SECTION 1: Object Oriented Concepts (1–15)

1. Which of the following is NOT a feature of OOP?
A) Encapsulation
B) Polymorphism
C) Compilation
D) Inheritance
Answer: C

---

2. An object is an instance of a:
A) Method
B) Class
C) Package
D) Interface
Answer: B

---

3. Encapsulation is achieved using:
A) Classes
B) Methods
C) Access Specifiers
D) Constructors
Answer: C

---

4. Which principle hides implementation details?
A) Abstraction
B) Inheritance
C) Polymorphism
D) Binding
Answer: A

---

5. Which concept allows one class to acquire properties of another?
A) Encapsulation
B) Abstraction
C) Inheritance
D) Overloading
Answer: C

---

6. Polymorphism means:
A) Many variables
B) Many classes
C) Many forms
D) Many objects
Answer: C

---

7. Method Overriding is an example of:
A) Compile-time polymorphism
B) Run-time polymorphism
C) Encapsulation
D) Abstraction
Answer: B

---

8. Binding of method call at runtime is called:
A) Static Binding
B) Dynamic Binding
C) Early Binding
D) Compile-time Binding
Answer: B

---

9. Message passing involves:
A) Communication between objects
B) Communication between variables
C) Communication between packages
D) Communication between threads
Answer: A

---

10. Which is NOT a pillar of OOP?
A) Abstraction
B) Encapsulation
C) Compilation
D) Polymorphism
Answer: C

---

11. Which keyword is used to inherit a class in Java?
A) implement
B) inherit
C) extends
D) include
Answer: C

---

12. Compile-time polymorphism is achieved by:
A) Overloading
B) Overriding
C) Inheritance
D) Abstract class
Answer: A

---

13. Overriding requires:
A) Same method name
B) Same parameters
C) Same return type
D) All of the above
Answer: D

---

14. A class is a:
A) Blueprint of object
B) Method
C) Variable
D) Package
Answer: A

---

15. Data hiding is related to:
A) Abstraction
B) Encapsulation
C) Inheritance
D) Binding
Answer: B

✓ SECTION 2: Introduction to Java (16–25)

16. Java was developed by:
A) Microsoft
B) Apple
C) Sun Microsystems
D) IBM
Answer: C

---

17. Java was initially called:
A) Oak
B) Pine
C) Cava
D) Brew
Answer: A

---

18. Java is a:
A) Procedural language
B) Object-oriented language
C) Machine language
D) Assembly language
Answer: B

---

19. Java programs are compiled into:
A) Machine code
B) Bytecode
C) Binary code
D) Native code
Answer: B

---

20. Bytecode runs on:
A) OS
B) Compiler
C) JVM
D) CPU
Answer: C

---

21. Java follows which principle?
A) Write Once Run Anywhere
B) Compile and Forget
C) One OS One Code
D) Native Only
Answer: A

---

22. Which is NOT a feature of Java?
A) Platform independent
B) Secure
C) Pointer support
D) Robust
Answer: C

---

23. Java supports:
A) Multithreading
B) Multiple inheritance via classes
C) Direct memory access
D) Operator overloading
Answer: A

---

24. Standalone applications are executed using:
A) Applet Viewer
B) Browser
C) JVM
D) Editor
Answer: C

---

25. Applets run inside:
A) JVM only
B) Browser
C) DOS
D) Compiler
Answer: B

✓ SECTION 3: JDK Tools (26–40)

26. Java compiler command is:
A) java
B) javac
C) javadoc
D) javap
Answer: B

---

27. Java interpreter command is:
A) javac
B) java
C) javadoc
D) javah
Answer: B

---

28. javap tool is used for:
A) Compilation
B) Execution
C) Disassemble bytecode
D) Documentation
Answer: C

---

29. javadoc tool is used for:
A) Documentation generation
B) Compilation
C) Execution
D) Debugging
Answer: A

---

30. javah tool is used for:
A) Native method header generation
B) Compilation
C) Execution
D) Editing
Answer: A

---

31. Which keyword is used to declare constant?
A) static
B) final
C) const
D) constant
Answer: B

---

32. Which is NOT a primitive data type?
A) int
B) float
C) String
D) char
Answer: C

---

33. Size of int in Java is:
A) 2 bytes
B) 4 bytes
C) 8 bytes
D) Depends on OS
Answer: B

---

34. Which loop executes at least once?
A) for
B) while
C) do-while
D) foreach
Answer: C

---

35. Array index in Java starts from:
A) 1
B) -1
C) 0
D) Depends
Answer: C

---

36. Which operator is used for comparison?
A) =
B) ==
C) :=
D) !=
Answer: B

---

37. Implicit type conversion is called:
A) Casting
B) Widening
C) Narrowing
D) Boxing
Answer: B

---

38. Explicit type conversion is called:
A) Widening
B) Promotion
C) Casting
D) Binding
Answer: C

---

39. Which is NOT a Java keyword?
A) static
B) public
C) include
D) private
Answer: C

---

40. Variable names in Java cannot start with:
A) Letter
B) _
C) $
D) Number
Answer: D

✓ SECTION 4: Classes and Objects (41–60)

