1. Which members of the Circle class are encapsulated?
   The private data members are encapsulated (for example, radius if it is declared private). Encapsulation means the data is hidden and accessed only through the class's public methods.
2. What name must the constructor of a class have?
   The constructor must have the same name as the class (and no return type).
3. Explain the difference between the private and public access modifiers.
   - private: accessible only inside the class where it is declared.
   - public: accessible from any other class.
     In short: private hides implementation details; public exposes the interface.

Consider the code:
Circle dot = new Circle(2);
dot.radius = 5;

4. Is the last statement valid or invalid? Explain.
   Invalid (in typical encapsulated design) if radius is private. You cannot directly assign to a private field from outside the class, you must use a public setter method (e.g. dot.setRadius(5)).
5. Using the Roo class shown, answer:
   (a) What is the name of the class?
   Roo
   (b) What is the name of the data member?
   x
   (c) List the accessor method.
   getX()
   (d) List the modifier method.
   setX(int z)
   (e) List the helper method.
   factor() (it's private and used inside the class)
   (f) What is the name of the constructor?
   Roo (the constructor has the same name as the class)
   (g) How many method members are there?
   There are 4 regular methods (setX, getX, calculate, factor) plus 1 constructor, so 4 methods + 1 constructor = 5 method-members total (if you count the constructor as a method-member).
6. What is the difference between a class and an object?
   A class is a blueprint or template (defines data members and methods). An object is an instance of that class , it has real state and can use the class's methods.

9. Given the data members in Moo:

- private double y;

- private static int x;
- private static final z; (a static final)
  (a) Which data member is a constant?
  The static final member (z) is the constant.
  (b) Which data members are variables?
  y and x are variables (changeable). x is static (class variable); y is instance variable.
  (c) Which data member(s) are instance members?
  y is an instance member.
  (d) Which data member(s) are class members?
  x and z are class members (they are static).

11. Compare and contrast overriding methods to overloading methods.

- Overriding
  - Happens when a subclass provides a new implementation for a method that has the same signature (name + parameters) as a method in its superclass.
  - Requires inheritance.
  - Enables runtime polymorphism, which method is decided at run time based on the actual object type.
  - Access rules: overridden method must have compatible visibility and return type (covariant returns are allowed).
- Overloading
  - Happens when two or more methods in the same class have the same name but different parameter lists(different number or types of parameters).
  - Does not require inheritance (though overloaded methods can be in subclasses too).
  - Resolved at compile time (which method to call is determined by the compile-time types and the argument list).
  - Used to provide multiple ways to call a method name with different inputs.