1 groupe 1

1.1 Qu'est-ce que la programmation orientée objet ?

La programmation orientée objet (POO), ou programmation par objet, est un paradigme de programmation informatique

1.2 A quoi sert la programmation orientée objet ?

la programmation orientée objet sert a créer une classe qui est un modèle qui va nous servir de construire des objets

1.3 POO est un nouveau langage de programmation ?

non

1.4 Différence entre la programmation orientée objet et la programmation procédurale

la programmation orientée objet et la programmation procédurale est que avec la programmation procédurale on code en block or la programmation orientée objet nous permet de créer classe pour bien structuré notre code

1.5 Qu'est-ce qu'une classe ? Objet ?

une classe est la représentation abstraite d'un objet.elle regroupe les methodes et les attributs
un objet est une instance de classe
Faire une équivalence entre variable et propriété en PHP.

1.6 Faire une équivalence entre fonction et méthode en PHP.

2 groupe 2

2.1 C'est quoi un constructeur et promotion de constructeur puis un destructeur ?

le constructeur nous permet de créer des objets le destructeur permet de détruire les objets

2.2 Faut-il implémenter explicitement les constructeurs ou les destructeurs ?

oui pour pouvoir créer des objets avec des propriétés qui ont des valeurs

2.3 3. A quoi sert le mot clé $this?

$this permet de faire reférence a l'objet courant

2.4 Y-a t-il un ramassage miette (Garbage collector comme en Java) en PHP .

2.5 C'est quoi une méthode en POO ?

un méthode poo est ensemble de code réutilisable

2.6 C'est quoi les setters et les getters ? Montrez comment elles sont nommées conventionnellement.

les **setters** nous permet de modifier les propriétés et les **getters** nous permet d'accéder aux propriétés d'une classe ils sont nommées conventionnellement setNomattribut ou getNomattribut

3 groupe 3

3.1 C'est quoi l'héritage en PHP dans POO ?

L'héritage permet à une classe de réutiliser le code d'une autre classe sans le dupliquer.

3.2 A quoi sert le mot clé extends ?

extends permet de définir qu'une classe hérite d'une autre classe

3.3 PHP supporte t-il l'héritage multiple ?

non

3.4 Comment appelle t-on la classe qui a hérité d'une autre classe ? Et celle dont a hérité ?

la classe qui a hérité d'une autre classe est appelé classe fille et celle dont a hérité elle appeler classe mère

3.5 Expliquer nous la notion de surcharge de méthodes

la surcharge de méthodes c'est le fait d'avoir plusieurs méthodes de même noms mais de signatures différentes dans une même classe

3.6 Expliquer nous la notion d'abstraction

L'abstraction c'est rendre une classe non instanciable

3.7 Quand est-ce qu'on utilise le modificateur de visibilité protected en PHP

on utilise le modificateur de visibilité protected en PHP quand on veut que les propriétés et les méthodes protégées soient visible dans la classe et ses classes filles

4 Groupe 4

5 Comment appliquer le modificateur de visibilité protected des propriétés et quand est-ce qu'il faut l'utiliser ?

le modificateur de visibilité protected des propriétés applique protected nomPropriete; elle est utiliser quand la classe mere ou les classe en ont besoin

6 C'est quoi le Polymorphisme ?

Le polymorphisme permet à des objets de différentes classes de répondre différemment en fonction du même message.

7 C'est quoi l'Interface ?

l'interface est une regroupement de constantes et des méthodes abstraite

8 groupe 5

8.1 Quel avantage offre le trait ?

le trait nous permet regrouper des méthodes de manière fine et cohérente

8.2 A quoi sert le mot clé use dans le cas du trait ?

le mot clé use nous permet d'utiliser un trait dans une classe

9 groupe 6

Les Méthodes statiques sont les methodes liés à la classe les propriétés sont les propriétés liés à la classe les propriétés statiques sont utiliser uniquement dans les Méthodes statiques la différence qui existe entre les propriétés de classe et les propriétés d'instance est que avec pour acceder les propriétés de classe on utlise la classe et pour acceder les propriétés d'instance on utlise l'objet

9.1 Donner une explication du mot clé static

le mot clé static permet de definir les methodes et propriétés statiques

9.2 Donner une explication du mot clé self et de l'opérateur ::

le mot clé self est utliliser pour accéder aux méthodes et propriétés statiques au sein de la classe. l'opérateur :: est utiliser pour invoker les methodes et les propriétés static

9.3 Comment utiliser les méthodes de classe à l'intérieur de la classe et à l'extérieur de la classe ?

pour utiliser les méthodes de classe à l'intérieur de la classe
self::nomMethode()

pour utiliser les méthodes de classe à l'extérieur de la classe
parent::nomMethode()

