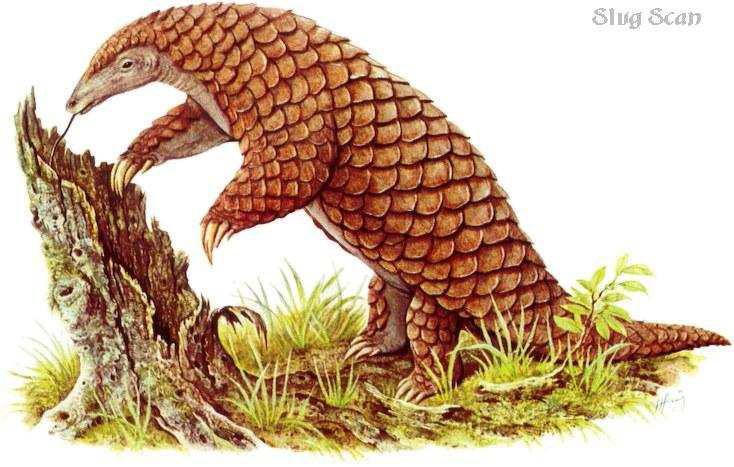
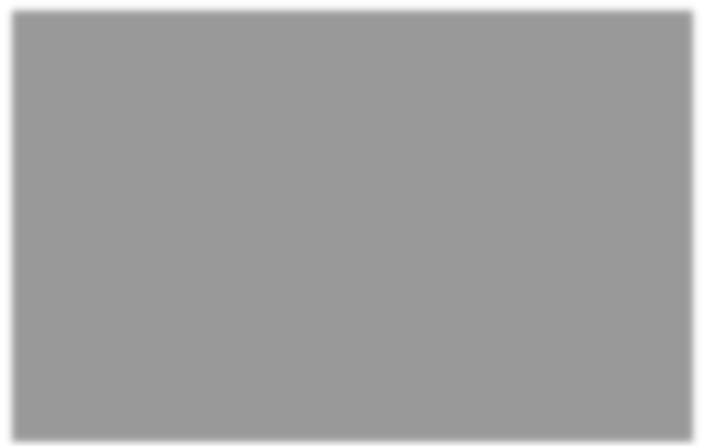
*Devoirs à rendre – Semaine 4 mars 2013*

# Algorithme / Exercices à rendre

**– semaine 4 –**

Système expert : trouver le PANGOLINS



Dans les exercices qui vous sont demandés cette semaine, nous allons utiliser une structure de données proche de celle écrite dans la séance de TD de la semaine 3. Nous allons coder en java. La connaissance zoologique des pangolins est un plus . L'objectif est de créer et de parcourir une structure de données récursive classique.

On se propose de programmer, dans un cas très simplifié, le type de programmes d’intelligence artificielle présenté par exemple à [http://y.20q.net/anon-fr.](http://y.20q.net/anon-fr) Ce type de programmes s’apparente à des systèmes experts, très utilisés dans plusieurs domaines. Un système expert est un système logiciel disposant d’une base de connaissances (représentées par des faits et des règles) destinée à aiguiller l’utilisateur du système vers une déduction.

Par exemple :

Question posée par le système (Q) : fièvre élevée ? Réponse utilisateur (R) : oui Q : courbatures ? R : oui

Q : maux de tête ? R : oui

Q : maux de gorge ? R : oui

Q : conjonctivite ? R : éventuellement

Q : grande fatigue ? R : éventuellement

Diagnostique proposé par la machine : très vraisemblablement grippe.

Ce type de programmes, prenant en compte des incertitudes et des probabilités (réponse

« éventuellement »), est assez élaboré. Nous allons nous placer dans un cadre restreint où les seules réponses possibles sont « oui » et « non ».

Dans notre cas, la machine doit deviner un animal auquel le joueur pense, en posant des questions dont la réponse peut être « oui » ou « non » uniquement.

Image : <http://www.deltacollege.edu/emp/jdebow/pangolin.jpg>

Une fois que la machine a suffisamment de questions en mémoire, un scénario pourrait être par exemple :

*Machine : Pensez à un animal. [Vous pensez à un chat]*

*Machine : Est-ce un mammifère ? Joueur : Oui*

*Machine : Est-ce un mammifère terrestre ? Joueur : Oui*

*Machine : Est-ce un carnivore ? Joueur : Oui*

*Machine : Est-ce un chien ? Joueur : Non*

*Machine : Qu'est-ce ?*

*Joueur : un chat*

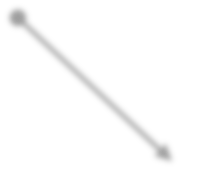
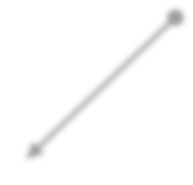
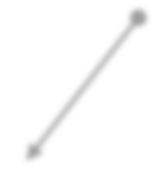
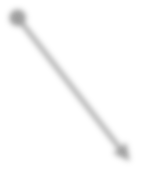
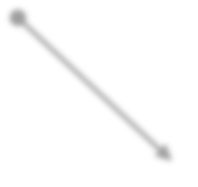
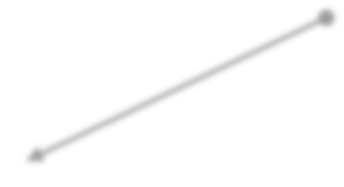
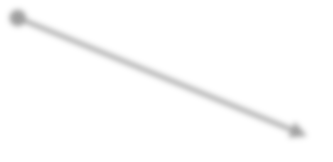
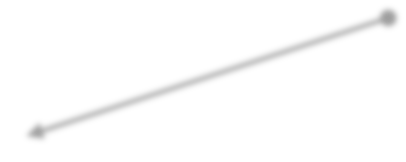
*Machine : Donner une question permettant de distinguer un chat d'un chien. Joueur : « Est-ce que ça aboie ? »*

*Machine : Voulez-vous rejouer ?*

En regardant ce qui se passe à la ligne bleu, on remarque que l'ordinateur propose une solution. Si c'est la bonne solution, il a gagné, s'il ne trouve pas l’animal recherché, il enrichit sa base de questions (les lignes suivantes). Dès lors, il est possible de commencer une partie avec une base de questions vide, les premières parties ne consistant qu’à enrichir les données.