41. Constructor name must be same as:
A) Method
B) Class
C) Variable
D) Package
Answer: B

---

42. Constructor is used to:
A) Destroy object
B) Initialize object
C) Execute program
D) Define class
Answer: B

---

43. Default constructor is provided by:
A) Programmer
B) Compiler
C) JVM
D) OS
Answer: B

---

44. Access specifier with maximum visibility:
A) private
B) protected
C) public
D) default
Answer: C

---

45. private members are accessible within:
A) Package
B) Subclass
C) Class only
D) Everywhere
Answer: C

---

46. main() method must be:
A) public
B) static
C) void
D) All of the above
Answer: D

---

47. Correct signature of main method is:
A) public void main()
B) public static void main(String args[])
C) static void main()
D) void main(String args)
Answer: B

---

48. Method overloading depends on:
A) Return type only
B) Number/type of parameters
C) Access specifier
D) Class name
Answer: B

---

49. this keyword refers to:
A) Parent class
B) Current object
C) Constructor
D) Static method
Answer: B

---

50. super keyword refers to:
A) Current class
B) Parent class
C) Object
D) Package
Answer: B

---

51. Which modifier prevents inheritance?
A) static
B) abstract
C) final
D) private
Answer: C

---

52. static members belong to:
A) Object
B) Class
C) Method
D) Constructor
Answer: B

---

53. Relationship between classes can be:
A) IS-A
B) HAS-A
C) Both A and B
D) None
Answer: C

---

54. HAS-A relationship represents:
A) Inheritance
B) Composition
C) Abstraction
D) Polymorphism
Answer: B

---

55. IS-A relationship represents:
A) Composition
B) Inheritance
C) Encapsulation
D) Binding
Answer: B

---

56. Which is used to restrict access within package?
A) private
B) protected
C) default
D) public
Answer: C

---

57. If no access specifier is used, it is:
A) public
B) private
C) default
D) protected
Answer: C

---

58. Objects are created using:
A) class
B) new keyword
C) extends
D) import
Answer: B

---

59. Which method is automatically called when object is created?
A) main()
B) finalize()
C) Constructor
D) display()
Answer: C

---

60. Which of the following is true about Java?
A) Supports multiple inheritance via classes
B) Uses pointers explicitly
C) Platform dependent
D) Bytecode based execution
Answer: D

SECTION 1: Inheritance and Interfaces (61–75)

61. Inheritance in Java is achieved using which keyword?
A) implement
B) extends
C) inherit
D) include
Answer: B

---

62. Single inheritance means:
A) One class inherits multiple classes
B) One class inherits one class
C) Multiple classes inherit one class
D) No inheritance
Answer: B

---

63. Multilevel inheritance involves:
A) One parent multiple children
B) Child → Parent → Grandparent
C) Multiple parents one child
D) No hierarchy
Answer: B

---

64. Java does NOT support multiple inheritance using:
A) Classes
B) Interfaces
C) Abstract classes
D) Packages
Answer: A

---

65. Multiple inheritance in Java can be achieved using:
A) extends
B) implements
C) super
D) final
Answer: B

---

66. Interface keyword in Java is:
A) interface
B) implements
C) extends
D) abstract
Answer: A

---

67. A class implements an interface using:
A) extends
B) implements
C) interface
D) inherit
Answer: B

---

68. All methods in an interface are by default:
A) private
B) protected
C) public and abstract
D) static
Answer: C

---

69. Interface supports:
A) Method body (normally)
B) Only abstract methods (traditional concept)
C) Constructors
D) Instance variables
Answer: B

---

70. Which inheritance is not supported directly in Java?
A) Single
B) Multilevel
C) Multiple (via classes)
D) Hierarchical
Answer: C

---

71. The super keyword is used to:
A) Call parent constructor
B) Create object
C) Access static method
D) Define interface
Answer: A

---

72. Which of the following is true about interface variables?
A) They are private
B) They are protected
C) They are public static final
D) They are instance variables
Answer: C

---

73. A class can implement:
A) Only one interface
B) Two interfaces only
C) Multiple interfaces
D) No interface
Answer: C

---

74. Hierarchical inheritance means:
A) One class inherits many classes
B) Many classes inherit one class
C) One class inherits itself
D) No inheritance
Answer: B

---

75. Which keyword prevents inheritance?
A) static
B) final
C) abstract
D) protected
Answer: B

✓ SECTION 2: Packages (76–90)

76. A package in Java is used to:
A) Store methods
B) Group related classes
C) Compile program
D) Execute program
Answer: B

---

77. The package keyword is written:
A) At end of program
B) After class
C) At beginning of program
D) Inside method
Answer: C

---

78. Which package is imported by default in Java?
A) java.util
B) java.io
C) java.lang
D) java.net
Answer: C

---

79. The java.lang package contains:
A) Collection classes
B) Utility classes
C) Fundamental classes like String
D) Database classes
Answer: C

