Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

1. What does Aggregation mean in object-oriented programming?

    a. The part cannot exist without the whole.
    b. The class has a reference to another class, and the part can exist on its own.
    c. Aggregation is the same as Composition, where the whole controls the part's life.
    d. The whole class owns the part, and the part cannot exist on its own.

Ans: b. The class has a reference to another class, and the part can exist on its own.

Explanation: Aggregation means one class uses another class, and the used class can exist on its own.

2. Which of the following best describes Aggregation in Java?

    a. A strong "has-a" relationship where both entities must exist together.
    b. A special type of inheritance where one class is a subclass of another.
    c. A weak "has-a" relationship where one class is part of another, but both can exist independently.
    d. A relationship where both classes are tightly coupled and cannot exist without each other.

Ans: c. A weak "has-a" relationship where one class is part of another, but both can exist independently.

Explanation: Aggregation in Java means one class uses another, but both can exist independently. It's a weak "has-a" relationship without ownership.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

3. In an Aggregation relationship, what happens if the aggregated class (part) is deleted?

   a. The whole class (whole) will be deleted as well.
   b. The whole class will continue to exist independently of the part.
   c. The whole class becomes invalid without the part.
   d. The part will be destroyed, but the whole remains unaffected.

Ans: b. The whole class will continue to exist independently of the part.

Explanation: In Aggregation, the whole class can exist independently if the part is deleted, as the part is not owned by the whole.

4. Which of the following is a characteristic of Aggregation in Java?

   a. The dependent class cannot exist without the base class.
   b. It represents a "part-of" relationship where one class is part of the other.
   c. The "whole" and the "part" classes are tightly coupled, meaning they cannot be separated.
   d. Both classes in an aggregation relationship can exist independently of each other.

Ans: d. Both classes in an aggregation relationship can exist independently of each other.

Explanation: In Aggregation, both classes can exist independently, as it's a weak association where the "whole" doesn't own the "part."

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

5. In which of the following scenarios would Aggregation typically be used?

   a. A Car object and an Engine object, where the engine can exist independently of the car.
   b. A Student object and a School object, where the school cannot exist without the student.
   c. A Cat object and a PetOwner object, where the cat is always dependent on the pet owner.
   d. A Teacher object and a Classroom object, where the teacher must always be in the classroom.

Ans: a. A Car object and an Engine object, where the engine can exist independently of the car.

Explanation: Aggregation fits a. A Car object and an Engine object, as the engine can exist separately from the car, representing a weak "has-a" relationship.

6. How does Aggregation differ from Composition in Java?

   a. Aggregation is a stronger "has-a" relationship than Composition.
   b. Aggregation allows both objects to exist independently, whereas Composition implies that the lifetime of the part is tied to the lifetime of the whole.
   c. Aggregation requires that one object owns the other object completely, whereas Composition does not.
   d. There is no difference; both are exactly the same.

Ans: b. Aggregation allows both objects to exist independently, whereas Composition implies that the lifetime of the part is tied to the lifetime of the whole.

Explanation: Aggregation allows both objects to exist independently, while Composition means the part's lifecycle is fully dependent on the whole.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

7. What does "unidirectional association" mean in object-oriented programming?

    a. Both classes can reference each other in both directions.
    b. One class can reference the other, but not the other way around.
    c. Both classes can reference each other back and forth.
    d. Neither class can reference the other.

Ans: b. One class can reference the other, but not the other way around.

Explanation: Unidirectional association means one class can reference another, but the referenced class does not know about the referencing class.

8. In a unidirectional relationship where a Department has Students, what is true?

    a. Students can reference the Department, but the Department cannot reference Students.
    b. The Department can reference Students, but Students cannot reference the Department.
    c. Both the Department and Students can reference each other.
    d. Neither class can reference the other.

Ans: b. The Department can reference Students, but Students cannot reference the Department.

Explanation: In a unidirectional relationship, the Department can reference Students, but Students cannot reference the Department.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

9. What describes the relationship between Department and Students in a unidirectional association?

    a. The Department can access Students, but Students cannot access the Department.
    b. Students can access the Department, but only through a two-way relationship.
    c. Both the Department and Students can access each other.
    d. The Department and Students cannot access each other.

Ans: a. The Department can access Students, but Students cannot access the Department.

Explanation: In a unidirectional association, the Department can access Students, but Students cannot access the Department.

10. How does unidirectional association help with code reusability?

    a. It allows one class to be reused easily in different places while keeping things simple.
    b. It forces both classes to be used together, reducing reusability.
    c. It makes the code more complicated and harder to reuse.
    d. It prevents code reuse by not allowing the classes to interact.

Ans: a. It allows one class to be reused easily in different places while keeping things simple.

Explanation: Unidirectional association allows one class to be reused independently in different places, keeping the code simple and enhancing reusability.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

11. If the Department and Students relationship were bidirectional, what would happen?

a. The Department and Students would be more flexible and reusable.
b. The code would become more tightly connected, reducing reusability.
c. The Students class would not affect the Department class.
d. The code would become simpler and easier to maintain.

Ans: b. The code would become more tightly connected, reducing reusability.

