

Corrigé des exercices de révision

NFA032

Exercice 1 *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 Exception{ }
class Exo12_2{
    static int indice(String[] lesVilles,String ville) throws Erreur{
        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) throws Erreur{
        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) throws Erreur{
        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 a interruption 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 obtiennent 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) throws Erreur{
        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) throws Erreur{
        int debut, fin;
        double prix = 0;
        double facteur = 1;
        debut = indice(lesVilles,ville1);
        fin = indice(lesVilles,ville2);
        if (debut<fin){
            for (int i=debut; i<fin; i++){
                prix = prix + (prixTroncon[i] *facteur);
                if (facteur > 0){
                    facteur = facteur -0.1;
                }
            }
        }
        else{
            for (int i=debut; i>fin; i--){
                prix = prix + (prixTroncon[i-1] *facteur);
            }
        }
    }
}

```

```

        if (facteur > 0){
            facteur = facteur -0.1;
        }
    }
}
return prix;
}
}

public static void main(String[] args) throws Erreur{
    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,
        "Salbris","Lamotte-Beuvron"));
    Terminal.ecrireDoubleLn(prixTrajet(villes,troncons,
        "Orleans","Salbris"));
    Terminal.ecrireDoubleLn(prixTrajet(villes,troncons,
        "Salbris","Orleans"));
}
}

```

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 2 *crible d'Erathostène*

Question 1 *version simple du crible*

Ecrivez une fonction qui prend en paramètre le nombre `n` en entrée et qui renvoie le tableau du crible sous la forme d'un tableau de booléens où le booléen note si l'entier correspondant à un indice donné est rayé ou non.

Question 2 *test de primalité*

Ecrivez une fonction qui teste si un nombre est premier. Pour cela, utilisez un crible d'Erathostène et voyez si ce nombre est rayé ou non dans le tableau.

Question 3 *crible, version sophistiquée*

Ecrivez une fonction qui calcule tous les nombres premiers inférieurs à un certain paramètre `n`. Le résultat sera donné sous la forme d'un tableau d'entiers dont tous les éléments sont ledits nombres

premiers. Le calcul est identique à celui de la question 1, mais la forme du résultat est différente.

```
class Exo12_1{
    static boolean[] cribleSimple(int n){
        if (n<1){
            return new boolean[0];
        }
        boolean[] res = new boolean[n+1];
        // initialisation du tableau
        res[0]=false;
        res[1]=false;
        for (int i=2; i<=n; i++){
            res[i]=true;
        }
        // calcul
        for (int i=2; i<=Math.sqrt(n); i++){
            if (res[i]){
                for (int j=2; j*i<=n; j++){
                    res[i*j]=false;
                }
            }
        }
        return res;
    }
    static boolean estPremier(int n){
        return cribleSimple(n)[n];
    }
    static int[] cribleSoph(int n){
        boolean[] tab = cribleSimple(n);
        int[] prov = new int[n-1];
        int nb = 0;
        for (int i=2; i<=n; i++){
            if (tab[i]){
                prov[nb]=i;
                nb++;
            }
        }
        int[] res = new int[nb];
        for (int i=0; i<nb; i++){
            res[i]=prov[i];
        }
        return res;
    }
    public static void main(String[] argv){
        boolean[] test = cribleSimple(100);
        for (int i=0; i<=100; i++){
            if (test[i]){
                Terminal.ecrireString(i+" ");
            }
        }
        Terminal.sautDeLigne();
        int[] retest = cribleSoph(100);
        for (int i=0; i<retest.length; i++){
```

```
        Terminal.ecrireString(retest[i]+"_");  
    }  
    Terminal.sautDeLigne();  
    Terminal.ecrireStringln("15_est-il_premier?_" + estPremier(15));  
    Terminal.ecrireStringln("17_est-il_premier?_" + estPremier(17));  
}  
}
```
