Université Paris Diderot — Année 2017-2018
L1 - Informatique — 15h30 - 18h00

Examen IP2 Lundi 25 juin 2018

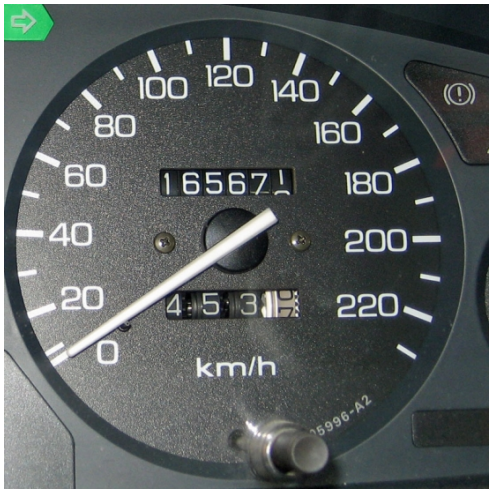
Aucun document autorisé

Ce sujet est composé de 4 exercices traitant chacun une partie de ce que nous avons vu ensemble. Ils sont indépendants. La stratégie la plus efficace pour cette épreuve consiste à répondre très proprement aux questions que vous aborderez en premier, ce qui devrait vous assurer la moyenne. Vous pourrez ensuite passer le temps qui vous reste sur ce qui vous semble difficile. Le barème est indicatif.

Soigner sa présentation est une très bonne idée : n'hésitez pas à utiliser des stylos de couleurs différentes, une règle ... vous avez suffisamment de temps pour ne pas rendre un brouillon.

Exercice 1 (5 points) (Modélisation)

Une entreprise de nom *Lambda* fabrique des compteurs kilométriques, et on cherche à en faire la modélisation sous la forme d'une classe Java.



1. Ecrivez une classe **Compteur** avec tous les éléments (attributs, constructeurs, méthodes ...) qui permettent de répondre aux descriptions que vous trouverez dans le tableau ci-dessous. Ecrivez en le code complet.

2. Reportez sur votre copie les deux dernières colonnes du tableau, et expliquez **en français** comment votre code répond à chacune des descriptions. Si besoin, dites quelles méthodes vos avez écrites pour permettre d'en réaliser la fonctionnalité.

Descriptions	Numéro	Explications
Sur les compteurs est imprimée une unité qui peut être soit km/h, soit miles/h	D1	
Ils peuvent recevoir un signal qui leur indique à quelle vitesse évolue le véhicule	D2	
Un bouton (noir en bas de la photo) sert à remettre à zéro le petit compteur (celui qui indique ici la distance 453,7 km)	D3	
Ils fonctionnent en recevant une information qui leur indique le temps passé à la vitesse courante, considérée constante sur cette période.	D4	
Tous les 10000 km un message s'affiche pour indiquer qu'il faut faire une révision.	D5	
Si l'entreprise se fait racheter, tous les compteurs changent le nom de leur marque	D6	

Exercice 2 (6 points) (Manipulation de cellules)

En cours nous avons vu comment modéliser les files d'attentes en se basant sur une structure de listes chaînées, et en implémentant des méthodes pour ajouter et retirer des éléments dans l'ordre qui correspond à l'idée que l'on se fait d'une file d'attente : le premier arrivé est le premier servi.

1. **(3 points)** Ecrivez les classes, constructeurs, et méthodes qui implémentent une file d'attente, dans le cas où les éléments sont des noms de personnes. (Il s'agit quasiment d'une question de cours)

2. **(3 points)** On s'intéresse à un objet un peu différent, une **file de priorité**. La description de son fonctionnement est :

   - au moment de l'insertion, un paramètre numérique entier supplémentaire est fourni et sera lié à la personne, on l'appelle *valeur de priorité*,
   - au moment du retrait, la personne qui sortira de la file sera, parmi celles de priorité la plus forte, la première qui aura été insérée.

   Il y a plusieurs façons de les implémenter. Notre stratégie sera la suivante : au moment de l'insertion la personne passe devant toutes celles qui ont une priorité strictement plus faible pour trouver sa place.

   Réécrivez complètement deux nouvelles classes qui modélisent une file de priorité avec la stratégie imposée. Vous vous inspirerez au moins des listes doublement chaînées, et écrirez ses constructeurs et ses méthodes d'ajout et de retrait.

Exercice 3 (4.5 points) (Arbres) On considère des arbres binaires simples dont les noeuds portent des étiquettes entières positives. Ces arbres binaires modélisent des arbres (de vrais arbres) : la valeur portée par une feuille correspond à l'énergie qu'elle tire de la photosynthèse. Cette énergie est transmise à son père et s'accumule ainsi de proche en proche, jusqu'à la racine.

Mais un noeud intermédiaire est limité dans l'énergie qu'il peut recevoir, c'est le sens qu'on donne à la valeur portée par ce noeud, et il ne pourra pas en transmettre davantage à son propre père.

- **(0.5 points)** Ecrivez les déclarations des attributs des classes utilisées pour modéliser les arbres.
- **(1.5 points)** Ecrivez une méthode qui retourne la valeur de l'énergie collectée par l'arbre : celle qui est remontée jusqu'à la racine.
- **(2.5 points)** Puisque chaque noeud intermédiaire peut être la cause d'une perte dans la remontée d'énergie, le jardinier s'intéresse au noeud qui en perd le plus. Analysez librement le problème et proposez une méthode qui retourne un tel noeud.

Exercice 4 (3×1.5 points) (Récursion linéaire)
Dans cet exercice on s'intéresse à des listes simplement chaînées contenant des entiers.
Donnez une explication en français, la plus synthétique possible, de ce que calculent les méthodes g, h et k. (On souhaite savoir si vous comprenez bien à quoi elles peuvent servir)
Illustrez également votre explication par un exemple.

```
public class Cell {
 private int content;
 private Cell next;

 // premiere methode
 public int g(){
  return g_aux(0, this.content);
 }

 private int g_aux(int a, int b){
  if (next==null) return a+1;
  if (next.content==b)
    return next.g_aux(a+1, b);
    else return next.g_aux(0, next.content);
 }

 // deuxieme methode
 public int h(){
  return h_aux(0, this.content);
 }

 private int h_aux(int a, int b){
  if (next==null || next.content != b) return a+1;
  else return next.h_aux(a+1, b);
 }

 // troisieme methode
 public int k(){
  return k_aux(1,1);
 }

 private int k_aux(int a, int b){
  if (next==null) return Math.max(a,b);
  if (next.content==this.content)
    return next.k_aux(a, b+1);
    else return next.k_aux(Math.max(a,b),1);
 }
```