Université d'Aix-Marseille III - Licence de Math-Info 2ème année

I4 : Programmation Objet - TD n^o3

Encapsulation

1. On reconsidère toutes les classes des TD 1 et 2. Tous les attributs sont maintenant privés.

– Pour chaque méthode, indiquer si elle doit être privée, privée-paquetage ou publique.

– Indiquer quels accesseurs ("`get`") et quels modifieurs ("`set`") sont utiles dans une utilisation normale de la classe.

2. On veut modifier la classe `ListeEntier` en lui ajoutant l'attribut (privé) `longueur` (de type `int`) qui permet à la méthode `int longueur()` de s'exécuter en temps constant en retournant simplement la valeur de cet attribut. Modifier les méthodes qui le nécessitent afin que cet attribut contienne toujours une valeur correcte après l'exécution de ces méthodes. (Vérifier que ces modifications sont transparentes pour un utilisateur qui ne connaît que l'existencede la liste des méthodes publiques)

3. On veut créer une classe `Complexe` pour représenter des nombres complexes. Un nombre complexe z a deux formes : $z = x+i.y$ ou $z = \rho.e^{i.\theta}$ donc on veut proposer les 8 méthodes suivantes : `double getX()`, `void setX(double)`, `double getY()`, `void setY(double)`, `double getRho()`, `void setRho(double)`, `double getTheta()`, `void setTheta(double)`.

Les relations entre ρ, θ, x et y sont : $x = \rho\cos\theta$, $y = \rho\sin\theta$, $\rho = \sqrt{x^2+y^2}$ et $\theta = \arcsin(\frac{y}{\sqrt{x^2+y^2}})$.

3.1 Dans un premier temps, on veut écrire des méthodes d'addition et de multiplication de nombres complexes qui fonctionnent quelle que soit la structure interne de la classe. On va donc considérer que les 8 accesseurs et modifieurs existent et on va les utiliser pour écrire les méthodes `void plus(Complexe)` et `void fois(Complexe)`. Ecrire ces méthodes en utilisant les formules $(x + iy) + (x' + iy') = (x + x') + i(y + y')$ et $(\rho e^{i\theta})(\rho' e^{i\theta'}) = (\rho\rho')e^{i(\theta+\theta')}$.

3.2 On va maintenant définir la structure interne de la classe `Complexe`. Ecrire les 8 accesseurs et modifieurs de la classe pour les 3 cas suivants :

– il n'existe que les attributs `x` et `y`.

– il n'existe que les attributs `rho` (ρ) et `theta` (θ).

– il existe les 4 attributs `x`, `y`, `rho` et `theta`. Dans ce cas, les accesseurs se contentent de retourner la valeur de leur attribut correspondant et les modifieurs modifient trois attributs.

3.3 Définir un constructeur qui prend en paramètre la partie réelle et la partie imaginaire d'un complexe. Définir deux méthodes d'affichage `afficheAlgeb` et `afficheTrigo` pour afficher un nombre complexe dans sa forme algébrique et dans sa forme trigonométrique.

Quel problème se pose si on veut définir un constructeur qui prend le module et l'argument d'un nombre complexe en paramètre ? Créer la méthode `instanceTrigo` qui crée et retourne une instance de `Complexe` en prenant le module et l'argument d'un nombre complexe en paramètre.

Tester la classe et toutes ses méthodes dans un programme.