Test technique

Contexte

Une librairie souhaite développer un nouvel outil informatique afin de :

- Importer les informations de son stock
- Obtenir des informations sur un livre en particulier
- Calculer le prix d'un panier de livres

Vous êtes en charge du développement de cet outil dont les fonctionnalités sont détaillées ci-dessous. Il vous est demandé d'implémenter les différentes méthodes exposées par l'interface décrite dans l'annexe 1, en suivant les spécifications ci-dessous.

Exercices

Import des données du stock

L'outil doit être en capacité d'importer les données de stock de la librairie à partir d'un fichier JSON dont le format est spécifié dans l'annexe 2. Il n'est pas demandé de stocker dans une base les informations importées, mais uniquement de les garder en mémoire.

Pour cet exercice, vous devez implémenter la méthode suivante :

```
void Import(string catalogAsJson)
```

Un exemple de JSON est disponible dans l'annexe 3.

Récupération des informations

Le gérant de la librairie doit pouvoir consulter, à partir du titre d'un livre, le stock disponible. Cela sera fait à partir de la méthode suivante :

```
int Quantity(string name)
```

Cette méthode prend en paramètre le titre d'un livre, et retourne le nombre d'exemplaires disponibles (i.e. le champ « Quantity » dans le JSON).

Exemple d'appel :

```
Store.Quantity("Ayn Rand - FountainHead");
```

Résultat :

10

Calcul des prix des paniers

La fonctionnalité la plus importante de l'outil est le calcul des prix des paniers (i.e. une sélection de livres). Cela sera fait en appelant la méthode suivante :

```
double Buy(params string[] basketByNames)
```

Le calcul des prix des paniers doit suivre des règles très précises qui sont décrites dans l'annexe 4.

Exemple d'appel :

```
Store.Buy("J.K Rowling - Goblet Of fire", "Isaac Asimov - Foundation");
```

Résultat :

24.00

Annexes 1 – Format JSON

```
{
  "type": "object",
  "title": "The Root Schema",
  "required": [
    "Category",
    "Catalog"
  ],
  "properties": {
    "Category": {
      "type": "array",
      "title": "List of existing category with associated discount",
      "items": {
        "type": "object",
        "title": "one category with its discount",
        "required": [
          "Name",
          "Discount"
        ],
        "properties": {
          "Name": {
            "type": "string",
            "title": "The unique name of the category, it is a functionnal
key",
            "default": "",
            "examples": [
              "Fantastique"
            ],
            "pattern": "^(.+)$"
          },
          "Discount": {
            "type": "number",
            "title": "the discount applies when buying multiple book of this
category",
            "default": 0.0,
            "examples": [
              0.05
            ]
          }
        }
      }
    },
    "Catalog": {
      "type": "array",
      "title": "The Catalog of the store",
      "items": {
        "type": "object",
        "title": "a book in the catalog",
        "required": [
          "Name",
          "Category",
          "Price",
          "Quantity"
        ],
        "properties": {
```

```
          "Name": {
            "type": "string",
            "title": "The unique Name of the book, it is a functionnal key",
            "default": "",
            "examples": [
              "J.K Rowling - Goblet Of fire"
            ],
            "pattern": "^(.+)$"
          },
          "Category": {
            "type": "string",
            "title": "The name of one the category existing in the Category
root properties.",
            "default": "",
            "examples": [
              "Fantastique"
            ],
            "pattern": "^(.+)$"
          },
          "Price": {
            "type": "number",
            "title": "the price of an copy of the book",
            "default": 0,
            "examples": [
              8
            ]
          },
          "Quantity": {
            "type": "integer",
            "title": "The Quantity of copy of the book in the catalog.",
            "default": 0,
            "examples": [
              2
            ]
          }
        }
      }
    }
  }
}
```

Annexe 2 – Exemple JSON


```
{
    "Category":[
        {
            "Name": "Science Fiction",
            "Discount": 0.05
        },
        {
            "Name": "Fantastique",
            "Discount": 0.1
        },
        {
            "Name": "Philosophy",
            "Discount": 0.15
        },
    ],
        "Catalog": [
                {
                    "Name": "J.K Rowling - Goblet Of fire",
                    "Category": "Fantastique",
                    "Price": 8,
                    "Quantity": 2
                },
                {
                    "Name": "Ayn Rand - FountainHead",
                    "Category": "Philosophy",
                    "Price": 12,
                    "Quantity": 10
                },
                {
                    "Name": "Isaac Asimov - Foundation",
                    "Category": "Science Fiction",
                    "Price": 16,
                    "Quantity": 1
                },
                {
                    "Name": "Isaac Asimov - Robot series",
                    "Category": "Science Fiction",
                    "Price": 5,
                    "Quantity": 1
                },
                {
                    "Name": "Robin Hobb - Assassin Apprentice",
                    "Category": "Fantastique",
                    "Price": 12,
                    "Quantity": 8
                }
        ],
}
```

Annexe 4 – Règle de calculs des prix

Afin de calculer le prix d'un panier plusieurs règle s'applique :

1. L'achat d'un livre seul se paye au prix du livre fourni dans le catalogue.

2. Si le client achète plusieurs livres alors une réduction s'applique si ces livres font partie de la même catégorie

   Exemple : si le client achète deux livres diffèrent dans la catégorie Fantastique une réduction de 10% s'appliquera.

3. Seul le premier exemplaire de chaque livre a le droit à la réduction

   *Exemple 1 : si un client achète un exemplaire de `J.K Rowling - Goblet Of fire` et deux de `Robin Hobb - Assassin Apprentice`, seuls Goblet of fire et un des deux exemplaires de Robin Hobb aura le droit à la réduction (ce qui donnerait un prix de 30 = 8 * 0.9 + 12 * 0.9 + 12).*

   *Exemple 2 : si un client achète un exemplaire de Rand, les deux ouvrages d'Asimov (un exemplaire de `Isaac Asimov - Robot series` et un de `Isaac Asimov - Foundation`) , l'ouvrage de Rowling en deux exemplaire et celui de Hobb en deux exemplaires, il doit payer 69.95 € = 12 + 5 * 0.95 + 16 * 0.95 + 8 *0.9 + 8 + 12 * 0.9 + 12*

4. Si un panier n'est pas valide car le catalogue ne contient pas assez d'ouvrage (par exemple si le client veut acheter deux exemplaires de Isaac Asimov Foundation), le propriétaire s'attend à recevoir une exception de type **NotEnoughInventoryException** contenant la liste des ouvrages introuvables.

Annexe 5 – Définition des interfaces et classes

```
public interface IStore
{
    void Import(string catalogAsJson);

    int Quantity(string name);

    double Buy(params string[] basketByNames);
}


public class NotEnoughInventoryException : Exception
{
    public IEnumerable<INameQuantity> Missing { get; }
}


public interface INameQuantity
{
    string Name { get; }

    int Quantity { get; }
}
```