---

80. java.util package contains:
A) Input output classes
B) Collection framework
C) Thread classes
D) Graphics classes
Answer: B

---

81. Which class belongs to java.util package?
A) String
B) Scanner
C) System
D) Math
Answer: B

---

82. Importing a package is done using:
A) include
B) import
C) using
D) package
Answer: B

---

83. Which symbol imports all classes of a package?
A) .
B) *
C) #
D) @
Answer: B

---

84. Fully qualified class name includes:
A) Class name only
B) Package name only
C) Package + Class name
D) Method name
Answer: C

---

85. Collection framework is available in:
A) java.io
B) java.lang
C) java.util
D) java.sql
Answer: C

---

86. Which of the following is a Collection class?
A) ArrayList
B) String
C) Thread
D) System
Answer: A

---

87. To create a user-defined package, we use:
A) import
B) package
C) extends
D) implements
Answer: B

---

88. Packages help in:
A) Name conflict resolution
B) Data hiding
C) Organization
D) All of the above
Answer: D

---

89. Which access modifier allows access within package only?
A) public
B) private
C) default
D) protected
Answer: C

---

90. Sub-packages are created using:
A) Dot (.) operator
B) Comma
C) Colon
D) Slash
Answer: A

✓ SECTION 3: Introduction to Threads (91–110)

91. A thread is:
A) Program
B) Lightweight process
C) Compiler
D) Package
Answer: B

---

92. Multithreading means:
A) One task
B) Multiple tasks simultaneously
C) No execution
D) Compilation
Answer: B

---

93. Thread class is available in:
A) java.lang
B) java.util
C) java.io
D) java.sql
Answer: A

---

94. Thread can be created by:
A) Extending Thread class
B) Implementing Runnable interface
C) Both A and B
D) None
Answer: C

---

95. Method used to start thread execution:
A) run()
B) execute()
C) start()
D) init()
Answer: C

---

96. Life cycle state before start():
A) Running
B) New
C) Dead
D) Waiting
Answer: B

---

97. Method to get current thread:
A) getThread()
B) currentThread()
C) thisThread()
D) activeThread()
Answer: B

---

98. Thread.sleep() causes:
A) Termination
B) Pause execution
C) Restart
D) Deadlock
Answer: B

---

99. Which is NOT thread state?
A) New
B) Runnable
C) Waiting
D) Compile
Answer: D

---

100. Synchronization is used to:
A) Speed up thread
B) Prevent race condition
C) Create thread
D) Stop thread
Answer: B

---

101. Race condition occurs when:
A) Multiple threads access shared data
B) Single thread runs
C) Compilation fails
D) No object exists
Answer: A

---

102. synchronized keyword is used for:
A) Package
B) Interface
C) Thread safety
D) Inheritance
Answer: C

---

103. Deadlock occurs when:
A) One thread waits
B) Threads wait for each other
C) Thread ends
D) Program compiles
Answer: B

---

104. join() method is used to:
A) Combine classes
B) Wait for thread to finish
C) Stop thread
D) Restart thread
Answer: B

---

105. stop() method is:
A) Recommended
B) Deprecated
C) Mandatory
D) Required
Answer: B

---

106. Yield() method:
A) Ends thread
B) Pauses and allows others to run
C) Compiles thread
D) Synchronizes thread
Answer: B

---

107. Daemon thread runs in:
A) Background
B) Foreground
C) Main class
D) Package
Answer: A

---

108. Which method is executed when thread starts?
A) start()
B) main()
C) run()
D) execute()
Answer: C

---

109. Runnable interface belongs to:
A) java.util
B) java.lang
C) java.io
D) java.sql
Answer: B

---

110. Multithreading improves:
A) Memory usage
B) CPU utilization
C) Compilation
D) Syntax
Answer: B

✓ SECTION 4: Exception Handling (111–125)

111. Exception is:
A) Compile error
B) Runtime error
C) Logical error
D) Syntax only
Answer: B

---

112. Exception handling keyword block starts with:
A) catch
B) try
C) throw
D) finally
Answer: B

---

113. Catch block handles:
A) Errors
B) Exceptions
C) Methods
D) Variables
Answer: B

---

114. Finally block executes:
A) Only if exception occurs
B) Only if no exception
C) Always
D) Never
Answer: C

---

115. Multiple catch blocks are used for:
A) Multiple methods
B) Multiple exceptions
C) Multiple classes
D) Multiple packages
Answer: B

---

116. throw keyword is used to:
A) Catch exception
B) Declare exception
C) Throw exception
D) End program
Answer: C

---

117. throws keyword is used in:
A) Catch block
B) Method declaration
C) Finally block
D) Package
Answer: B

---

118. ArithmeticException occurs when:
A) Divide by zero
B) Array out of bound
C) Null object
D) File not found
Answer: A

---

119. NullPointerException occurs when:
A) Object is null
B) Divide by zero
C) File missing
D) Syntax error
Answer: A

---

120. ArrayIndexOutOfBoundsException occurs when:
A) Wrong type
B) Invalid index
C) Null object
D) Compilation error
Answer: B

