Seul document autorisé : une feuille A4 recto-verso manuscrite (non photocopiée). Aucune machine.
Le barème est indicatif. Une question peut toujours être traitée en utilisant les précédentes (traitées ou non).
Les morceaux de code Java devront être clairement présentés, indentés et commentés.
L'utilisation de collections de l'API Java (en particulier `LinkedList` et `ArrayList`) est interdite.

- Tous les attributs d'instance de toutes les classes introduites doivent être privés.
- Il est possible d'écrire des méthodes non spécifiquement demandées si c'est plus pratique, en spécifiant pour chaque méthode dans quelle classe elle se trouve.

On veut modéliser la salle d'attente des urgences d'un hôpital : les patients ne sont pas examinés dans l'ordre de leur arrivée, mais en fonction de la gravité de leur cas.

1 Patient

Un patient est représenté par son nom, son prénom, un identifiant entier unique et un entier signalant la gravité de son cas (de `0` : urgence absolue, à `10` : pas grave).

On modélise ainsi un patient comme une instance de la classe **`Patient`** qui contient donc au moins : deux attributs de type chaîne de caractères : **`nom`**, **`prenom`**, et deux attributs entiers : **`id`** et **`gravite`**. L'interface de cette classe est donnée par :

```
/** renvoie l'identifiant du patient (entier unique) */
int id();

/** renvoie une presentation avec les nom, prenom et identifiant du patient */
String toString();

/** affecte a l'attribut gravite la valeur du parametre,
si ce dernier n'est pas compris entre 0 et 10, affecte la valeur 10 */
void setGravite(int gravite);

/** renvoie la valeur de l'attribut gravite */
int getGravite();
```

Cette classe possède en outre deux constructeurs. L'un d'eux prend en paramètre deux chaînes de caractères représentant respectivement le nom et le prénom du patient ; l'autre prend un argument supplémentaire représentant la gravité de sa situation.

Exercice 1. (4,5 points) Écrire la classe **`Patient`** (attributs, constructeurs et méthodes).

2 Salle d'attente

Dans cette partie, on modélise la salle d'attente sous forme d'une liste chaînée : la tête de liste est représentée par la classe **`Urgence`**, les cellules sont représentées par la classe **`Cellule`**.

Lorsqu'un nouveau patient arrive, il est ajouté à la liste en fonction de sa gravité, de façon à ce que les éléments de la liste soient rangés dans l'ordre croissant des gravités.

L'interface de la classe **`Urgence`** contient au moins les méthodes suivantes :

```
/** renvoie le nombre de patients dans la salle d'attente (ie la longueur de la liste) */
int nbPatients();

/** renvoie le nombre de patients en urgence absolue (ie gravite=0) */
int nbUrgencesAbsolues();

/** affiche la liste des patients (nom, prenom et identifiant) */
void listePatients();

/** affiche tous les patients dont la gravite est donnee en parametre */
void listeParGravite(int gravite);

/** ajoute un nouveau patient dans la liste, en respectant l'ordre des etats de gravite */
void ajouterNouvPatient(Patient p);

/** supprime de la liste un patient donne par son identifiant
```

```
 * @return true si le patient existait, false sinon */
boolean suppressionPatient(int id);
```

Exercice 2. (3 points) Donner les attributs et les constructeurs des classes **`Urgence`** et **`Cellule`**. Il s'agit de décider quels constructeurs écrire pour que ces deux classes soient utilisables et qu'il soit possible d'écrire toutes les méthodes données dans le cadre ci-dessus pour **`Urgence`**.

Exercice 3. (2 points) Écrire les méthodes **`nbPatients`** et **`nbUrgencesAbsolues`**.

Exercice 4. (2 points) Écrire les méthodes **`listePatients`** et **`listeParGravite`**.

Exercice 5. (1,5 points) Écrire la méthode **`suppressionPatient`**.

Exercice 6. (2 points) Écrire la méthode **`ajouterNouvPatient`**. La règle d'ajout est la suivante : un patient est ajouté le plus loin possible dans la liste, de façon à ce que l'ordre des niveaux de gravité soit croissant, voir exemple en figure 1.

