Corrigé des exercices sur les structures de données

Exercice 11.1 employés et services

Question 1

1. Ajouter au programme une méthode pour afficher la liste des employés avec pour chaque employé le service auquel il appartient.
2. Ajouter au programme une méthode qui affiche les employés travaillant dans un certain service. La méthode demandera à l'utilisateur de choisir un service via une sorte de menu.
3. Ajouter une méthode qui affiche service par service la liste de tous les employés.
4. Une personne dénommée Xavier se présente à l'embauche dans cette entreprise. A votre avis, sera-t-il embauché ?

```
public class Depart{
   public static void main(String[] args){
      String[] departs = {"bureau d'étude","ressources humaines",
                     "service commercial","gestion financière"};
      String[] employes = {"Jean", "John", "Jeannot","Jeanne","Jeannette",
                      "Janeton", "Giovanna", "Juan", "Juanito"};
      int[] service = {0, 1, 0, 2, 1, 3, 3,2, 0};
      printEmployes(departs,employes,service);
      System.out.println();
      employesDuDep(departs,employes,service);
      System.out.println();
      printByDepartment(departs,employes,service);
   }
   public static void printEmployes(String[] serv, String[] noms,
                            int[] affect){
      for (int i=0; i<noms.length; i++){
         System.out.println(noms[i] + " affecté à " + serv[affect[i]]);
      }
   }
   public static void employesDuDep(String[] serv, String[] noms,
                            int[] affect){
      int leserv;
```

```
      for (int i=0; i<serv.length;i++){
         System.out.println((i+1) + " service " + serv[i]);
      }
      System.out.println("Entrez le numéro du service: ");
      leserv = Terminal.lireInt()-1;
      System.out.println("Employés du service " +serv[leserv]);
      for (int i=0; i< affect.length; i++){
         if (affect[i] == leserv)
            System.out.println("   " + noms[i]);
      }
   }
   public static void printByDepartment(String[] serv, String[] noms,
                                int[] affect){
      for (int leserv=0; leserv<serv.length; leserv++){
         System.out.println("Employés du service " +serv[leserv]);
         for (int i=0; i< affect.length; i++){
            if (affect[i] == leserv){
               System.out.println("   " + noms[i]);
            }
         }
      }
   }
}
```

Question 2

1. Modifiez le programme pour que la liste des employés puisse changer en utilisant la méthode des grands tableaux avec cases vides à droite. Modifiez les tableaux du main et les méthodes pour les adapter à cette nouvelle représentation.
2. Écrivez une méthode permettant d'ajouter un nouvel employé.
3. Écrivez une méthode permettant de retirer un employé partant à la retraite.

```
public class Depart2{
   public static void main(String[] args){
      String[] departs = {"bureau d'étude","ressources humaines",
                          "service commercial","gestion financière"};
      String[] preemployes = {"Jean", "John", "Jeannot","Jeanne",
                              "Jeannette","Janeton", "Giovanna",
                              "Juan", "Juanito"};
      int[] preservice = {0, 1, 0, 2, 1, 3, 3,2, 0};
      String[] employes;
      int[] service;
      int[] nbemp = {0};
      employes = new String[100];
      service = new int[100];
      for (int i=0; i<preemployes.length; i++){
```

```
            employes[i]=preemployes[i];
            service[i]=preservice[i];
            nbemp[0] = nbemp[0]+1;
        }
        printEmployes(departs,employes,service,nbemp);
        System.out.println();
        employesDuDep(departs,employes,service,nbemp);
        System.out.println();
        printByDepartment(departs,employes,service,nbemp);
        System.out.println();
        embaucher(departs,employes,service,nbemp);
        printEmployes(departs,employes,service,nbemp);
        System.out.println();
        enRetraite(employes,service,nbemp);
        printEmployes(departs,employes,service,nbemp);
    }
    public static void printEmployes(String[] serv, String[] noms,
                                int[] affect, int[] nb){
        for (int i=0; i<nb[0]; i++){
            System.out.println(noms[i] + "␣affecté␣à␣" + serv[affect[i]]);
        }
    }
    public static void employesDuDep(String[] serv, String[] noms,
                                int[] affect, int[] nb){
        int leserv;
        for (int i=0; i<serv.length;i++){
            System.out.println((i+1) + "␣service␣" + serv[i]);
        }
        System.out.println("Entrez␣le␣numéro␣du␣service:␣");
        leserv = Terminal.lireInt()-1;
        System.out.println("Employés␣du␣service␣" +serv[leserv]);
        for (int i=0; i< nb[0]; i++){
            if (affect[i] == leserv)
                System.out.println("␣␣␣" + noms[i]);
        }
    }
    public static void printByDepartment(String[] serv, String[] noms,
                                    int[] affect, int[] nb){
        for (int leserv=0; leserv<serv.length; leserv++){
            System.out.println("Employés␣du␣service␣" +serv[leserv]);
            for (int i=0; i< nb[0]; i++){
                if (affect[i] == leserv){
                    System.out.println("␣␣␣" + noms[i]);
                }
            }
        }
    }
```

```
   public static void embaucher(String[] serv, String[] noms,
                                int[] affect, int[] nb){
      String nom;
      int numserv;
      System.out.print("Nom du nouvel embauché: ");
      nom = Terminal.lireString();
      for (int i=0; i<serv.length;i++){
         System.out.println((i+1) + " service " + serv[i]);
      }
      do{
         System.out.println("Entrez le numéro du service de " +
                        nom + ": ");
         numserv = Terminal.lireInt()-1;
      }while(numserv<0  numserv>= serv.length);
      noms[nb[0]]=nom;
      affect[nb[0]]=numserv;
      nb[0]=nb[0]+1;
      System.out.println("Embauche terminée");
   }
   public static void enRetraite(String[] noms, int[] affect,
                          int[] nb){
      int numemp;
      for (int i=0; i<nb[0]; i++){
         System.out.println((i+1) + " - " + noms[i]);
      }
      do{
         System.out.print("Numéro de l'employé à la retraite: ");
         numemp = Terminal.lireInt()-1;
      }while(numemp<0  numemp>=nb[0]);
      noms[numemp]=noms[nb[0]-1];
      affect[numemp]=affect[nb[0]-1];
      nb[0] = nb[0]-1;
      System.out.println("Départ à la retraite enregistré");
   }
}
```

