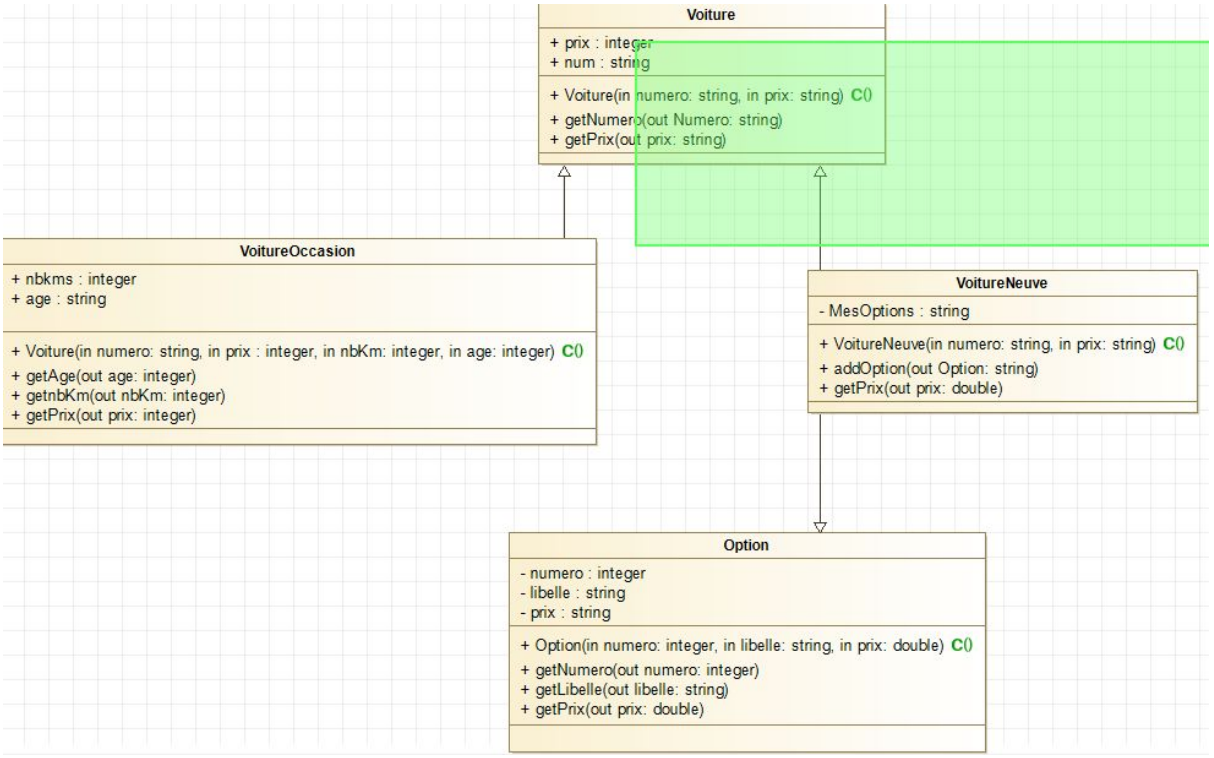
Compte rendu TP4 SLAM4

1 / Diagramme de classe de Voiture

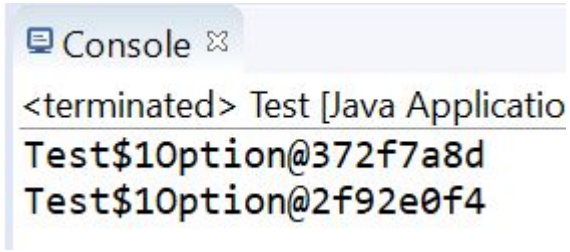


2/ Ici nous avons créer un main dans la classe **Test** et nous l'avons tester :

Test.java

```
1  public class Test {
2
3      public static void main(String[] args) {
4
5          class Option {
6          public Option(int i, String string, int j) {
7                  // TODO Auto-generated constructor stub
8              }
9          protected int numero;
10         protected String libelle;
11         protected double prix;
12         }
13
14         // TODO Auto-generated method stub
15         Option op1 = new Option(1,"Airbag",1000);
16          Option op2 = new Option(2,"Toit ouvrant",1500);
17          System.out.println(op1.toString());
18          System.out.println(op2.toString());
19
20
21     }
22     }
23
```

Résultat de l'exécution :