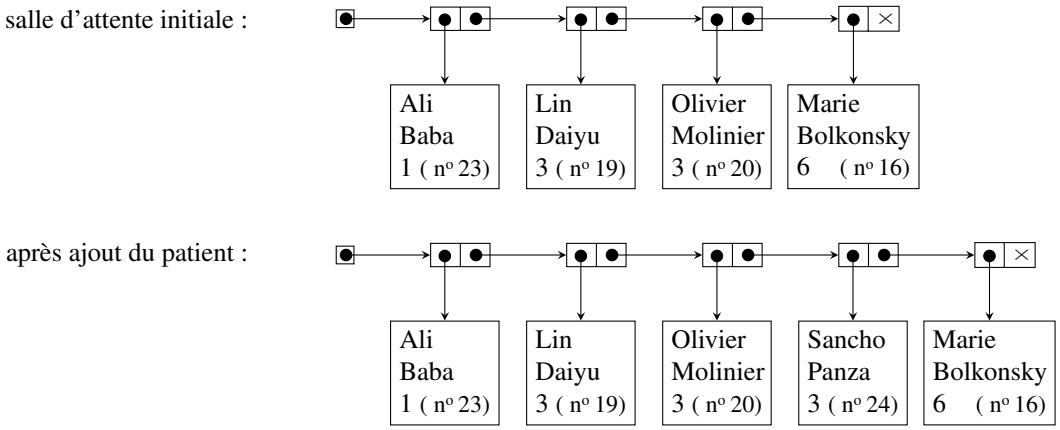

FIGURE 1 – Ajout du patient Sancho Panza dont le niveau de gravité est 3.

3 Médecins

On cherche à gérer l'attribution des patients aux différentes équipes médicales. Afin de simplifier la situation, on suppose que la salle d'attente n'est jamais vide (sans avoir à le vérifier), que chaque équipe s'occupe en permanence d'un patient (pas toujours le même, bien entendu) et que le nombre d'équipes dans le service est constant, soit **`NB_EQUIPES`**. Chaque équipe possède un identifiant entre **`0`** et **`NB_EQUIPES-1`**. L'attribution d'un patient à une équipe se fait à l'aide d'un tableau : la case **`i`** du tableau contient une référence vers le patient traité par l'équipe d'identifiant **`i`** (étant entendu que chaque équipe traite un seul patient à la fois). On ajoute donc les attributs nécessaires à la classe **`Urgence`** : le nombre **`NB_EQUIPES-1`** et le tableau de correspondance équipe médicale/patient.

Exercice 7. (1,5 points) Donner les déclarations des attributs supplémentaires de la classe **`Urgence`** en expliquant les choix faits (types et éventuels qualificatifs).

Exercice 8. (1,5 points)

1. Écrire une fonction
   `String toString(int equipeMed);`
   qui renvoie pour l'équipe médicale dont l'identifiant a été fourni en paramètre une présentation du patient qu'elle est en train de traiter (nom, prénom et identifiant).
2. Dans quelle(s) classe(s) cette fonction peut-elle être écrite ? Justifier.

Exercice 9. (2 points) Quand une équipe médicale termine de s'occuper d'un patient, deux situations peuvent se présenter :

— le patient rentre chez lui. Dans ce cas l'équipe récupère immédiatement le premier patient de la liste d'attente (qui est d'une part attribué à l'équipe *via* le tableau de correspondance équipe médicale/patient, et d'autre part supprimé de la liste) ;

— le patient reste pour des examens complémentaires, la gravité de son cas étant remise à jour, et il retourne donc dans la salle d'attente (il est ajouté dans la liste des malades) ; puis l'équipe médicale récupère immédiatement le premier patient de la liste d'attente (qui est d'une part attribué à l'équipe médicale *via* le tableau de correspondance équipe médicale/patient, et d'autre part supprimé de la liste d'attente).

1. Écrire deux méthodes dans la classe **`Urgence`** qui correspondent aux deux situations précédentes.
2. Ces deux méthodes peuvent-elles avoir le même nom ? Justifier.