Si vous ne les avez pas résolus dans la fiche numéro 3, faites les deux derniers exercices, ils vous sont donnés ici en rappel.

Exercice 1 (objets)

Créer une classe `vecteur` permettant de créer des vecteurs de $\mathbb{R}^3$ et de les manipuler comme par exemple obtenir le produit par un scalaire, le produit vectoriel, le produit scalaire, la norme, etc. Comme par exemple:

```
vecteur v1(1,1), v2(0,1);
v1.afficher();
v2.afficher();
double norme = v1.norme();
vecteur v3 = v1.produitVectoriel(v2.produit(5.2));
v3.afficher();
```

La méthode `afficher` permet d'obtenir l'affichage agréable du vecteur concerné, par exemple : `v1.afficher()` pourrait produire `(1,1)` directement sur l'écran.

Exercice 2 (objets)

Créer un classe permettant de représenter un QCM. Tout d'abord une classe `Item` contenant une question (texte), une suite de réponses (texte) et le numéro de la bonne réponse. Un QCM sera une liste d'`Item`s (le type `vector` de C++ pourra être utilisé).
On implémentera une fonction permettant de créer un QCM (avec des valeurs par défaut dans le code) et une autre pour tester le QCM auprès de l'utilisateur. Ensuite on réfléchira à la possibilité de lire un QCM depuis un fichier d'entrée.

Exercice 3 (composition)

Créer une classe `Point` représentant les points de l'espace à deux dimensions $\mathbb{R}^2$ et utiliser cette classe pour définir la classe des `SegmentDeDroite`. On devra pouvoir écrire un programme comme celui-ci :

```
Point p1(4,12.7);
Point p2(9,24.5);
std::cout << p1.toString() << std::endl; // affichage du point p1 sous une forme «agréable»
Segment s1(p1,p2);
Segment s2(4,5.5,6,7.7);
std::cout << s1.toString() << std::endl; // affichage du segment s1 sous une forme «agréable»
std::cout << s2.longueur() << std::endl; // affichage de la longueur de s2
```

La méthode `toString` permet d'obtenir, non pas l'affichage mais la représentation sous la forme d'une chaîne de caractères représentant le vecteur concerné, par exemple : `v1.toString()` pourrait produire la chaîne `(1,1)` est ensuite affiché à l'aide de `cout`.

Exercice 4

Créer une classe `Point` représentant les points de l'espace à deux dimensions $\mathbb{R}^2$ (on peut réutiliser la classe de l'exercice précédent) ainsi qu'une classe `Vecteur` de ce même espace. On implémentera les opérations permettant d'obtenir un vecteur à partir de deux vecteurs, un point à partir d'un point et d'un vecteur, comme par exemple:

```
Point o(1,1);
Vecteur v(5,5);
Point p = o.translater(v);
```