leq_o q$.*

3 Décomposition en sous-problèmes disjoints

Pour simplifier la construction de relations d'ordre total à partir de relations d'ordre polaire, nous allons décomposer le problème en deux sous-problèmes disjoints séparés par la droite reliant le point le plus bas à gauche de $\mathcal{P}$ et le point le plus haut à droite de $\mathcal{P}$.

Déf. II.5 (Point en bas à gauche, en haut à droite) *Soient p et q deux points; p est en bas à gauche de q (et q est alors en haut à droite de p) si et seulement si $(y_p < y_q) \vee ((y_p = y_q) \wedge (x_p < x_q))$. Cette notion est étendue à des ensembles de points en prenant le minimum et le maximum de l'ensemble selon la relation d'ordre précédente.*

Un couple de points est représenté par le type de base `COUPLE`.

Nous supposons prédéfinies les constantes et les fonctions suivantes dont le calcul se termine quelles que soient les valeurs de leurs paramètres. Elles pourront éventuellement être utilisées dans les réponses aux questions :

- `FUNCTION C_Creer(p1,p2:POINT):COUPLE;` renvoie un couple dont les éléments sont `p1` et `p2`,
- `FUNCTION C_Un(c:COUPLE):POINT;` renvoie le premier élément de `c`,

- `FUNCTION C_Deux(c:COUPLE):POINT;` renvoie le second élément de `c`.

Question II.8 *Écrire en PASCAL une fonction `extreme(p1,p2:POINT):COUPLE;` telle que l'appel `extreme(p1,p2)` renvoie le couple `C_Creer(p1,p2)` si `p1` est en bas à gauche de `p2` et le couple `C_Creer(p2,p1)` sinon (dans ce cas, `p2` est en bas à gauche de `p1`).*

Question II.9 *Écrire en PASCAL une fonction `extremes(e:SUITE):COUPLE;` telle que l'appel `extremes(e)`, sur un ensemble non vide de points `e`, renvoie un couple de points `C_Creer(b,h)` de l'ensemble `e`. `b` et `h` sont les point en bas à gauche et en haut à droite par rapport aux points contenus dans `e`. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.10 *Expliquer la fonction `extremes` proposée pour la question précédente.*

Question II.11 *Soit $\mathcal{P}$ un ensemble de points, soit $B \in \mathcal{P}$ le point en bas à gauche de $\mathcal{P}$, soit $H \in \mathcal{P}$ le point en haut à droite de $\mathcal{P}$; montrer que B et H sont des sommets de l'enveloppe convexe $\mathcal{E}_C(\mathcal{P})$.*

Théorème II.1 *Soit $\mathcal{P}$ un ensemble de points; soient :*

- *$\mathcal{P}_\mathcal{D}$ les points à droite de la droite (B,H) n'appartenant pas à (B,H),*
- *$\mathcal{P}_\mathcal{G}$ les points à gauche de la droite (B,H) n'appartenant pas à (B,H),*
- *$\mathcal{P}_{\mathcal{BH}}$ les points de $]B,H[$,*

alors $\{B\} \cup \mathcal{P}_\mathcal{D} \cup \{H\} \cup \mathcal{P}_\mathcal{G} \cup \mathcal{P}_{\mathcal{BH}}$ forme une partition disjointe de $\mathcal{P}$ et $\mathcal{E}_C(\mathcal{P}) = \mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D}) \cup \mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{G})$.

Exemple II.3 *L'ensemble de points $\mathcal{P}$ de l'exemple II.1 est partitionné en :*

- $B = (1,0)$
- $H = (3,6)$
- $\mathcal{P}_\mathcal{D} = \{(5,4),(3,2),(4,1),(5,3),(6,5)\}$
- $\mathcal{P}_\mathcal{G} = \{(0,3),(1,5),(2,4)\}$
- $\mathcal{P}_{\mathcal{BH}} = \emptyset$

Question II.12 *Montrer que la relation d'ordre polaire $\leq_B$ est une relation d'ordre total sur $\mathcal{P}_\mathcal{D} \cup \mathcal{P}_{\mathcal{BH}} \cup \{H\}$.*

4 Algorithme de Jarvis : Génération des sommets

L'algorithme étudié dans cette partie est dû à Jarvis. Il est dérivé de la construction de l'ensemble des arêtes composant le contour de l'enveloppe convexe. Il repose sur le fait que les sommets d'un polygone convexe se trouvent soit tous à droite, soit tous à gauche de chaque arête du polygone.