---

121. Checked exceptions are checked at:
A) Runtime
B) Compile time
C) Execution end
D) Package level
Answer: B

---

122. Unchecked exceptions occur at:
A) Compile time
B) Runtime
C) Linking time
D) Editing time
Answer: B

---

123. Error class belongs to:
A) java.lang
B) java.util
C) java.io
D) java.sql
Answer: A

---

124. Debugging means:
A) Writing code
B) Removing errors
C) Compiling
D) Packaging
Answer: B

---

125. Exception class is subclass of:
A) Error
B) Object
C) Throwable
D) Thread
Answer: C

SECTION 1: Object Oriented Concepts (1–15)

1. What is Object Oriented Programming (OOP)?
OOP is a programming approach based on objects and classes.

2. What is an object?
An object is an instance of a class.

3. What is a class?
A class is a blueprint or template for creating objects.

4. What is a method?
A method is a function defined inside a class.

5. What is abstraction?
Abstraction hides implementation details and shows only essential features.

6. What is encapsulation?
Encapsulation binds data and methods together into a single unit.

7. What is polymorphism?
Polymorphism means one name having multiple forms.

8. What is inheritance?
Inheritance is acquiring properties of one class into another class.

9. What is dynamic binding?
Dynamic binding means method call is resolved at runtime.

10. What is message passing?
Message passing is communication between objects through methods.

11. What are the four pillars of OOP?
Abstraction, Encapsulation, Inheritance, Polymorphism.

12. What is compile-time polymorphism?
Polymorphism achieved by method overloading.

13. What is runtime polymorphism?
Polymorphism achieved by method overriding.

14. What is data hiding?
Restricting access to data using access modifiers.

15. What is IS-A relationship?
IS-A relationship represents inheritance.

✓ SECTION 2: Introduction to Java (16–25)

16. Who developed Java?
Sun Microsystems.

17. What was Java originally called?
Oak.

18. In which year was Java released?
1995.

19. What is JVM?
Java Virtual Machine executes Java bytecode.

20. What is bytecode?
Intermediate code generated after compilation.

21. What does WORA stand for?
Write Once, Run Anywhere.

22. Name any two features of Java.
Platform independent, Secure.

23. What are types of Java programs?
Application programs and Applets.

24. What is an Applet?
A Java program that runs inside a web browser.

25. Why is Java platform independent?
Because it runs on JVM.

✓ SECTION 3: JDK Tools (26–45)

26. What is JDK?
Java Development Kit used for developing Java programs.

27. What is javac?
Java compiler command.

28. What is java command?
Used to run Java programs.

29. What is javap tool?
Used to disassemble bytecode.

30. What is javadoc tool?
Used to generate documentation.

31. What is javah tool?
Used to generate native method headers.

32. What is applet viewer?
Tool to run applets without browser.

33. What are Java keywords?
Reserved words used in Java.

34. How many primitive data types are there in Java?
Eight.

35. Name four primitive data types.
int, float, char, boolean.

36. What is a variable?
A named memory location.

37. What are literals?
Fixed constant values assigned to variables.

38. What is type conversion?
Converting one data type into another.

39. What is implicit type conversion?
Automatic type conversion by compiler.

40. What is explicit type conversion?
Manual type conversion using casting.

41. Name three looping constructs in Java.
for, while, do-while.

42. What is an array?
A collection of similar data elements.

43. What is the default value of int?
0.

44. What is an operator?
Symbol used to perform operations.

45. What is variable naming convention?
Rules for naming variables (start with letter, no spaces, etc.).

✓ SECTION 4: Classes and Objects (46–60)

46. How do you declare a class in Java?
Using the class keyword.

47. How is an object created?
Using the new keyword.

48. What is a constructor?
A special method used to initialize objects.

49. What is the name of constructor?
Same as class name.

50. What is default constructor?
Constructor provided by compiler if none is defined.

51. What are access specifiers?
Keywords that define accessibility (public, private, protected).

52. What is public access modifier?
Accessible from anywhere.

53. What is private access modifier?
Accessible within class only.

54. What is protected access modifier?
Accessible within package and subclasses.

55. What is main() method?
Entry point of Java program.

56. Write the syntax of main method.
public static void main(String[] args)

57. What is method overloading?
Same method name with different parameters.

58. What is HAS-A relationship?
Represents composition.

59. What is static modifier?
Belongs to class rather than object.

60. What is final modifier?
Prevents modification or inheritance.

SECTION 1: Inheritance and Interfaces (61–75)

