Questions sur la Programmation Orientée Objet (1)

1. A quoi sert la surcharge de méthode dans Java ? Utiliser la même méthode ajout d'autres argument mais sans modifier sa logique

2. A quoi sert la redéfinition de méthode dans Java ? Le rédefinition d'une méthode sert à la modifier partiellement ou entièrement
En utilisant le keyword "override" de coup il retourne une résultat différent de la classe parent

3. A quoi sert le masquage de méthode dans Java ? Il sert à protéger les proprietés ou les methodes d'une classe point de vue accès
Que rédefinition car on peut pas les modifier de la subclasse

4. Quelle est la différence entre l'abstraction et Héritage dans la POO ? L'héritage permet de réutiliser la totalité d'une classe dans les class
Héritier alors que l'abstraction sert seulement à définir une classe abstraite qu'on peut pas initier mais on peut utiliser pour implementer des
Avec des fonctionnalit"és similaires

5. Quelle est la différence entre les patterns Decorator, Proxy et Adapter en Java ? Selon la définition proxy utilise le même interface de
L'objet alors que decorateur l'améliore et l'adapter change carrément l'interface de l'objet

6. Quelle est la différence entre une interface et une classe abstraite ? Dans la classe abstrait on peut définir tout types de méthodes même et on peut les détailler pour qu'il soit réutilisable sans modification. La classe abstraite permet d'heriter qu'une fois alors que l'interface par défaut déclare les propriétés et les méthodes en abstract et public mais on peut définir des methodes statiques aussi (java 8)

7. Comment savez-vous si un casting d'objet explicite est nécessaire ? Si le casting concerne le superclass vers le subclass

8. Quelle est la différence entre un constructeur et une méthode ? Un constructeur est une méthode qui sert à initialiser
Les variables d'une fonction mais on peut pas l'appeler alors que la méthode peut être appeler dans les objects et utiliser souvent
Les proprietés intialisé dans le constructeur

9. Pouvez-vous écrire une classe Java qui pourrait être utilisée à la fois comme une applet et comme une application ? Oui, en intégrant la
Méthode main et appelant les différents composants de l'application

10. Expliquer l'utilisation des packages dans Java ? Le package permet de relier une logique similaire d'une dossier et l'utiliser partout dans ce dossier ou d'un autre package(selon l'accessibilité défini)

11. Que feriez-vous pour comparer deux variables String – l'opérateur == ou la méthode equals () ? j'utilserai "==" car il compare la valeur
Pas la nature de l'objet

12. Une classe interne déclarée à l'intérieur d'une méthode peut-elle accéder aux variables locales de cette méthode ? Non, le classe devient Un block de code qui peut pas accédér aux arguments d'une méthode

13. Comment une sous-classe peut appeler une méthode ou un constructeur défini dans une superclasse ? Il doit utiliser le keyword "super"

14. Java est un langage purement orienté objet ? sinon pourquoi ? Oui java est en totalité orienté object , car pour l'utiliser on doit définir une classe avec une fonction main si on veut éxecuter une algorithme à l'intérieur

15. La différence entre surcharge et redéfinition d'une méthode ? Le fonctionnement d'une méthode dans le surcharge reste le même alors que dans La redefinition on est libre de changer la totalité de la méthode sauf son nom

16. Peut-on surcharger une méthode statique en Java ? Oui, on peut modifier les arguments d'une méthode

17. Peut-on redéfinir une méthode statique en Java ? Oui biensur, il nous permet de manipuler les propriétés et methodes librement à l'intérieur De la classe

18. Peut-on redéfinir une méthode privée en Java ? Non on ne peut pas redefinir une méthode privé

19. Quelle est la différence entre Composition et Héritage dans la POO ? Héritage permet d'hériter la totalité des propriétés et méthodes alors que La commposition permet d'utiliser un object comme un propriété à l'intérieur d'une classe

20. Pouvons-nous changer la liste d'arguments d'une méthode redéfinie ? Oui on peut changer les arguments que la méthode en elle-même

21. Pouvons-nous avoir une méthode non abstraite à l'intérieur d'une interface ? Non tout les methodes sont public et abstraite par défaut