TP2 : analyse statique

Yasmine Filali , Doha CHEMOAU

October 2021

1 Exo1 : Application d'analyse statique

Afin de pouvoir tester le code sur plusieurs applications, vous pouvez modifier la variable "folder" se trouvant dans la classe tp_ast/exo1/Test.java

1.1 Q1 : Nombre de classes

Nous avons l'affichage suivant sur le terminal, où on peut trouver le nombre de classes ainsi que les noms des classes présentes dans l'application.

```
_____________________________________________________________Q1 : nombre de classes
le nombre total des classes : 9
Voici les classes présentes dans notre application : Liquor , Necessity , Product , TaxHolidayVisitor , TaxVisitor , Test , Tobacco , Visitable , Visitor
```

1.2 Q2 : Nombre de lignes de code de l'application

Ici, nous affichons le nombre total de lignes présentes dans toutes les classes de l'application, y compris les commentaires, sauts de lignes , ...

```
_____________________________________________________________Q2 : nombre de lignes de l'application
Le nombre de lignes dans toute l'application : 341
```

1.3 Q3 : Nombre total de méthodes de l'application

Nous affichons le nombre total des méthodes présentes dans toutes les classes de l'application , y compris les constructeurs.

```
_____________________________________________________________Q3 : nombre total des méthodes
Le nombre total de méthodes sur tout l'ensemble des classes : 25
```

1.4 Q4 : Nombre total des packages de l'application

Nous affichons le nombre total des packages, ainsi que leurs noms.

```
_____________________________________________________________________Q4 : nombre des packages

Le nombre de packages sur l'ensemble des classes : 1

voici les packages présents sur l'ensemble de l'application : behavioral.visitor
```

1.5 Q5 : Nombre moyen des méthodes par classe

Nous récupérons le nombre moyen des méthodes par classe en divisant le nombre total des méthodes par le nombre total des classes. Comme nous avons 9 classes au total et 25 méthodes au total , cela nous fait 2 méthodes en moyenne par classe.

```
_____________________________________________________________Q5 : nombre moyen des méthodes dans une classe

Le nombre moyen de méthodes par classe : 2
```

1.6 Q6 : Nombre moyen de lignes de code par méthode

Nous calculons le nombre de lignes dans chaque méthodes et le divisons pas le nombre de classes. 169 lignes sont consacrées aux méthodes. 25 méthodes présentes sur l'ensemble de l'application. Nous nous retrouvons alors avec 6 lignes en moyenne par méthode.

```
_____________________________________________________________Q6 : nombre moyen de ligne par méthode

le nombre total des lignes consacrées aux méthodes sur toute l'application : 169
le nombre moyen de lignes dédiées aux méthodes dans chaque classe : 6
```

1.7 Q7 : Nombre moyen d'attributs par classe

Nous calculons d'abord le nombre total d'attributs dans toute l'application. Nous trouvons qu'il est égal à 4. Comme nous avons 9 classes, nous divisons 4 par 9 ce qui nous donne 0 attributs en moyenne par classe.

```
_____________________________________________________________________Q7 : nombre moyen d'attributs

le nombre total d'attributs dans toutes les classes : 4

Le nombre moyen d'attributs par classe : 0
```

1.8 Q8 : 10% des classes qui possèdent le plus grand nombre de méthodes

Nous avons donné la possibilité au testeur de rentrer n'importe quel pourcentage qu'il souhaite. Dans l'image ci-dessous , vous pouvez voir les 10% des classes qui ont le plus grand nombre de méthodes.

```
_____________________________________________________________________Q8 : x% des classes qui ont le plus de méthodes

entrez un pourcentage pour savoir quelles sont les classes qui ont le plus grand nombre de méthodes
10
Les 10% des classes qui ont le plus de méthodes : TaxVisitor
```

1.9 Q9 : 10% des classes qui possèdent le plus grand nombre d'attributs

Nous avons donné la possibilité au testeur de rentrer n'importe quel pourcentage qu'il souhaite. Dans l'image ci-dessous , vous pouvez voir les 10% des classes qui ont le plus grand nombre d'attributs.

```
_____________________________________________________________________Q9 : x% des classes qui ont le plus d'attributs

entrez un pourcentage pour savoir quelles sont les des classes qui ont le plus grand nombre d'attributs
10
Les 10% des classes qui ont le plus d'attrbiuts : TaxVisitor
```