Théorème II.2 *Un polygone S de sommets $(s_1,\ldots,s_n)$ est convexe si et seulement si, pour toute arête $[s_i,s_{i+1}]$, les autres sommets de S sont, soit tous à gauche (contour positif), soit tous à droite (contour négatif) par rapport à la droite (s_i,s_{i+1}).*

L'algorithme initial proposé par Jarvis consiste à engendrer tous les segments possibles connectant deux points de l'ensemble, puis à éliminer les segments qui ne sont pas des arêtes. Cet algorithme est trop coûteux pour être utilisé pratiquement. Nous allons étudier une variante permettant d'éviter la construction de tous les segments potentiels.

Cette variante est basée sur la même propriété des arêtes. Elle consiste à choisir un premier sommet, puis à construire l'enveloppe en parcourant le contour positif.

L'ensemble $\mathcal{P}$ est partitionné selon le théorème II.1.

Nous allons étudier le traitement de la partie droite $\mathcal{P}_\mathcal{D}$ de $\mathcal{P}$ en partant du point en bas à gauche B. Le traitement de la partie gauche $\mathcal{P}_\mathcal{G}$ sera symétrique en partant du point en haut à droite H.

Notons $(s_1,\ldots,s_n)$ les sommets de $\mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D})$. Nous avons montré dans la question II.11 que $s_1 = B$ et $s_n = H$. Nous allons construire $s_2,\ldots,s_{n-1}$ selon la relation suivante : s_{i+1} est le minimum de $\mathcal{P}_\mathcal{D} \setminus \{s_2,\ldots,s_i\}$ selon $\leq_{s_i}$. Montrons dans une première étape la correction de cette technique :

Question II.13 *Soit p le minimum de $\mathcal{P}_\mathcal{D} \cup \{H\}$ selon $\leq_B$; montrer que p est un sommet de $\mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D})$.*

Question II.14 *Soit $1 < i < n$, soient $(s_1,\ldots,s_i)$ les i premiers sommets de $\mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D})$, soit p le minimum de $(\mathcal{P}_\mathcal{D} \cup \{H\}) \setminus \{s_1,\ldots,s_i\}$ selon $\leq_{s_i}$; montrer que p est un sommet de $\mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D})$.*

Question II.15 *Déduire des questions précédentes que $(s_1,\ldots,s_n)$ est la suite des sommets de $\mathcal{E}_C(\{B,H\} \cup \mathcal{P}_\mathcal{D})$.*

Question II.16 *Écrire en PASCAL une fonction `min_polaire(o:POINT;e:SUITE):SUITE` telle que l'appel `min_polaire(o,e)`, sur un ensemble non vide de points, renvoie une suite `S_Ajouter(p,r)` composée d'un premier point `p`, minimum dans l'ensemble `e` selon l'ordre $\leq_o$ et du reste `r` composé des points de `e` privé de `p`. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.17 *Expliquer la fonction `min_polaire` proposée pour la question précédente.*

Question II.18 *Calculer une estimation de la complexité de la fonction `min_polaire` en fonction de la taille de la liste `e`. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Question II.19 *Écrire en PASCAL une fonction `demi_jarvis(b,h:POINT;e:SUITE):SUITE;` telle que l'appel `demi_jarvis(b,h,e)` avec `e` un ensemble de points situés à droite de la droite (b,h), renvoie la suite $(s_1,\ldots s_n)$ composée de points pris dans $\mathtt{e} \cup \{\mathtt{b},\mathtt{h}\}$ avec :*

- *$s_1 = \mathtt{b}$,*
- *$s_n = \mathtt{h}$,*
- *s_{i+1} est le minimum de $(\mathtt{e} \cup \{\mathtt{h}\}) \setminus \{s_1,\ldots,s_i\}$ selon $\leq_{s_i}$ (avec $i < n$).*

Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.