Exercice 11.2 prix des billets d'autocar

Question 1

1. écrivez une méthode permettant de retrouver l'indice d'une ville dont on donne le nom en paramètre (c'est à dire sa position dans le premier tableau).
2. écrivez une méthode qui calcule le prix d'un trajet étant donnés les noms des villes de départ et d'arrivée.

```
class Erreur extends RuntimeException{}
```

```
class Exo12_2{
   static int indice(String[] lesVilles,String ville){
      for (int i=0; i<lesVilles.length; i++){
         if (ville.equals(lesVilles[i])){
            return i;
         }
      }
      throw new Erreur();
   }
   static double prixTrajet(String[] lesVilles, double[] prixTroncon,
                     String ville1, String ville2) {
      int v1, v2, debut, fin;
      double prix = 0;
      v1 = indice(lesVilles,ville1);
      v2 = indice(lesVilles,ville2);
      if (v1<v2){
         debut = v1;
         fin = v2;
      }else{
         debut = v2;
         fin = v1;
      }
      for (int i=debut; i<fin; i++){
         prix = prix + prixTroncon[i];
      }
      return prix;
   }
   public static void main(String[] args) {
      String[] villes = {"Vierzon", "Salbris", "Nouans", "Lamotte-Beuvron",
                     "La Ferte Saint-Aubin", "Orleans"};
      double[] troncons = {3.2,1.8,2.3,4.2,5.0};
      Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                              "Orleans","Salbris"));
      Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                              "Salbris","Lamotte-Beuvron"));
      Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                              "Salbris","Salbris"));
   }
}
```

Dans ce programme, la boucle `for` de la méthode `indice` ne s'exécute pas complètement : dès qu'on a trouvé une case de tableau ayant le bon nom de ville, l'instruction `return` renvoie le numéro de cette case, ce qui provoque la fin de l'exécution de la méthode et donc du `for`. Il y ainterruption du parcours du tableau quand on a trouvé le résultat recherché. En revanche, si on sort de la boucle autrement que par ce `return`, cela signifie qu'on n'a pas trouvé la ville. C'est alors que l'exception `Erreur` est levée.

Notez que les deux méthodes `indice` et `prixTrajet` sont deux pures fonctions : elles ob-

tiennent les données utiles par des paramètres (nombreux ici) et renvoient leur résultat par `return`.