1.10 Q10: Les classes qui font partie en meme temps des deux catégories précédentes

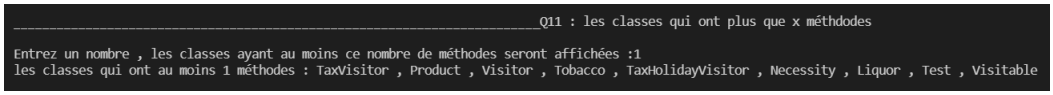
Nous affichons les classes qui font partie des deux catégories précédentes.

```
_____________________________________________________Q10 : les classes qui font partie en meme temps aux deux catégories précédentes

les classes qui font partie des 2 catégories précédentes : TaxVisitor
```

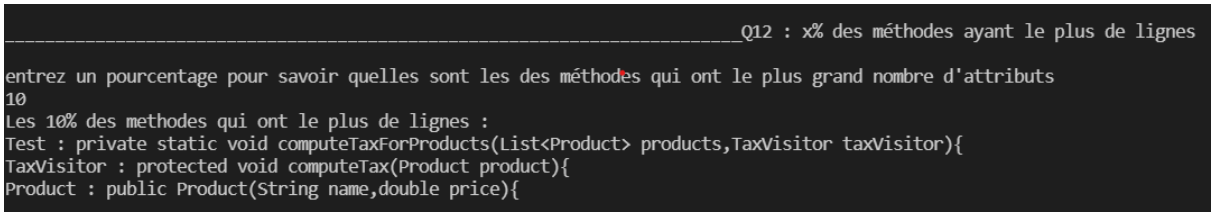
1.11 Q11: Les classes qui possèdent plus que X méthodes

Le testeur peut rentrer un nombre. Les classes ayant au moins ce nombre de méthodes seront affichées.



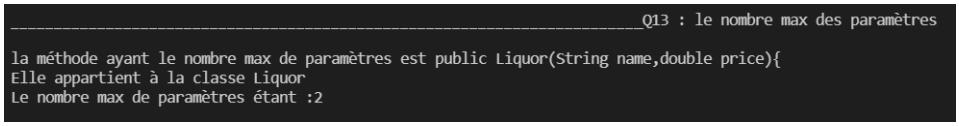
1.12 Q12: 10% des méthodes qui possèdent le plus grand nombre de lignes de code

Sont affichées les méthodes ayant le plus grand nombre de lignes.



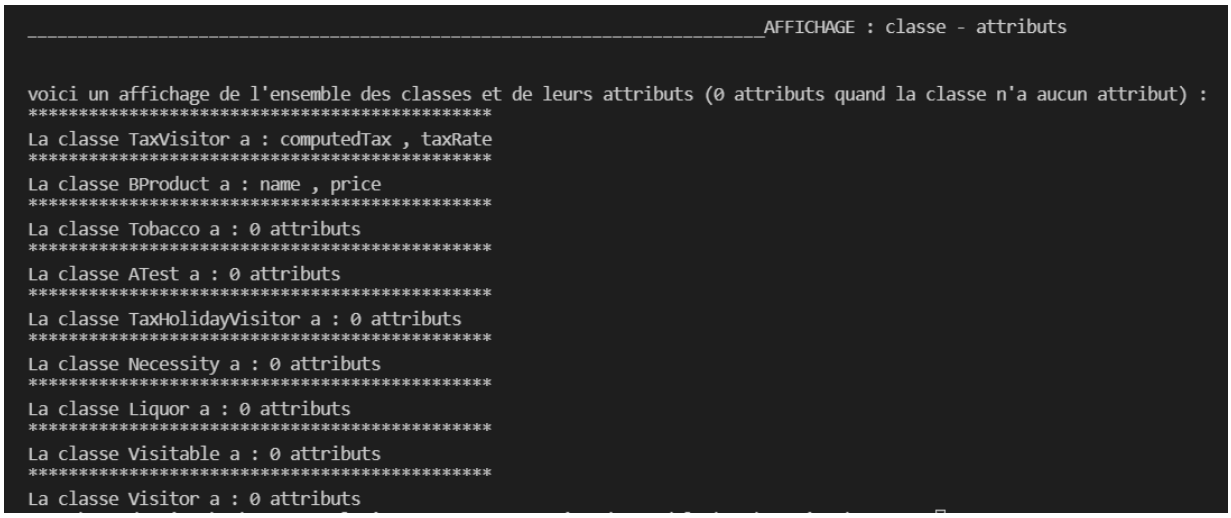
1.13 Q13: Le nombre maximal de paramètres par rapport à toutes les méthodes de l'application

Le nombre maximal de paramètres figurant sur toutes les méthodes de l'application ainsi que la méthode en question sont affichés.