61. What is inheritance in Java?
Inheritance is the process of acquiring properties of one class into another class.

62. Which keyword is used for inheritance?
extends

63. What is single inheritance?
When one class inherits from one parent class.

64. What is multilevel inheritance?
When a class inherits from a class which itself inherits another class.

65. What is hierarchical inheritance?
When multiple classes inherit from one parent class.

66. Does Java support multiple inheritance through classes?
No.

67. How can Java achieve multiple inheritance?
Through interfaces.

68. What is an interface?
An interface is a collection of abstract methods.

69. Which keyword is used to define an interface?
interface

70. Which keyword is used to implement an interface?
implements

71. Can a class implement multiple interfaces?
Yes.

72. What is the default access level of interface methods?
public and abstract.

73. Can interfaces have variables?
Yes, they are public static final.

74. What is the purpose of super keyword?
To access parent class members.

75. What is IS-A relationship?
It represents inheritance.

✓✓ SECTION 2: Packages (76–90)

76. What is a package in Java?
A package is a collection of related classes and interfaces.

77. Which keyword is used to create a package?
package

78. Where should package statement be written?
At the beginning of the program.

79. Which package is imported automatically?
java.lang

80. What is the use of import keyword?
To use classes from other packages.

81. What is fully qualified class name?
Package name followed by class name.

82. What is java.lang package?
Package containing fundamental classes like String and System.

83. What is java.util package?
Package containing utility classes like Scanner and Collection classes.

84. What is Collection Framework?
A set of classes and interfaces for storing and manipulating groups of data.

85. Name one collection class.
ArrayList.

86. What is the advantage of packages?
Avoids name conflicts and improves organization.

87. How do you create a user-defined package?
By using package keyword and compiling with proper directory structure.

88. Can packages contain sub-packages?
Yes.

89. Which symbol is used to import all classes of a package?
*

90. What is default access modifier visibility?
Within same package only.

✓ SECTION 3: Introduction to Threads (91–110)

91. What is a thread?
A lightweight process.

92. What is multithreading?
Executing multiple threads simultaneously.

93. What is single-threaded application?
Application that executes one task at a time.

94. Which class is used to create a thread?
Thread class.

95. Which interface can be implemented to create thread?
Runnable interface.

96. Which method starts a thread?
start()

97. Which method contains thread code?
run()

98. What is life cycle of thread?
New → Runnable → Running → Waiting → Terminated.

99. What is currentThread() method?
Method that returns reference of currently executing thread.

100. What is synchronization?
Controlling access to shared resources in multithreading.

101. What problem occurs without synchronization?
Race condition.

102. What is deadlock?
When two threads wait for each other indefinitely.

103. What is Thread.sleep()?
Method to pause thread execution.

104. What is join() method?
Waits for a thread to finish execution.

105. What is a daemon thread?
Background service thread.

106. Which keyword is used for synchronization?
synchronized

107. What is thread priority?
Value that influences thread scheduling.

108. What is Runnable state?
Thread ready to run.

109. What happens after run() method completes?
Thread enters terminated state.

110. What improves CPU utilization?
Multithreading.

✓ SECTION 4: Exception Handling (111–125)

111. What is an error in Java?
A serious problem that application cannot handle.

112. What is an exception?
An event that disrupts normal program flow.

113. What is checked exception?
Exception checked at compile time.

114. What is unchecked exception?
Exception checked at runtime.

115. Which block is used to handle exceptions?
try-catch block.

116. What is syntax of try block?
try { // code } catch(Exception e) { }

117. Can we use multiple catch blocks?
Yes.

118. What is finally block?
Block that executes always.

119. What is throw keyword?
Used to explicitly throw an exception.

120. What is throws keyword?
Used to declare exception in method signature.

121. What is ArithmeticException?
Exception for arithmetic errors like divide by zero.

122. What is NullPointerException?
Occurs when accessing null object.

123. What is ArrayIndexOutOfBoundsException?
Occurs when accessing invalid array index.

124. What is debugging?
Process of finding and fixing errors.

125. Which class is superclass of all exceptions?
Throwable.

SECTION 1: Object Oriented Concepts

---

1. Explain the four pillars of OOP with example.

Answer:

The four pillars of OOP are:

1. **Encapsulation** – Binding data and methods together.
2. **Abstraction** – Hiding implementation details.
3. **Inheritance** – Acquiring properties of parent class.
4. **Polymorphism** – One name with multiple forms.

Example:

```
class Animal {
    void sound() {
        System.out.println("Animal makes sound");
    }
}

class Dog extends Animal {
    void sound() {
        System.out.println("Dog barks");
    }
}

public class Test {
    public static void main(String[] args) {
        Animal obj = new Dog();  // Polymorphism
        obj.sound();             // Dynamic Binding
    }
}
```

---

2. Differentiate between Abstraction and Encapsulation.

Answer:

Abstraction	Encapsulation
Hides implementation	Hides data
Achieved using abstract class/interface	Achieved using access modifiers
Focus on behavior	Focus on security

---

3. Explain Dynamic Binding with example.

Answer:

Dynamic binding occurs when method call is resolved at runtime.

```
class A {
    void display() {
        System.out.println("Class A");
    }
}

class B extends A {
    void display() {
        System.out.println("Class B");
    }
}

public class Demo {
    public static void main(String[] args) {
        A obj = new B();
        obj.display();  // Runtime binding
    }
}
```

---

4. Explain Message Passing in Java.

Answer:

Message passing means calling methods through objects.

