Programmation orientée objet (UAA14)

Exercices : série 3

Objets d'apprentissage :

- ✔ Modéliser une logique de programmation orientée objet.
- ✔ Déclarer une classe.
- ✔ Instancier une classe.
- ✔ Utiliser les méthodes de l'objet instancié.
- ✔ Traduire un algorithme dans un langage de programmation.
- ✔ Commenter des lignes de code.
- ✔ Tester le programme conçu.
- ✔ Caractériser les attributs dans une classe (encapsulation).
- ✔ Caractériser les méthodes dans une classe (encapsulation).
- ✔ Décrire la création d'un constructeur.
- ✔ Extraire d'un cahier des charges les informations nécessaires à la programmation.
- ✔ Programmer en recourant aux instructions et types de données nécessaires au développement d'une application.
- ✔ Corriger un programme défaillant.
- ✔ Développer une classe sur la base d'un cahier des charges en respectant le paradigme de la programmation orientée objet.
- ✔ Programmer en recourant aux classes nécessaires au développement d'une application orientée objet.
- ✔ Améliorer un programme pour répondre à un besoin défini.

Exercice 1 : tournoi

On te demande de créer un programme PHP sur base du domaine d'application suivant :

Dans un tournoi de football, plusieurs équipes se rencontrent. Une rencontre oppose deux équipes : une équipe « locale » et une équipe « visiteur ». Au fur et à mesure du déroulement de la rencontre, des points sont donnés à chacune des équipes par un arbitre. Cet arbitre peut également pénaliser une équipe en lui attribuant des fautes.

Étape 1

Crée la classe `Rencontre` qui contient les v.i suivantes :

- `$nomLocaux` : le nom de l'équipe locale
- `$nomVisiteurs` : le nom de l'équipe visiteur
- `$pointsLocaux` : le nombre de points de l'équipe locale
- `$pointsVisiteurs` : le nombre de points de l'équipe visiteur
- `$nbFautesLocaux` : le nombre de fautes de l'équipe locale
- `$nbFautesVisiteurs` : le nombre de fautes de l'équipe visiteur

Prévois un constructeur permettant de créer des objets de type `Rencontre`.

Ajoute les méthodes suivantes :

- `vainqueur` : retourne le nom de l'équipe gagnante ou « Egalité »
- `équipeFairPlay` : retourne le nom de l'équipe qui a fait le moins de fautes ou « Egalité »
- `égalité` : retourne un booléen avec `true` si les équipes ont toutes les deux le même nombre de points et `false` sinon
- `présenterLocaux` : retourne une chaine de caractère qui présente l'équipe locale sous la forme « l'équipe locale `<nom de l'équipe>` »
- `présenterVisiteurs` : retourne une chaine de caractère qui présente l'équipe visiteur sous la forme « l'équipe visiteur `<nom de l'équipe>` »
- `ajouterPointsLocaux` : ajoute 1 point à l'équipe locale
- `ajouterPointsVisiteurs` : ajoute 1 point à l'équipe visiteur
- `ajouterFauteLocaux` : ajoute 1 faute à l'équipe locale
- `ajouterFauteVisiteur` : ajoute 1 faute à l'équipe visiteur

Étape 2

En utilisant la classe `Tournoi`,on te demande :

- de créer au moins un objet de type `Rencontre`
- d'ajouter des points et des fautes aux deux équipes d'une même rencontre (utilise les méthodes adéquates !)
- d'afficher à l'écran : la présentation des deux équipes, le nom de l'équipe victorieuse et le nom de l'équipe la plus fairplay
- de tester (en faisant appel à la méthode adéquate) s'il s'agit d'une rencontre où les équipes sont ex aequo

Étape 3

Ajoute une méthode `présenterAdversaire` qui retourne une chaine de caractères qui présente les deux équipes de la manière suivante :

« L'équipe locale `<nom de l'équipe locale>` reçoit l'équipe des visiteurs `<nom de l'équipe visiteur>` ».

Attention: tu dois obligatoirement utiliser les méthodes déjà existantes `présenterLocaux` et `présenterVisiteurs`.

Étape 4

Ajoute une méthode appelée `ajouterPoints` qui ajoute 1 point à l'équipe locale ou visiteur en fonction d'un caractère reçu en argument : 'L' pour l'équipe locale et 'V' pour l'équipe visiteur.

Attention: tu dois obligatoirement utiliser les méthodes déjà existantes `ajouterPointsLocaux` et `ajouterPointsVisiteurs`.

Étape 5

Adapte le constructeur de la classe `Rencontre` de telle manière que l'on puisse créer facilement une rencontre où les deux équipes n'ont pas encore de point ni de faute (c'est-à-dire 0 partout).