1.14 Quelques affichages vers la fin

Veuillez trouver:
– l'affichage classe - attributs , ou on peut trouver les noms d'attributs présents dans chaque classe.

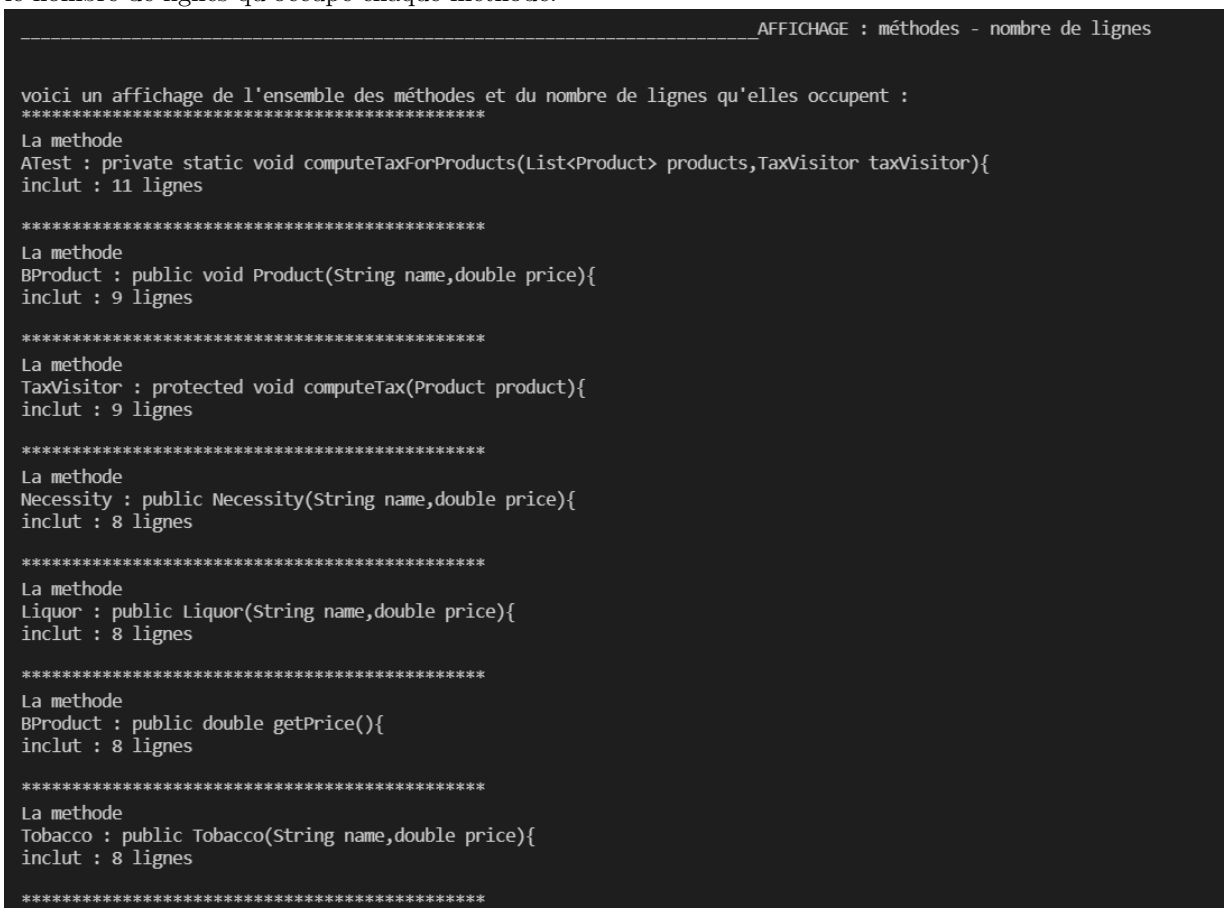

_____AFFICHAGE : classe - attributs
voici un affichage de l'ensemble des classes et de leurs attributs (0 attributs quand la classe n'a aucun attribut) :
****
La classe TaxVisitor a : computedTax , taxRate
****
La classe BProduct a : name , price
****
La classe Tobacco a : 0 attributs
****
La classe ATest a : 0 attributs
****
La classe TaxHolidayVisitor a : 0 attributs
****
La classe Necessity a : 0 attributs
****
La classe Liquor a : 0 attributs
****
La classe Visitable a : 0 attributs
****
La classe Visitor a : 0 attributs