Example:

```
class Student {
    void show() {
        System.out.println("Hello");
    }
}
```

Here, calling `obj.show()` is message passing.

---

5. What is Polymorphism? Explain its types.

Answer:

Polymorphism allows one method name to have multiple behaviors.

Types:

- Compile-time (Overloading)
- Runtime (Overriding)

✓ SECTION 2: Introduction to Java

---

6. Explain history and features of Java.

Answer:

- Developed by **Sun Microsystems (1995)**
- Initially called Oak
- Created by James Gosling

Features:

- Platform independent
- Secure
- Robust
- Multithreaded
- Object-oriented

---

7. Explain JVM, JRE and JDK.

Answer:

- **JVM:** Executes bytecode
- **JRE:** JVM + libraries
- **JDK:** JRE + development tools

---

8. Explain types of Java programs.

Answer:

1. Application Program
2. Applet Program

✓ SECTION 3: JDK Tools

---

9. Explain Java compilation process.

Answer:

Step 1: Write code (.java)
Step 2: Compile using `javac` → generates `.class`
Step 3: Run using `java` command

---

10. Explain javap and javadoc tools.

Answer:

- `javap` → Disassembles bytecode
- `javadoc` → Generates documentation

---

11. Explain primitive data types in Java.

Answer:

There are 8 primitive data types:

byte, short, int, long, float, double, char, boolean

---

12. Write a program to demonstrate type casting.

```
public class TypeCast {
    public static void main(String[] args) {
        double d = 10.5;
        int i = (int)d;  // Explicit casting
        System.out.println(i);
    }
}
```

---

13. Write a program using loop to print numbers 1 to 10.

```
public class LoopExample {
    public static void main(String[] args) {
        for(int i=1; i<=10; i++) {
            System.out.println(i);
        }
    }
}
```

---

14. Write a program to find sum of array elements.

```
public class ArraySum {
    public static void main(String[] args) {
        int arr[] = {10,20,30};
        int sum = 0;
        for(int i=0;i<arr.length;i++) {
            sum += arr[i];
        }
        System.out.println("Sum = " + sum);
    }
}
```

---

15. Explain variable naming rules in Java.

Answer:

- Must start with letter, _ or $
- Cannot start with number
- No spaces allowed
- Cannot use keywords

✓ SECTION 4: Classes and Objects

---

16. Define class and object with example.

```
class Student {
    int roll;
    void show() {
        System.out.println(roll);
    }
}

public class Main {
    public static void main(String[] args) {
        Student s = new Student();
        s.roll = 10;
        s.show();
    }
}
```

---

17. Explain constructor with example.

```
class Demo {
    Demo() {
```

```
        System.out.println("Constructor called");
    }
}
```

---

18. Explain types of constructors.

- Default constructor
- Parameterized constructor

---

19. Write program to demonstrate method overloading.

```
class Add {
    int sum(int a, int b) {
        return a+b;
    }

    double sum(double a, double b) {
        return a+b;
    }
}
```

---

20. Explain access specifiers.

- public → accessible everywhere
- private → within class
- protected → package + subclass
- default → package only

---

21. Write program to demonstrate private access modifier.

```
class Test {
    private int x = 10;

    void show() {
        System.out.println(x);
    }
}
```

---

22. Explain static modifier with example.

```
class Demo {
    static int count = 0;
}
```

Static variable belongs to class.

---

23. Explain final keyword.

- final variable → constant
- final method → cannot override
- final class → cannot inherit

---

24. Write program to pass arguments to method.

```
class Demo {
    void display(int x) {
        System.out.println(x);
    }

    public static void main(String[] args) {
        Demo d = new Demo();
        d.display(50);
    }
}
```

---

25. Explain relationship between classes (IS-A and HAS-A).

- IS-A → Inheritance
- HAS-A → Composition

---

26. Write program demonstrating HAS-A relationship.

```
class Engine {
    void start() {
        System.out.println("Engine started");
    }
}

class Car {
    Engine e = new Engine();   // HAS-A
}
```

---

27. Explain main() method.

```
public static void main(String[] args)
```

- public → accessible
- static → no object needed
- void → no return
- String[] → arguments

---

28. Write program using parameterized constructor.

```
class Student {
    int roll;

    Student(int r) {
        roll = r;
    }
}
```

---

29. Explain method overriding with example.

```
class A {
    void show() {
        System.out.println("A");
    }
}

class B extends A {
    void show() {
        System.out.println("B");
    }
}
```

---

30. Write a complete program using OOP concepts.

```
class Person {
    private String name;

    Person(String name) {
        this.name = name;
    }

    void display() {
        System.out.println("Name: " + name);
    }
}
```

```
public class Test {
    public static void main(String[] args) {
        Person p = new Person("Rahul");
        p.display();
    }
}
```

Demonstrates:

- Encapsulation
- Constructor
- Object creation
- Method call

SECTION 1: Inheritance and Interfaces

---

1. Explain single inheritance with example.

Answer:

Single inheritance means one class inherits another class.

```
class Animal {
    void eat() {
        System.out.println("Eating...");
    }
}

class Dog extends Animal {
    void bark() {
        System.out.println("Barking...");
    }
}

public class Test {
    public static void main(String[] args) {
        Dog d = new Dog();
        d.eat();
        d.bark();
    }
}
```

---

2. Explain multilevel inheritance with example.

Answer:

Multilevel inheritance occurs when a class inherits from a class that itself inherits another class.

```
class A {
    void showA() { System.out.println("Class A"); }
}

class B extends A {
    void showB() { System.out.println("Class B"); }
}

class C extends B {
    void showC() { System.out.println("Class C"); }
}
```

---

3. What is hierarchical inheritance?

When multiple classes inherit from one parent class.

---

4. Why does Java not support multiple inheritance using classes?

To avoid ambiguity problem (Diamond Problem).

---

5. Explain interface with example.

Answer:

An interface is a collection of abstract methods.

```
interface Shape {
    void draw();
}

class Circle implements Shape {
    public void draw() {
        System.out.println("Drawing Circle");
    }
}
```

---

6. Difference between abstract class and interface.

Abstract Class	Interface
Can have methods with body	Traditionally only abstract methods
Uses extends	Uses implements
Can have constructors	Cannot have constructors

---

7. Write a program implementing multiple interfaces.

```
interface A {
    void show();
}

interface B {
    void display();
}

class Demo implements A, B {
    public void show() { System.out.println("Show"); }
    public void display() { System.out.println("Display"); }
}
```

---

8. Explain super keyword with example.

```
class Parent {
    Parent() {
        System.out.println("Parent Constructor");
    }
}

class Child extends Parent {
    Child() {
        super();
        System.out.println("Child Constructor");
    }
}
```

✓ SECTION 2: Packages

---

9. What is a package? Explain with example.

A package is a group of related classes.

```
package mypack;

public class Demo {
    public void show() {
        System.out.println("Package Example");
    }
}
```

---

10. How do you import a package?

```
import java.util.Scanner;
```

---

11. Explain java.lang package.

It contains fundamental classes like:

- String
- System
- Math
- Object

It is imported automatically.

---

12. Explain java.util package.

Contains utility classes like:

- Scanner
- ArrayList
- Date
- Collections

---

13. Write a program using ArrayList.

```
import java.util.ArrayList;

public class Demo {
    public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
        list.add("A");
        list.add("B");
        System.out.println(list);
```

```
    }
}
```

---

14. Explain fully qualified class name.

Example:

```
java.util.Scanner
```

---

15. Advantages of packages.

- Avoid name conflict
- Improve organization
- Provide access protection

✓ SECTION 3: Threads

---

16. What is a thread? Explain thread life cycle.

Thread is a lightweight process.

Life cycle:
New → Runnable → Running → Waiting → Terminated

---

17. Write program by extending Thread class.

```
class MyThread extends Thread {
    public void run() {
        System.out.println("Thread running");
    }

    public static void main(String[] args) {
        MyThread t = new MyThread();
        t.start();
    }
}
```

---

18. Write program using Runnable interface.

```
class MyThread implements Runnable {
    public void run() {
        System.out.println("Runnable Thread");
    }

    public static void main(String[] args) {
        Thread t = new Thread(new MyThread());
        t.start();
    }
}
```

---

19. Difference between single-threaded and multithreaded applications.

Single Thread	Multithread
One task at a time	Multiple tasks
Slow performance	Better CPU utilization

---

20. Explain synchronization with example.

Synchronization prevents race condition.

```
class Counter {
    int count = 0;

    synchronized void increment() {
        count++;
    }
}
```

---

21. What is deadlock?

Deadlock occurs when two threads wait for each other indefinitely.

---

22. Explain currentThread() method.

```
Thread t = Thread.currentThread();
System.out.println(t.getName());
```

---

23. What is Thread.sleep()?

Pauses execution for specified milliseconds.

✓ SECTION 4: Exception Handling

---

24. Explain exception handling syntax.

```
try {
    // risky code
} catch(Exception e) {
    System.out.println(e);
} finally {
    System.out.println("Always executes");
}
```

---

25. Explain checked and unchecked exceptions.

- Checked → Checked at compile time (IOException)
- Unchecked → Runtime (ArithmeticException)

---

26. Write program for divide by zero exception.

```
public class Demo {
    public static void main(String[] args) {
        try {
            int x = 10/0;
        } catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero");
        }
    }
}
```

---

27. Explain multiple catch blocks.

```
try {
    int a[] = new int[2];
    a[5] = 10;
} catch (ArithmeticException e) {
    System.out.println("Arithmetic Error");
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.println("Array Error");
}
```

---

28. Explain throw keyword with example.

```
throw new ArithmeticException("Custom Error");
```

---

29. Write user-defined exception example.

```
class MyException extends Exception {
    MyException(String msg) {
        super(msg);
    }
}
```

---

30. What is debugging? Explain its importance.

Debugging is the process of identifying and fixing errors in a program.