Étape 6

Prévois, toujours dans la classe `Rencontre`, une méthode nommée `présenter` qui décrit une rencontre de la manière suivante :

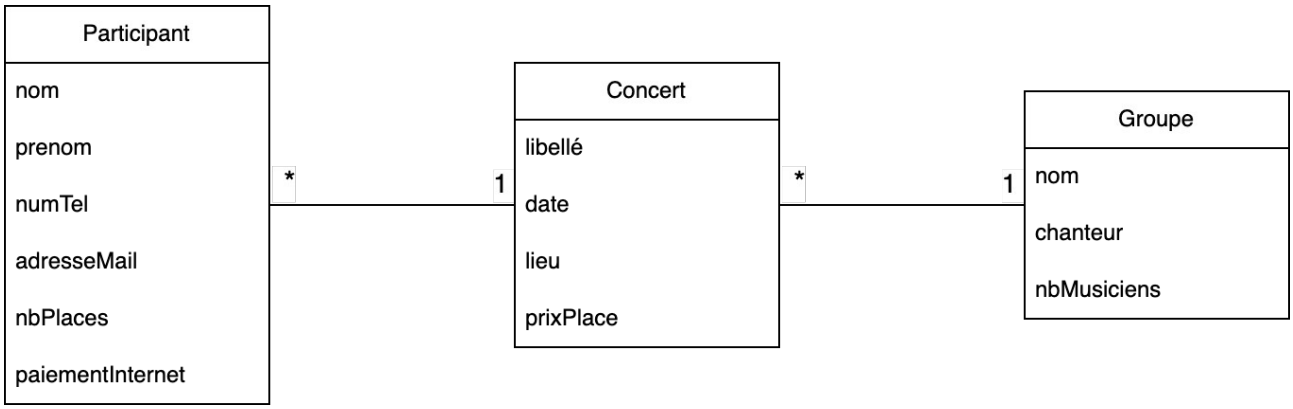
« L'équipe locale `<nom de l'équipe locale>` reçoit l'équipe visiteur `<nom de l'équipe visiteur>`. L'équipe victorieuse est `<nom de l'équipe gagnante>`.»

Étape 7

Adapte les méthodes `ajouterPointsLocaux` et `ajouterPointsVisiteurs` afin qu'elles reçoivent en argument un entier facultatif représentant le nombre de points à ajouter à l'équipe (donc 1 par défaut).

Exercice 2 : concert

L'objectif de cet exercice est de modéliser la situation suivante en PHP :



Étape 1

Crée une classe intitulée `Concert` permettant de créer des concerts.

Un concert est caractérisé par :

- un libellé
- une date
- un lieu
- un prix d'entrée

Étape 2

Crée un constructeur permettant de créer des concerts.

Crée ensuite au moins deux concerts.

Étape 4

Dans la classe `Concert`, crée une méthode `présenter` qui renvoie une chaine de caractère présentant le concert.

Affiche ensuite la présentation des concerts créés.

Étape 5

Affiche la date et l'endroit des concerts créés (ajoute les méthodes nécessaires dans la classe `Concert`).

Étape 6

Crée une classe intitulée `Participant` permettant de créer des personnes qui participent à un concert.

Un participant ne peut s'inscrire qu'à un seul concert. Bien évidemment, un concert peut accueillir plusieurs participants.

Un participant est caractérisé par :

- un nom
- un prénom
- un numéro de téléphone
- une adresse mail
- le nombre de places achetées
- le fait qu'il souhaite payer par internet ou non

Étape 7

Prévois un constructeur permettant de créer des participants.

Crée ensuite au moins trois participants.

Étape 8

Dans la classe `Participant`, prévois une méthode nommée `identité` retournant l'identité (nom / prénom) d'un participant.

Étape 9

Prévois une méthode nommée `présenter` qui retourne une chaîne de caractère décrivant un participant de la manière suivante :

« `<identité du participant>` participe au concert `<présentation du concert>`.»

Attention : pour implémenter cette méthode, tu dois utiliser la méthode `identité` ainsi que la méthode `présenter` de la classe `Concert` !

Affiche ensuite la présentation des participants créés.

Étape 10

Dans la classe `Participant`, prévois les méthodes permettant :

- de modifier (augmenter / diminuer) le nombre de places achetées sur base d'un entier reçu en argument
- de calculer le montant à payer

Ensuite, (en utilisant les méthodes adéquates !):

- augmente le nombre d'inscrits pour un des participants
- diminue le nombre d'inscrits pour un des autres participants
- affiche le montant total à payer pour chacun des participants
- affiche l'identité des participants

Étape 11

Affiche :

- le libellé du concert choisi par le premier participant créé
- la date du concert choisi par le deuxième participant créé
- le prix par personne du concert choisi par le troisième participant créé

Étape 12

Crée une classe nommée `Groupe` permettant de gérer des groupes musicaux.

Un groupe est caractérisé par :

- un nom
- le nom du chanteur
- le nombre de musiciens

Étape 13

Crée un constructeur pour la classe `Groupe` permettant des créer des groupes musicaux.

Crée quelques groupes musicaux de ton choix.

Étape 14

Prévois la méthode `présenter` qui retourne une chaine de caractère décrivant un groupe.

Affiche la présentation des groupes créés.

Étape 15

Modifie la classe `Concert` de telle manière que l'on puisse retrouver le groupe qui se produit lors du concert. N'oublie pas d'adapter le constructeur !

Adapte les instructions de création des objets de type `Concert`.

Étape 16

Adapte la méthode `présenter` de la classe `Concert` afin que la présentation reprenne la présentation du groupe qui se produit lors de ce concert.

Vérifie que la description des concerts est correcte (elle doit reprendre la description des groupes !).

Étape 17

Enfin :

- affiche le nom du groupe se produisant au concert choisi par le premier participant créé
- affiche le nombre de musiciens du groupe se produisant au concert choisi par le deuxième participant créé
- affiche le nom du chanteur se produisant au concert choisi par le troisième participant créé

Références

Les présents exercices ont été élaborés à l'aide des ressources suivantes :

- https://www.pierre-giraud.com/php-mysql-apprendre-coder-cours/introduction-programmation-orientee-objet/
- Cours de « Développement d'applications WEB », Hénallux (2019)