9.4 Donner l'importance des constantes

Les constantes nous permet de definir les variables qui ne vont evolués lors du programme

9.5 Parler nous de late static binding et comment l'utiliser efficacement.

late static binding appelée liaisons statiques tardives en francais est utiliser pour
invoker les methodes et propriétés static
pour l'utiliser on fait static::nomMethode s'il sagit d'une methode static
static::nomPropriete s'il sagit d'une propriété static

10 groupe 7

10.1 Faire des recherches sur Les constantes de classes

Les constantes de classes sont les constantes liées à la classe .Les classes filles peuvent modifier les constantes de cette classe s'il n'est pas finale

10.2 Savoir maîtriser Late Static Binding

late static binding, en français la résolution statique à la volée, qui peut être utilisée pour référencer la classe appelée dans un contexte d'héritage statique

10.3 Comprendre comment les méthodes magiques fonctionnent :

__**get ()** Cette méthode est automatiquement appelée lorsqu'on tente d'accéder à une propriété inaccessible d'un objet,elle prend en paramètre un le nom de la propriété inaccessible
__ **set()** Cette méthode est automatiquement appelée lorsqu'on tente de modifier à une propriété inaccessible d'un objet, elle prend en paramètre la propriété et la valeur qui l'on souhaite donné à la propriété et ne retourne rien
__ **call()** est une méthode qui est invoquée lorsque l'appel d'une méthode non déclarée ou inaccessible est effectué sur un objet
__ **callStatic()** Cette méthode est automatiquement appelée lorsqu'on tente d'accéder à une propriété inaccessible d'une classe,
__ **invoke()** permet à un objet d'etre invoker comme une fonction.exemple nomObjet(),
__ **toString ()** est une méthodes qui ne prend rien en paramètre et retourne une chaine de caractère .cette methode est appeler si l'on affiche un objet

11 GROUPE 8

Recherche :

11.1 Sérialisation des Objets avec la méthode serialize ( )

sérialiser un objet consiste a transformé cet objet en chaîne de caractère et lui stocké dans une variable

11.2 Désérialisation des objets avec la méthode unserialize ( )

11.3 clonage des objets

cloner un objet c'est faire le sauvegarde de cet objet dans une variable

11.4 Comparaison des objets

11.5 Les classes anonymes

Une classe anonyme est une classe sans nom déclaré.Pour obtenir le nom généré, vous pouvez utiliser la fonction get_class().

12 Groupe 9

12.1 Les namespaces

Les namespaces nous permet d'utiliser des classe de même nom dans le même projet en définissant chaque classe dans des namespaces différents.pour définir un namespace on fait
namespace nomDeNamespace;

12.2 Le mot clé use dans namespaces

Le mot clé use dans namespaces permet d'importer une classe à partir de son namespace

12.3 le mot clé alias

les alias permet de renommer les chemins vers les fichiers

13 groupe 10

13.1 Chargement automatique des classes à l'aide de la fonction spl_autoload_registrer( )

la fonction spl_autoload_registrer() Enregistre une fonction en tant qu'implémentation de __autoload() elle prend en paramètre un collbak le collbak prend en paramètre la classe et on définie le chemin vers la classe

13.2 Chargement automatique des classes à l'aide de la fonction _ _autoload( ) mais obsolète

la fonction _ _autoload( ) Tente de charger une classe indéfinie.elle prend en paramètre le nom de la classe et ne retourne rien

13.3 Chargement automatique des classes à l'aide du gestionnaire des paquets de PHP appelé composer

13.4 psr-4

Le PSR-4 définit un système d'autoloading (chargement automatique des classes) standard pour PHP, permettant aux développeurs de charger automatiquement les classes sans avoir à les inclure manuellement. dans le PSR-4 nous définition un tableau associatif qui a pour clé nos namespace et valeurs le nom de nos dossiers

14 groupe 11

14.1 Gestion des erreurs

14.2 set_error_handler()

14.3 set_exception_handler()

14.4 try ... catch ...finally

14.5 Throw

Le throw permet de propager l'erreur.

14.6 la classe Exception

15 groupe 12

15.1 class_exists()

class_exists() prend en paramètre le nom de la classe et vérifie si la classe existe

15.2 method_exists ()

method_exists ()Vérifie si la méthode de classe existe.elle prend en paramètre l'objet et la classe et la méthode et elle retourne un boolean

15.3 property_exists ()

property_exists () permet de vérifier si une propriété existe dans une classe.elle prend en paramètre l'objet de la classe et le nom de la propriété en chaîne et retourne true si la méthode existe et false si elle existe pas