Explanation: In a bidirectional relationship, the code becomes more tightly coupled, which can reduce flexibility and reusability due to the dependency between both classes.

12. Which of the following describes Aggregation in object-oriented programming?

a. It represents a part-of relationship and is a stronger form of association.
b. It represents a part-of relationship and is a weaker form of association.
c. It represents an is-a relationship and is a stronger form of association.
d. It represents an is-a relationship and is a weaker form of association.

Ans: b. It represents a part-of relationship and is a weaker form of association.

Explanation: Aggregation represents a part-of relationship and is a weaker form of association, as the part can exist independently of the whole.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

13. Which of the following best defines aggregation in Java?

a. A stronger relationship than composition, indicating total dependency.
b. A form of inheritance where one class inherits properties from multiple parent classes.
c. A 'has-a' relationship between two classes where one class contains a reference to another.
d. Strictly a 'uses-a' relationship indicating only method utilization and no property sharing.

Ans: c. A 'has-a' relationship between two classes where one class contains a reference to another.

Explanation: Aggregation in Java is a 'has-a' relationship where one class contains a reference to another, meaning the classes are related but can exist independently.

14. Why is aggregation considered a weak relationship?

a. Because the contained object can exist independently of the container.
b. It is not considered a weak relationship; this is a misconception.
c. Because it requires more memory than composition.
d. Due to the high coupling between the container and the contained object.

Ans: a. Because the contained object can exist independently of the container.

Explanation: Aggregation is a weak relationship because the contained object can exist on its own, independent of the container.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

15. When is it appropriate to use aggregation in Java?
    a. When there is a 'has-a' relationship and the lifetime of objects is independent.
    b. When multiple classes need to inherit properties from a single class.
    c. To enforce strict encapsulation and security of information.
    d. Whenever you need to create a deep relationship between classes.

Ans: a. When there is a 'has-a' relationship and the lifetime of objects is independent.

Explanation: Aggregation is used when there is a 'has-a' relationship and the lifetime of the objects is independent, meaning one object can exist without the other.

16. Which of the following is not a consequence of using aggregation in Java?
    a. Increased flexibility in object relationships.
    b. Low cohesion among classes.
    c. Simplification of complex problems by breaking them into smaller parts.
    d. Enhanced code reusability.

Ans: b. Low cohesion among classes.

Explanation: Aggregation increases flexibility, reusability, and simplicity, but it does not lead to low cohesion among classes.

17. In an aggregation relationship, which class is considered the container?
    a. Both classes are considered containers.
    b. Neither class is considered a container; it's a mutual relationship.
    c. The class that contains a reference to another class.
    d. The class that is being referenced.

Ans: c. The class that contains a reference to another class.

Explanation: In an aggregation relationship, the container is the class that contains a reference to another class.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

18. How can aggregation be implemented in Java?

a. Utilizing the 'implements' keyword for interface realization.
b. By declaring one class as 'final' and the other as 'static'.
c. Through the use of the 'extends' keyword.
d. By having one class include a reference variable of another class type.

Ans: d. By having one class include a reference variable of another class type.

Explanation: Aggregation in Java is implemented by having one class include a reference variable of another class type, indicating the "has-a" relationship.

19. What is the primary reason for using aggregation over inheritance in certain scenarios?

a. To reduce memory usage by shared object references.
b. Because Java does not support inheritance.
c. To enable multiple inheritance.
d. To avoid tight coupling and inheritance of unwanted properties.

Ans: d. To avoid tight coupling and inheritance of unwanted properties.

Explanation: The primary reason for using aggregation over inheritance is to avoid tight coupling and inheritance of unwanted properties, promoting more flexible and reusable code.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

20. Which of the following scenarios is best suited for using aggregation?

    a. A person and their heart, where the heart cannot function independently of the person.
    b. A graphical user interface (GUI) and its elements, where GUI elements are strictly tied to the presence of the GUI.
    c. A vehicle and its engine, where the engine cannot exist without the vehicle.
    d. A library system where a library 'has' books.

Ans: d. A library system where a library 'has' books.

Explanation: In a library system, a library "has" books, and books can exist independently, making it a suitable scenario for aggregation.

21. In Java, which statement about the access modifier in an aggregation relationship is true?

    a. The 'protected' modifier is strictly forbidden in aggregation relationships.
    b. The reference variable must always be declared with a 'private' access modifier.
    c. The choice of access modifier for the reference variable is flexible and depends on design needs.
    d. Public access modifier is not allowed for the reference variable in aggregation.

Ans: c. The choice of access modifier for the reference variable is flexible and depends on design needs.

Explanation: In aggregation, the choice of access modifier for the reference variable is flexible and depends on the design needs of the application.

Name of the Bundle	Proficient Bundle V2	Subject	Interview Skills in Programming V2
Topic	Aggregation	Last updated on	21 March 2025

22. Which relationship describes a "has-a" connection between two classes?

a. Aggregation
b. Inheritance
c. Association
d. Composition

Ans: a. Aggregation

Explanation: Aggregation represents a "has-a" relationship where one class contains an object of another class, but both can exist independently.