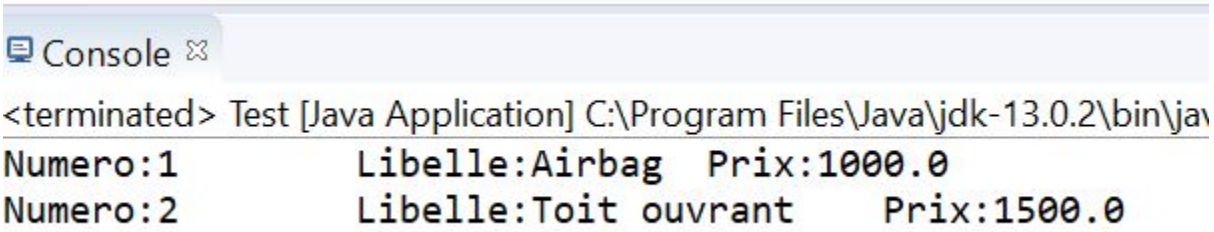
Il a exécuté le programme mais de façon désordonné.
On vient de faire des « Test First », c'est-à-dire d'écrire des programmes de test en premier. Ils testent des instructions qui n'ont pas encore été écrites !

3/ Une fois ceci fait, nous avons créer une nouvelle classe qui se nommera "**Option**" dont il sera composé de son constructeur et ses 3 accesseurs et sa méthode "**ToString**". Comme ci-dessous :

Test.java | Option.java

```
1  public class Option {
2      protected int numero;
3      protected String libelle;
4      protected double prix;
5
6      public Option(int numero, String libelle, double prix) {
7          this.numero = numero;
8          this.libelle = libelle;
9          this.prix = prix;
10     }
11
12     public int getNumero(int numero) {
13         return numero;
14     }
15
16     public String getLibelle(String libelle) {
17         return libelle;
18     }
19
20     public double getPrix(double prix) {
21         return prix;
22     }
23
24     public String toString(){
25         return "Numero:"+this.getNumero(numero) + "\tLibelle:"+this.getLibelle(libelle)+"\tPrix:"+ this.getPrix(prix);
```

Voici lors de l'exécution :



4/ Maintenant, nous avons implémenté la classe abstraite "**Voiture**" avec son constructeur et son accesseur "**getNumero**". Comme ci-dessous:

Test.java | Option.java | Voiture.java

```
1  public abstract class Voiture extends Option {
2          protected int numero;
3
4          public Voiture (int numero, String libelle, double prix) {
5              super(numero, libelle, prix);
6              this.numero=numero;
7              this.libelle=libelle;
8              this.prix=prix;
9          }
10
11         public int getNumero() {
12             return numero;
13         }
```

5/ Lorsque j'ai essayé de créer une voiture dans le main de la classe **Test**, une erreur s'est apparu. Comme ci-dessous:

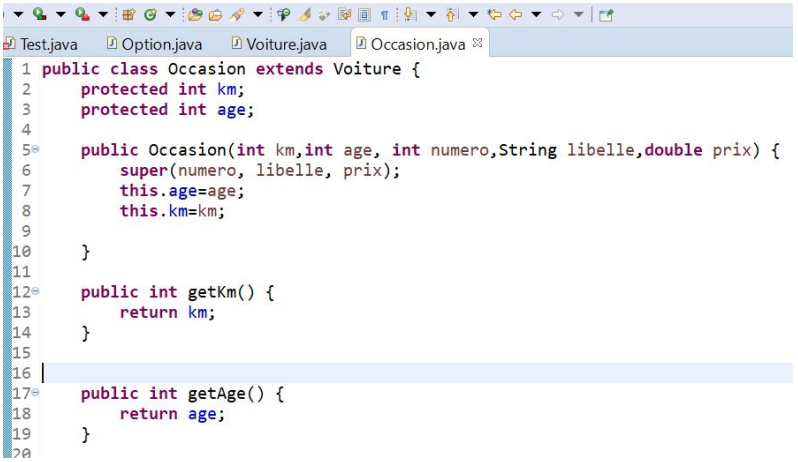
```
21          Voiture v1 = new Voiture(1);
```

L'erreur est qu'il ne peut pas initié le type "Voiture" à cause de la classe abstract.

6/ Nous allons ajouter une méthode "**getPrix**" qui retournera le prix. Comme ci-dessous :

```
public abstract double getPrix();
```

7/ Maintenant, nous allons créer une nouvelle classe "**Occasion**" qui héritera de la classe "**Voiture**" avec son constructeur et ces 2 accesseurs qui permettra de retourner le nombre de kilomètres et l'âge de la voiture en années. Comme ci-dessous :



8/ Maintenant, créons une voiture d'occasion dans la classe **Test** dans le main. Comme ci-dessous:

```
Occasion o1= new Occasion (33333,2016, 1, "Mercedes", 30000);
System.out.println(o1);
```

Lors de la création, le programme a été accepté.

