**Compte Rendu**

Devoir POO en JAVA

**Réalisé par : EZZOUBAIR ZARQI**

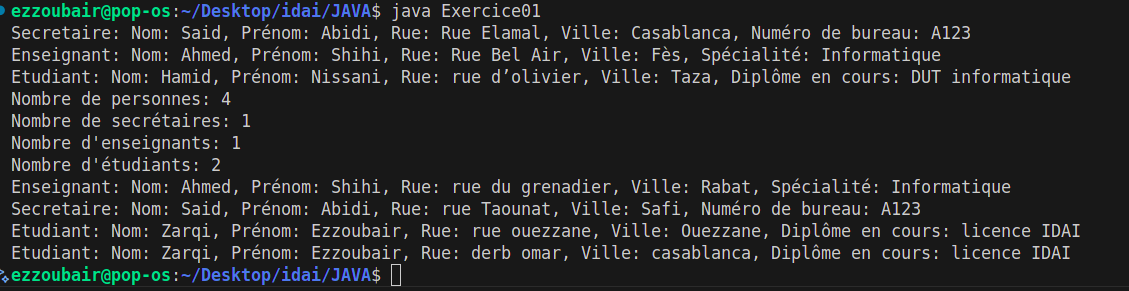
**Exercice 01 :**

Dans cet exercice, nous explorons les concepts fondamentaux de la **programmation orientée objet (POO)** en Java, en particulier l'utilisation des **classes abstraites** et de **l'héritage**. L'objectif est de modéliser une hiérarchie de classes représentant différentes catégories de personnes dans un établissement d'enseignement : les **secrétaires**, les **enseignants** et les **étudiants**.

À travers cet exercice, nous verrons comment :

* Utiliser une **classe abstraite** pour définir une structure commune.
* Implémenter des **méthodes abstraites** dans les classes dérivées.
* Gérer des **variables de classe** (statiques) pour compter le nombre d'instances créées.
* Manipuler des objets et tester leur comportement à l'aide d'une classe de test(**Exercice01**).

voilà le résultat d'exécution du class **Exercice01** :



**Exercice 02 :**

**Partie 01**

Dans cet exercice, nous explorons la **programmation orientée objet (POO)** en Java en nous concentrant sur les **classes abstraites** et l'**héritage** pour modéliser des objets graphiques. L'objectif est de créer une hiérarchie de classes représentant différents types d'objets graphiques, tels que des **rectangles**, des **boutons** et des **cercles**, tout en utilisant une classe abstraite commune, ObjetGraphique, pour encapsuler les propriétés et comportements partagés.

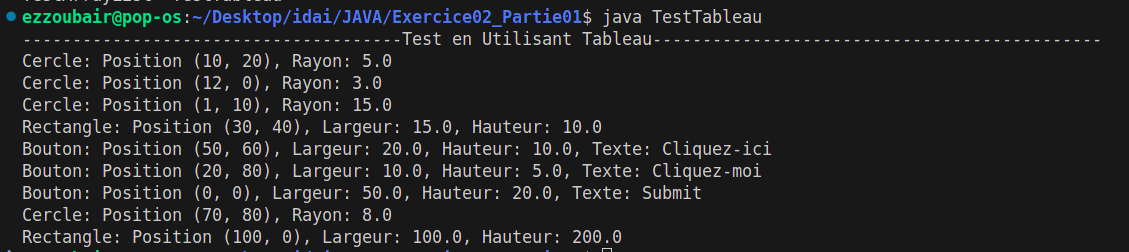
À travers cet exercice, nous verrons comment :

* Utiliser une **classe abstraite** pour définir une structure de base commune à tous les objets graphiques.
* Implémenter des **méthodes abstraites** dans les classes dérivées pour personnaliser le comportement de chaque type d'objet.
* Gérer des collections d'objets graphiques à l'aide de tableaux et de la classe ArrayList du package java.util.
* Parcourir une collection d'objets à l'aide d'un **itérateur** (ListIterator).

**les classe du Test :**

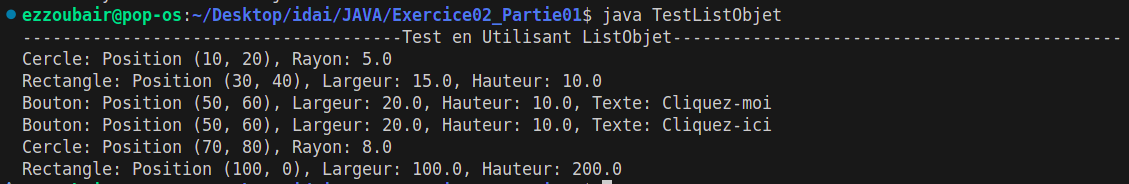
TestTableau.java : Classe de Test pour les Objets Graphiques en utilisant tableau .

Resultat :



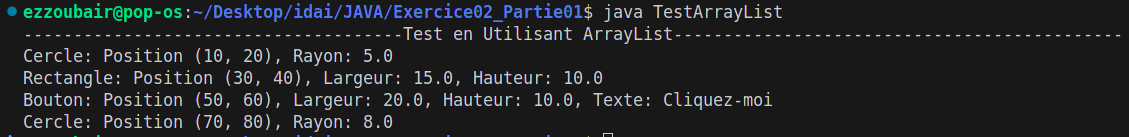
TestListObjet.java : Classe de Test pour les Objets Graphiques en utilisant le classe ListeObjetGraphique.

Resultat :



TestArrayList.java : Classe de Test pour les Objets Graphiques en utilisant le classe ArrayList

Resultat :



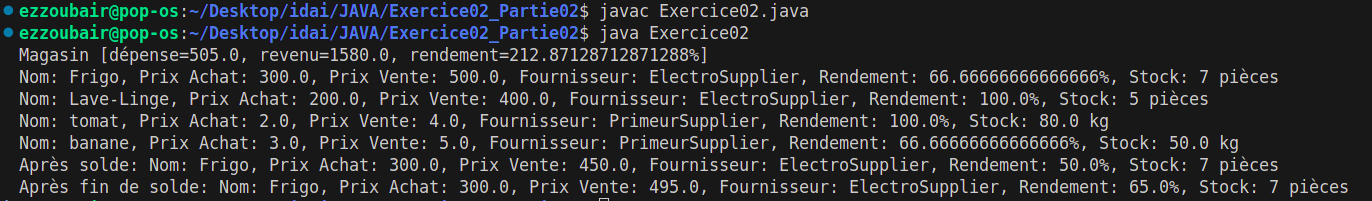
**Partie 02**

Dans cette partie de l'exercice, nous abordons la notion d'**interfaces** en Java pour modéliser un système de gestion de stock pour un magasin. L'objectif est de créer une application qui permet de gérer différents types de produits, tels que des **articles électroménagers** et des **articles primeurs**, en utilisant des interfaces pour définir des comportements spécifiques comme la vente au kilo, la vente à la pièce, et la gestion des soldes.

À travers cet exercice, nous verrons comment :

* Utiliser des **interfaces** pour définir des contrats que les classes doivent respecter, comme VendableKilo, VendablePiece, et Solde.
* Implémenter ces interfaces dans des classes concrètes pour gérer des produits spécifiques.
* Structurer une application en utilisant des classes de base (Article) et des classes dérivées (ArticleElectromenager, ArticlePrimeur) pour représenter différents types de produits.
* Gérer les stocks, les ventes, et les revenus dans une classe Magasin qui centralise les opérations.

Resultat d’execution du programme Exercice02Partie2.java :



**Réalisé par : EZZOUBAIR ZARQI**