Pour la structure de données, nous allons nous inspirer de la Liste imaginée en semaine 3. Ici l'élément de base de la structure de données ne sera pas un 'simple' *Maillon*, mais une Question contenant deux éléments suivants : la Question qui doit être posée si on répond "oui" à la question courante et la Question qui correspond au "non". La base Pangolin, quant à elle, contient uniquement la racine de la structure arborescente de questions.

Autrement dit, la structure de la base sera de cette forme (les types sont indiqués en bleu):



*Est-ce un mammifère ?*

*Est-ce un mammifère terrestre ?*

*Est-ce un oiseau ?*

*Est-ce un carnivore ?*

*Est-ce un … ?*

*Est-ce un … ?*

*C'est un chien*

*C'est un pangolin*

racine

Pangolin

Question

Question

Question

Question

Question

Question

Question

Question

On remarque ainsi qu'un nœud terminal (une *feuille* dans le langage des arbres) est une question comme les autres ("*C'est un chien*"), sauf qu'elle n'a pas de suivants.



**(SEM02AR01)**

Imaginer d’autres scénarios. Dérouler deux exemples correspondant à des suites de questions allant de la question initiale jusqu’à la proposition finale. Ces exemples seront des tests pour le programme final, ils doivent être rédigés soigneusement.

**(SEM02AR02)**



Proposer une classe Question permettant de représenter les questions et les propositions finales (feuilles). Cette question aura trois attributs : le texte de la question, une question pour la réponse oui et une question pour la réponse non. Les questions *courantes* et les *propositions finales* se distinguent uniquement parce que les dernières n'ont pas d'éléments suivants.

Compléter la classe Question en déclarant ses trois attributs et en implémentant les méthodes données dans la spécification ci-dessous et la tester dans une classe Test.

public class Question {

// déclaration des attributs

...

// Constructeur d'une Question qui prend en paramètre les valeurs

// correspondant aux différents attributs

public Question(String texte, Question o, Question n){...}

// renvoie vrai si la question est une feuille, faux sinon public boolean estTerminale(){...}

// affiche le texte de la question et attend la saisie de la réponse

// de l'utilisateur. Si la réponse est Oui et qu'on n'est pas sur un

// question terminale, pose la question correspondant à oui; sinon,

// pose la question correspondant à non. (Attention la réponse

// doit être lue sous forme de chaîne de caractères et convertie en

// booléen car on ne sait pas lire un booléen au clavier). Si l'on

// est sur une question terminale, propose un animal. public void poser() {...}

}// fin classe Question.



**(SEM02AR03)**

Proposer une classe Pangolin permettant de représenter l'arbre des questions. Cette classe a pour attribut une seule question (la racine de l'arbre).

Compléter les méthodes suivantes dans la classe Pangolin et la tester dans la classe Test.

public class Pangolin {

...

// Crée une base de connaissance ne contenant qu'un seul animal : le

// Pangolin. Pangolin(){...}

void demarrerPartie(){...}

}// fin classe Pangolin.



On vous donne une première version d'un programme principal assez minimaliste. A vous de le faire évoluer par la suite pour avoir un programme plus attractif…

**public class** Main {

**public static void** main(String[] args) {

Question pangol = **new** Question ("C'est un Pangolin", **null**, **null**); Question canari = **new** Question ("C'est un Canari", **null**, **null**); Question dauphin = **new** Question ("C'est un Dauphin", **null**, **null**); Question papillon = **new** Question ("C'est un Papillon", **null**, **null**);

Question intermed1 = **new** Question ("Est-ce qu'il chante ?",canari, papillon); Question intermed2 = **new** Question ("Est-ce qu'il nage en pleine mer ?",

dauph, pang);

Question racine = **new** Question ("Est-ce un mammifère ?, intermed2,

intermed1); Pangolin pang = **new** Pangolin (racine); Pangolin.demarrerPartie();

}

}



**(SEM02AR04) Facultatif**

Lorsque l'on arrive sur une feuille en cours de partie, l'ordinateur pose la question du type "C'est un Chien ?". Si la réponse de l'utilisateur est non, il faut demander à l'utilisateur de saisir au clavier le nom de l'animal auquel il pensait, puis, lui demander une nouvelle question qui permette de différencier les Chien de l'animal auquel il pensait.

Codez la méthode void addQuestion) dans la classe Question, qui permet de rajouter cette question ainsi que le nouvel animal afin d'enrichir la base de connaissance.

// demande à l'utilisateur un nouvel animal et crée la feuille

// correspondante ; demande à l'utilisateur de saisir une question

// qui différencie cet animal de l'animal contenu dans question

// courante et fait le bon chainage. public void addQuestion() {...}

Modifiez la méthode poser de la classe question afin de rajouter cette nouvelle question quand on est sur une feuille et que la réponse de l'utilisateur est 'non'.



Le main ressemblera, dans ce cas, à quelque chose comme ça :

**public static void** main(String [] args){ Pangolin pgl = **new** Pangolin(); String rejouer = "O"; **while**(rejouer.equals("O")){

pgl.demarrer();

System.out.println("\n\n Voulez-vous rejouer ? (O/N)"); rejouer = Clavier.lireString();

}

}