Question 2

On veut instaurer des tarifs dégressifs selon le nombre de tronçons parcourus : le premier tronçon est payé à plein tarif, le second avec 10% de réduction, le second avec 20%, etc. Ecrivez une méthode qui réalise le calcul du prix d'un trajet selon ce principe.

```
class Erreur extends Exception{}
class Exo12_2_2{
   static int indice(String[] lesVilles,String ville) {
      for (int i=0; i<lesVilles.length; i++){
         if (ville.equals(lesVilles[i])){
            return i;
         }
      }
      throw new Erreur();
   }
   static double prixTrajet(String[] lesVilles, double[] prixTroncon,
                     String ville1, String ville2) {
      int v1, v2, debut, fin;
      double prix = 0;
      double facteur = 1;
      v1 = indice(lesVilles,ville1);
      v2 = indice(lesVilles,ville2);
      if (v1<v2){
         debut = v1;
         fin = v2;
      }else{
         debut = v2;
         fin = v1;
      }
      for (int i=debut; i<fin; i++){
         prix = prix + (prixTroncon[i] *facteur);
         if (facteur > 0){
            facteur = facteur -0.1;
         }
      }
      return prix;
   }
   public static void main(String[] args) {
      String[] villes = {"Vierzon", "Salbris", "Nouans", "Lamotte-Beuvron",
                  "La Ferte Saint-Aubin", "Orleans"};
      double[] troncons = {3.2,1.8,2.3,4.2,5.0};
      Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                              "Orleans","Salbris"));
      Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                              "Salbris","Lamotte-Beuvron"));
```

```
        Terminal.ecrireDoubleln(prixTrajet(villes,troncons,
                                    "Salbris","Salbris"));
    }
}
```

Question 3

On veut généraliser le système pour un ensemble de lignes de bus : que faut-il changer dans le programme ? Vous écrirez un programme avec deux lignes de bus et un seul programme qui calcule les prix des trajets. Les billets sont pour un trajet sur une ligne donnée, on ne fait pas de billet avec correspondance.

Tel qu'on l'a écrit, notre programme fonctionne déjà pour un nombre quelconque de lignes de bus. La seule contrainte est qu'il faut passer en paramètre à `prixTrajet` deux tableaux différents qui décrivent une seule et même ligne de bus. Le premier donne les noms des arrêts desservis et le second, le prix de chaque tronçon.

Exercice 11.3 planning mensuel

Les abonnés d'un transporteur ferroviaire bénéficient d'un tarif très avantageux à condition d'éviter les jours de pointe. Le transporteur publie chaque mois le planning des jours de pointe du mois suivant. Le fait d'être un jour de pointe ou pas concerne la totalité d'une journée. Il s'agit de représenter le planning des jours de pointe d'un mois en Java.

Un analyste programmeur hésite entre deux représentations pour un tel planning :

— un tableau de booléens de 31 cases donnant pour chaque tantième du mois si oui ou non c'est un jour de pointe.

— un tableau d'entiers contenant les tantièmes des jours de pointe. La taille de ce tableau pourra varier d'un mois à l'autre selon le nombre de jours de pointes.

Question 1

On suppose que pour le mois prochain, les 10, 11, 12, 17 et 22 du mois sont des jours de pointe. Dessinez les deux structures proposées.

Choix 1 :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	
F	F	F	F	F	F	F	F	F	T	T	T	F	F	F	F	T	F	F	F	…

	20	21	22	23	24	25	26	27	28	29	30
…	F	T	F	F	F	F	F	F	F	F	F

Choix 2 :

0	1	2	3	4
10	11	12	17	22

Question 2

On veut écrire une méthode qui vérifie si un jour est un jour de pointe.

```
public static boolean estJourDePointe1(boolean[] tab, int jour){
   boolean resultat = false;
   if (tab[jour-1]){
      resultat = true;
   }else{
      resultat = false;
   }
   return resultat;
}
public static boolean estJourDePointe2(int[] tab, int jour){
   boolean resultat = false;
   for (int num=0; num<tab.length; num=num+1){
      if (tab[num]==jour){
         resultat = true;
      }
   }
   return resultat;
}
```

La première représentation permet d'avoir directement l'information cherchée alors que le seconde nécessite un parcours du tableau. La première est donc meilleure pour cette opération.

Question 3

```
public static int nbVoyagesJDP1(boolean[] tabjdp, boolean[] tabvoy){
   int res = 0;
   for (int num= 0; num<31; num=num+1){
      if (tabjdp[num] && tabvoy[num]){
         res = res + 1;
      }
   }
   return res;
}
public static int nbVoyagesJDP2(int[] tabjdp, int[] tabvoy){
   int res = 0;
   for (int num = 0; num<tabjdp.length; num=num+1){
      for (int num2=0; num<tabvoy.length; num=num+1){
         if (tabjdp[num]==tabvoy[num2]){
            res = res + 1;
         }
      }
   }
   return res;
}
```

Pour cette opération, la représentation avec tableaux de booléens permet de calculer le résultat avec une seule boucle alors qu'avec les tableaux d'entiers il faut deux boucles imbriquées et beaucoup de comparaisons pour obtenir le résultat.

Pour les deux opérations étudiées, les tableaux de booléens sont préférables.