Question II.20 *Expliquer la fonction `demi_jarvis` proposée pour la question précédente.*

Question II.21 *Calculer une estimation de la complexité de la fonction `demi_jarvis` en fonction de la taille de l'ensemble `e` et du nombre de sommets de l'enveloppe convexe n. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Question II.22 *Écrire en PASCAL une fonction* `jarvis(e:SUITE):SUITE;` *telle que l'appel* `jarvis(e)`*, sur un ensemble* `e` *contenant au moins trois points, renvoie la suite des points composant l'enveloppe convexe des points contenus dans l'ensemble* `e`*. Cette fonction devra être récursive ou faire appel à des fonctions auxiliaires récursives.*

Question II.23 *Expliquer la fonction* `jarvis` *proposée pour la question précédente.*

Question II.24 *Calculer une estimation de la complexité de la fonction* `jarvis` *en fonction de la taille de l'ensemble* `e`*. Cette estimation ne prendra en compte que le nombre d'appels récursifs effectués.*

Partie III : Automates et langages

Le but de cet exercice est l'étude des propriétés de l'opération d'inversion d'un automate.

Pour simplifier l'exercice, nous considérerons uniquement des automates qui ne possèdent pas de transitions arbitraires ϵ. Les résultats étudiés sont également valides dans ces cas-là, mais la formalisation et la construction des preuves sont plus complexes.

Soit l'alphabet X, un ensemble de symboles, soit Λ (parfois noté abusivement avec le même symbole que la transition arbitraire ϵ) le symbole représentant le mot vide ($\Lambda \notin X$), soit $X^\star$ l'ensemble des mots composés de symboles de X ($\Lambda \in X^\star$); un automate fini sur X est un quintuplet $A = (Q,X,I,T,\gamma)$ composé de :

- Un ensemble d'états : Q,
- Un ensemble d'états initiaux : $I \subseteq Q$,
- Un ensemble d'états terminaux : $T \subseteq Q$,
- Une relation de transition : $\gamma \subseteq X \times Q \times Q$.

Remarquons que γ est souvent notée sous la forme d'une fonction totale de transition $\delta \subseteq X \times Q \to \mathcal{P}(Q)$ dont les valeurs sont définies par :

$$\forall x \in X, \forall q \in Q, \delta(x,q) = \{q' \in Q \mid (x,q,q') \in \gamma\}$$

La notation γ, contrairement à δ, est symétrique. Cette propriété simplifie la formalisation et la construction des preuves.

Les valeurs de la relation de transition γ seront représentées par un graphe dont les nœuds sont les états. Un état initial sera entouré d'un cercle (i) et un état final sera entouré d'un double cercle $((t))$. Une arête étiquetée par le symbole $x \in X$ ira de l'état q à l'état q' si et seulement si $(x,q,q') \in \gamma$.

Soit $\gamma^\star$ l'extension de γ à $X^\star \times Q \times Q$ définie par :

$$\begin{cases} \forall q \in Q, (\Lambda,q,q) \in \gamma^\star \\ \forall x \in X, \forall q \in Q, \forall q' \in Q, (x,q,q') \in \gamma^\star \Leftrightarrow (x,q,q') \in \gamma \\ \forall x \in X, \forall m \in X^\star, \forall q \in Q, \forall q' \in Q, (x.m,q,q') \in \gamma^\star \Leftrightarrow \exists q'' \in Q, (x,q,q'') \in \gamma \wedge (m,q'',q') \in \gamma^\star \end{cases}$$

Le langage sur $X^\star$ reconnu par un automate fini est :

$$L(A) = \{m \in X^\star \mid q_I \in I, q_T \in T, (m,q_I,q_T) \in \gamma^\star\}$$

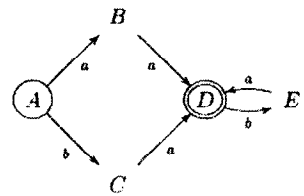
Déf. III.1 (État accessible) *Soit l'automate fini $A = (Q,X,I,T,\gamma)$, $q \in Q$ est accessible si et seulement si $\exists q_I \in I, \exists m \in X, (m,q_I,q) \in \gamma^\star$.*

Déf. III.2 (État co-accessible) *Soit l'automate fini $A = (Q,X,I,T,\gamma)$, $q \in Q$ est co-accessible si et seulement si $\exists q_T \in T, \exists m \in X, (m,q,q_T) \in \gamma^\star$.*

Nous considérerons uniquement dans cet exercice des automates dont tous les états sont accessibles et co-accessibles.

