**Multiple Choice Questions (MCQs)**

1. What is the functional approach based on?
   * A. Objects
   * B. Classes
   * C. Functions
   * D. Data
2. Which of the following is a disadvantage of the functional approach?
   * A. Easy to reuse
   * B. Functions are separated from data
   * C. Suitable for complex systems
   * D. Easy to maintain
3. What does the object-oriented approach organize the solution around?
   * A. Functions
   * B. Data structures
   * C. Objects
   * D. Algorithms
4. Which of the following is **not** an object-oriented concept?
   * A. Inheritance
   * B. Polymorphism
   * C. Functions
   * D. Encapsulation
5. An object is a combination of:
   * A. Data and inheritance
   * B. Data and behavior
   * C. Behavior and class
   * D. Methods and messages
6. What defines the **state** of an object?
   * A. Methods
   * B. Attributes
   * C. Links
   * D. Messages
7. What describes an **object’s identity**?
   * A. Its attributes
   * B. Its memory address or uniqueness
   * C. Its name
   * D. Its class
8. What is an example of an object?
   * A. Student class
   * B. Integer
   * C. Micheal (a specific student)
   * D. A function
9. What is **encapsulation**?
   * A. Sharing functions globally
   * B. Hiding object internals from the outside
   * C. Making all attributes public
   * D. Copying methods from other classes
10. Which of the following **is an advantage** of encapsulation?
    * A. Global access to data
    * B. Increased complexity
    * C. Easier maintenance
    * D. Tighter coupling
11. What is a **class** in object-oriented development?
    * A. A data type
    * B. A description of object structure and behavior
    * C. A function definition
    * D. An instance of an object
12. What is the relationship between class and object?
    * A. A class is an instance of an object
    * B. A class contains multiple objects
    * C. An object is an instance of a class
    * D. Class and object are the same
13. Inheritance allows:
    * A. Object communication
    * B. Code rewriting
    * C. Method hiding
    * D. Reuse of attributes and methods
14. What is **generalization** in OOP?
    * A. Adding new features to a subclass
    * B. Creating subclasses from parent classes
    * C. Building a parent class from shared subclass features
    * D. Deleting redundant methods
15. What is **specialization**?
    * A. Sharing code
    * B. Hiding data
    * C. Building subclasses by adding unique features
    * D. Removing attributes
16. Which of the following is **not** true about inheritance?
    * A. Reduces development cost
    * B. Avoids code reuse
    * C. Helps class management
    * D. Supports polymorphism
17. Which of the following is an example of **single inheritance**?
    * A. Class A inherits from Class B and Class C
    * B. Class A inherits from only Class B
    * C. Class A has no parent
    * D. Class A is inherited by Class B and Class C
18. **Multiple inheritance** means:
    * A. A class inherits from multiple parent classes
    * B. A class is reused multiple times
    * C. A class contains multiple attributes
    * D. A class uses multiple interfaces only
19. What is polymorphism?
    * A. Using the same function everywhere
    * B. Defining different functions for the same method name in different classes
    * C. Making all attributes private
    * D. Making objects identical
20. Which of the following best describes **method overriding**?
    * A. Using method names with different parameters
    * B. Defining the same method in a subclass with a different implementation
    * C. Defining a method with the same body
    * D. Hiding a method
21. What does **dynamic linking** in polymorphism mean?
    * A. Linking all methods at compile time
    * B. Choosing the method to execute at runtime based on object class
    * C. Calling methods directly by name
    * D. Avoiding method inheritance
22. What does an abstract class contain?
    * A. Only private methods
    * B. No attributes
    * C. One or more abstract methods
    * D. Only static methods
23. Can an abstract class be instantiated?
    * A. Yes
    * B. No
    * C. Only in subclasses
    * D. Only if it has a constructor
24. What must subclasses of an abstract class do?
    * A. Implement all abstract methods
    * B. Create new constructors
    * C. Hide attributes
    * D. Reuse the parent’s name
25. What is the benefit of abstraction?
    * A. Complexity increase
    * B. Separation of implementation and definition
    * C. Making all data visible
    * D. Copy-paste code
26. Which type of message creates an object?
    * A. Destructor
    * B. Getter
    * C. Constructor
    * D. Setter
27. What message type is used to destroy an object?
    * A. Create
    * B. Destructor
    * C. Instantiator
    * D. Remover
28. Which of the following is **not** typically part of a message in OOP?
    * A. Constructor
    * B. Setter
    * C. Algorithm
    * D. Getter
29. What is one advantage of object-oriented development over the functional approach?
    * A. Easier to implement small systems
    * B. Suitable for simple data
    * C. Lower development cost for complex systems
    * D. Code tied closely to functions
30. In OOP, which of the following statements is true?
    * A. Object = data only
    * B. Class = specific instance
    * C. Object = algorithms + data structures
    * D. Class = object’s memory address
31. **C** – Functions
32. **B** – Functions are separated from data
33. **C** – Objects
34. **C** – Functions
35. **B** – Data and behavior
36. **B** – Attributes
37. **B** – Its memory address or uniqueness
38. **C** – Micheal (a specific student)
39. **B** – Hiding object internals from the outside
40. **C** – Easier maintenance
41. **B** – A description of object structure and behavior
42. **C** – An object is an instance of a class
43. **D** – Reuse of attributes and methods
44. **C** – Building a parent class from shared subclass features
45. **C** – Building subclasses by adding unique features
46. **B** – Avoids code reuse
47. **B** – Class A inherits from only Class B
48. **A** – A class inherits from multiple parent classes
49. **B** – Defining different functions for the same method name in different classes
50. **B** – Defining the same method in a subclass with a different implementation
51. **B** – Choosing the method to execute at runtime based on object class
52. **C** – One or more abstract methods
53. **B** – No
54. **A** – Implement all abstract methods
55. **B** – Separation of implementation and definition
56. **C** – Constructor
57. **B** – Destructor
58. **C** – Algorithm
59. **C** – Lower development cost for complex systems
60. **C** – Object = algorithms + data structures