9/ Maintenant, nous allons créer une méthode "getPrix" dans la classe "Occasion". Comme ci-dessous :

```
public double getPrix() {
    return prix;
}
```

10/Maintenant, nous allons créer une méthode "ToString" dans la classe "Voiture" pour qu'il retourne la plaque d'immatriculation. Comme ci-dessous :

```
@Override
public String toString() {
    return "Le numéro de la plaque d'immatriculation de la Mercedes AMG est :" +numero;
}
```

Résultat lors de l'éxécution:

```
Console
<terminated> Test [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe  (11 oct. 2020 à 21:25:51 – 21:25:52)
Le numéro de la plaque d'immatriculation de la Mercedes AMG est :975432834
```

11/ Maintenant, nous allons créer une méthode "ToString" dans la classe "Occasion" comme ci-dessous:

```
public String toString(){
     return super.toString()+"\t Nb de km: " + km + "\t anciennete:" +
    this.getAge() + "\t Prix de vente:" + this.getPrix()+"\n";
    }
```

12/ Maintenant, nous allons tester ce programme :

```
Occasion o2= new Occasion(50000,2, 98747794, "Mercedes", 7500);
System.out.println(o2.toString());
```

Résultat lors de l'exécution:

```
Console
<terminated> Test [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe  (11 oct. 2020 à 21:33:28 – 21:33:31)
Le numéro de la plaque d'immatriculation de la Mercedes AMG est :98747794       Nb de km: 50000        anciennete:2    Prix de vente:7000.0
```

13/ Maintenant, nous allons créer la classe "Neuve" qui héritera de la classe "Voiture" avec son constructeur et une méthode qui permettra d'ajouter des options comme ci-dessous :

Test.java | Option.java | Voiture.java | Occasion.java | Neuve.java

```
1  public class Neuve extends Voiture {
2      private String Option1;
3      private String Option2;
4
5
6
7      public Neuve(int numero, String libelle, double prix, String Option1, String Option2) {
8          super(numero, libelle, prix);
9          this.Option1=Option1;
10         this.Option2=Option2;
11
12     }
13
14
15     public double getPrix() {
16         return prix;
17     }
18
19
20     public String getOption1() {
21         return Option1;
22     }
23
24
25     public String getOption2() {
26         return Option2;
```

14/ Maintenant, nous allons créer une méthode "getPrix" qui retournera le prix de la voiture en tenant compte de toutes ses options.Comme ci-dessous :

```
public double getPrix() {
    return prix;
}
```

15/ Maintenant, nous allons créer une méthode "ToString" dans la classe "Neuve" selon le même principe que la méthode « toString » de la classe « Occasion ». Comme ci-dessous:

Méthode "ToString":

```
@Override
public String toString() {
    return "La plaque d'immatriculation de la AMG est " +numero+ " dont le prix avec les options seront de " + prix + " € dont " + Option1
}
```

Dans la classe Test:

```
Neuve n1= new Neuve (23457234,null,12000, "vitre teintés","Kit carroserie");
System.out.println(n1.toString());
```

Résultat du code:

```
Console
<terminated> Test [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe  (12 oct. 2020 à 21:40:51 – 21:40:51)
La plaque d'immatriculation de la AMG est 23457234 dont le prix avec les options seront de 12000.0 € dont vitre teintés et Kit carroserie
```

16/ Maintenant, nous allons créer la classe "**Garage**" comme ci-dessous :

```
public class Garage extends Neuve {

    public Garage(int numero, String libelle, double prix, String Option1,  String Option2) {
        super(numero,libelle,prix,Option1, Option2);
    }


    public double getPrix() {
        return prix;
    }


    public static void addNeuve(Neuve v1) {
        // TODO Auto-generated method stub

    }


    public double getNeuve( Neuve v1, double Neuve) {
        return Neuve;
    }


    public static void addOccasion(Occasion o9) {
        // TODO Auto-generated method stub

```

17/ Nous avons alimenté le garage en voitures et en options

```
Neuve v1=new Neuve(887655,null,10000, "vitre teintés","Kit carroserie");
Occasion o9= new Occasion(50000,2, 98747794, "Mercedes", 7500);
Occasion o10= new Occasion(100000,3, 76467943, "BMW", 9000);
Option op1 = new Option(1,"Airbag",800);
Option op2 = new Option(2,"Fermeture centralisée",900);
Option op3 = new Option(3,"Climatisation",1500);
Option op4 = new Option(4,"Toit ouvrant",700);
Option op5 = new Option(5,"Alarme",1000);


Garage.addNeuve(v1);
Garage.addOccasion(o9);
Garage.addOption(op3);
```

18

Je n'ai pas pu le faire.