– l'affichage classe-méthodes, ou on peut trouver les noms des méthodes présentes dans chacune des classes.

```
_____________________________________________________________________AFFICHAGE : classe : méthodes


voici un affichage de l'ensemble des classes et de leurs méthodes :
*****************************************
La classe TaxVisitor a : getComputedTax , getTaxRate , computeTax , visit , visit , visit
*****************************************
La classe BProduct a : Product , getName , setName , getPrice , setPrice
*****************************************
La classe Visitor a : visit , visit , visit
*****************************************
La classe Tobacco a : Tobacco , accept
*****************************************
La classe ATest a : main , computeTaxForProducts
*****************************************
La classe TaxHolidayVisitor a : visit , visit
*****************************************
La classe Necessity a : Necessity , accept
*****************************************
La classe Liquor a : Liquor , accept
*****************************************
La classe Visitable a : accept
```

– un extrait de l'affichage méthode - nombre de lignes , ou on peut trouver le nombre de lignes qu'occupe chaque méthode.



1.15 Remarque

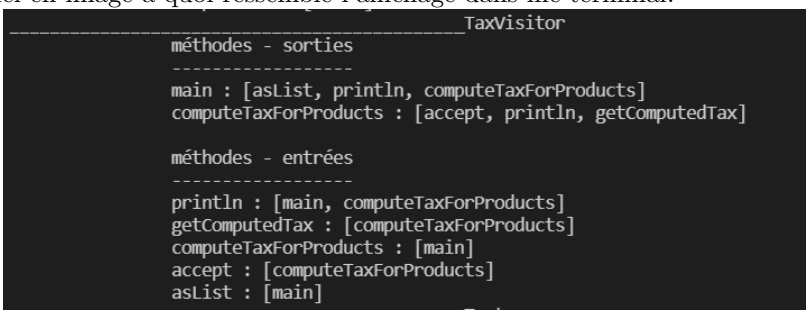
Nous tenons à préciser que l'affichage dans le terminal ne suit pas l'ordre des questions, mais comme vous l'aurez remarqué , il y a une indication au dessus de chaque partie ou on peut trouver le numéro de la question ainsi que ce qui est demandée dans celle-ci.

2 Exo2 : Graphe d'appel de l'application

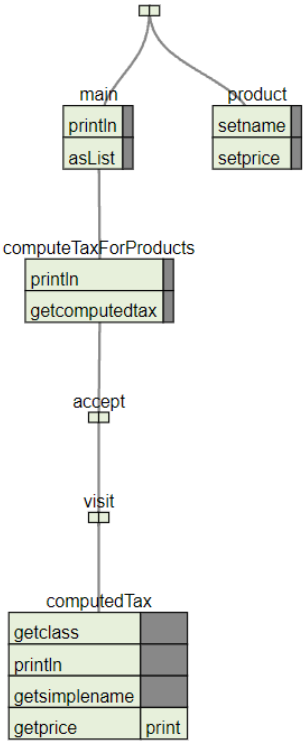
Nous avons affiché pour chaque méthode les méthodes en sorties (méthodes appelées dans le corps d'une méthode) et les méthodes en entrées (par quelle méthode est appelée la méthode en question)
Prenons l'exemple de la classe TaxVisitor.

– Les méthodes en sorties :
La méthode main fait appel aux méthodes suivantes : asList, println et computeTaxForProducts.
La méthode computeTaxForProducts fait appel à : accept, println et getCopmutedTax.

– Les méthodes en entrée :
La méthode println est appelée par les méthodes : main et computeTaxForProducts.
La méthode getComputedTax est appelée par la méthode computeTaxForProducts uniquement .
La méthode computeTaxForProducts est appelée par la méthode main .
La méthode accept est appelée par computeTaxForProducts;
La méthode asList est appelée par la méthode main.
Voici en image à quoi ressemble l'affichage dans me terminal.



Aussi nous fournissons l'image du graphe d'appel des méthodes dans toute l'application .


main
println
asList
product
setname
setprice
computeTaxForProducts
println
getcomputedtax
accept
visit
computedTax
getclass
println
getsimplename
getprice
print