Question 2

a) L'intention du patron composite dans ce cas est de pouvoir utiliser les objets individuels et multiples de la même façon. Ceci veut dire que les objets de type TeamMember(objet individuel) et de type Team(objets multiples) sont utilisés de la même façon, car les classes dérivent de la première classe abstraite AbsTeamComponent qui contient juste des méthodes virtuelles pures.

Question 3

a) L'intention d'un patron de décorateur est de pouvoir rajouter, de manière dynamique, plus de fonctionnalités à un objet individuel sans changer le comportement des objets de la même classe. Dans ce cas, la classe TeamMemberRole permet d'ajouter des responsabilités à un objet de la classe TeamMember.

Question 4

Oui elle respecte les principes du modèle vue contrôleur. Les différentes classes du programme comblent tous les niveaux du MVC. Pour la partie vue, il y a la classe TeamComponentView qui est l'interface de l'application. C'est la classe qui gère les entrées et sorties du système et la seule qui n'est pas de lien direct avec le modèle. Pour le niveau controleur, il y a la classe AbsTeamComponent et celles qui dérivent de cette dernière qui font le lien entre la vue et le modèle. Finalement il y a les classes TeamManger et TeamComponentContainer qui representent le modèle, car elles contiennent toute la logique du système. TeamComponentContainer représente la base de données, la méthode de utilisée pour le stockage de données et TeamManager l'ensemble de méthodes utilisées pour manipuler ces données.