1 Étude de l'exemple $\mathcal{E}$

Soit l'automate fini $\mathcal{E} = (\{A,B,C,D,E\},\{a,b\},\{A\},\{D\},\gamma_\mathcal{E})$ dont la relation de transition est représentée par le graphe :



Question III.1 *Donner sans la justifier une expression régulière ou ensembliste représentant le langage reconnu par $\mathcal{E}$.*

2 Opération d'inversion d'un automate : $\mathcal{A}^{-1}$

Soit l'opération interne d'inversion d'un automate fini définie par :

Déf. III.3 *Soit $\mathcal{A} = (Q,X,I,T,\gamma)$ un automate fini, l'automate $\mathcal{B} = \mathcal{A}^{-1}$, inverse de l'automate $\mathcal{A}$, est défini par :*

$$\mathcal{B} = (Q,X,T,I,\gamma^{-1})$$
$$\forall x \in X, \forall q \in Q, \forall q' \in Q, (x,q,q') \in \gamma^{-1} \Leftrightarrow (x,q',q) \in \gamma$$

Question III.2 *Construire l'automate $\mathcal{E}^{-1}$.*

Question III.3 *Justifier que l'automate $\mathcal{E}^{-1}$ est non déterministe puis le déterminiser (seuls les états et les transitions utiles, c'est-à-dire accessibles depuis les états initiaux et co-accessibles depuis les états terminaux, devront être construits). L'automate obtenu est noté $\widehat{\mathcal{E}^{-1}}$.*

Question III.4 *Caractériser le langage reconnu par $\mathcal{E}^{-1}$ par une expression régulière ou ensembliste. Comparer ce langage avec le langage reconnu par $\mathcal{E}$.*

Question III.5 *Construire l'automate $\widehat{\mathcal{E}^{-1}}^{-1}$.*

Question III.6 *Justifier que l'automate $\widehat{\mathcal{E}^{-1}}^{-1}$ est non déterministe puis le déterminiser (seuls les états et les transitions utiles, c'est-à-dire accessibles depuis les états initiaux et co-accessibles depuis les états terminaux, devront être construits). L'automate obtenu est noté $\widehat{\widehat{\mathcal{E}^{-1}}^{-1}}$.*

Question III.7 *Caractériser le langage reconnu par $\widehat{\widehat{\mathcal{E}^{-1}}^{-1}}$ par une expression régulière ou ensembliste. Comparer ce langage avec le langage reconnu par $\mathcal{E}$.*

3 Étude de l'automate inverse

Question III.8 *Montrer que :*

$$\forall m \in X^*, \forall x \in X, \forall q \in Q, \forall q' \in Q, (m.x,q,q') \in \gamma^* \Leftrightarrow \exists q'' \in Q, (m,q,q'') \in \gamma^* \wedge (x,q'',q') \in \gamma$$

Déf. III.4 *L'opération d'inversion de mots sur X^* est définie par :*

$$\begin{cases} \Lambda^{-1} = \Lambda \\ \forall x \in X, \forall m \in X^*, (x.m)^{-1} = m^{-1}.x \end{cases}$$

Question III.9 *Montrer que :* $\forall m \in X^*, \forall q \in Q, \forall q' \in Q, (m,q,q') \in \gamma^* \Leftrightarrow (m^{-1},q',q) \in \gamma^{-1^*}$.

Question III.10 *Quelle relation liant les langages reconnus par $\mathcal{A}$ et $\mathcal{A}^{-1}$ peut-on en déduire ?*

Fin de l'énoncé