Importance:

- Improves program reliability
- Removes logical errors
- Ensures correct output

SECTION 1: Object Oriented Concepts

---

1. Explain Object Oriented Programming (OOP) and its advantages.

Answer:

OOP is a programming paradigm based on objects and classes.

Advantages:

- Code reusability
- Data security
- Modularity
- Easy maintenance

Example:

```
class Student {
    int roll;
    String name;

    void display() {
        System.out.println(roll + " " + name);
    }
}
```

---

2. Explain the four pillars of OOP with suitable examples.

Answer:

1. Encapsulation
2. Abstraction
3. Inheritance
4. Polymorphism

```
class Animal {
    void sound() { System.out.println("Animal sound"); }
}

class Dog extends Animal {
    void sound() { System.out.println("Dog barks"); }
}
```

Demonstrates inheritance and polymorphism.

---

3. Explain Abstraction using abstract class with program.

```
abstract class Shape {
    abstract void draw();
}

class Circle extends Shape {
    void draw() {
        System.out.println("Drawing Circle");
    }
}
```

Abstraction hides implementation details.

---

4. Explain Encapsulation with example.

```
class Person {
    private String name;

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

Data is hidden using private modifier.

---

5. Explain Polymorphism and its types with examples.

Compile-time → Overloading
Runtime → Overriding

```
class Add {
    int sum(int a, int b) { return a+b; }
    double sum(double a, double b) { return a+b; }
}
```

---

6. Explain Dynamic Binding with example.

```
class A {
    void show() { System.out.println("A"); }
}

class B extends A {
    void show() { System.out.println("B"); }
}

public class Test {
    public static void main(String[] args) {
        A obj = new B();
        obj.show();
    }
}
```

---

7. Explain Message Passing in Java.

Calling methods using object reference is message passing.

✓ SECTION 2: Introduction to Java

---

8. Explain history and features of Java.

- Developed by Sun Microsystems (1995)
- Created by James Gosling
- Initially called Oak

Features:

- Platform independent
- Secure
- Robust
- Multithreaded
- Object-oriented

---

9. Explain JVM, JRE and JDK with diagram explanation.

- JVM → Executes bytecode
- JRE → JVM + libraries
- JDK → JRE + development tools

---

10. Explain types of Java programs with example.

1. Application
2. Applet

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello Java");
    }
}
```

✓ SECTION 3: JDK Tools and Language Basics

---

11. Explain Java compilation and execution process.

Source Code → javac → Bytecode → JVM → Output

---

12. Explain important JDK tools.

- javac → compiler
- java → interpreter
- javap → disassembler
- javadoc → documentation
- javah → native header

---

13. Explain primitive data types with example.

```
int a = 10;
float b = 5.5f;
char c = 'A';
boolean flag = true;
```

---

14. Explain type conversion with program.

```
public class TypeCast {
    public static void main(String[] args) {
        double d = 10.9;
        int i = (int)d;
        System.out.println(i);
    }
}
```

---

15. Write program using loops to print factorial.

```
public class Factorial {
    public static void main(String[] args) {
        int n=5, fact=1;
        for(int i=1;i<=n;i++)
            fact *= i;
        System.out.println(fact);
    }
}
```

---

16. Explain arrays with program.

```
public class ArrayDemo {
    public static void main(String[] args) {
        int arr[] = {1,2,3};
        for(int i=0;i<arr.length;i++)
            System.out.println(arr[i]);
```

```
    }
}
```

✓ SECTION 4: Classes and Objects

---

17. Explain class and object with example.

```
class Car {
    String model;
    void display() {
        System.out.println(model);
    }
}
```

---

18. Explain constructor and its types.

Default and Parameterized.

```
class Demo {
    Demo() { System.out.println("Default"); }
    Demo(int x) { System.out.println(x); }
}
```

---

19. Explain method overloading with program.

```
class MathOp {
    int add(int a,int b){ return a+b; }
    double add(double a,double b){ return a+b; }
}
```

---

20. Explain access specifiers with example.

public, private, protected, default.

---

21. Explain static and final keywords.

- static → belongs to class
- final → constant or prevents inheritance

---

22. Explain main() method in detail.

```
public static void main(String[] args)
```

- public → accessible
- static → no object required
- void → no return
- String[] → arguments

---

23. Explain HAS-A relationship with program.

```
class Engine {
    void start(){ System.out.println("Engine Start"); }
}

class Car {
    Engine e = new Engine();
}
```

✓ SECTION 5: Inheritance and Interfaces

---

24. Explain types of inheritance with example.

Single, Multilevel, Hierarchical.

---

25. Write program demonstrating multilevel inheritance.

```
class A { void showA(){System.out.println("A");} }
class B extends A { void showB(){System.out.println("B");} }
class C extends B { void showC(){System.out.println("C");} }
```

---

26. Explain interface with program.

```
interface Shape {
    void draw();
}

class Circle implements Shape {
    public void draw(){
        System.out.println("Circle");
```

```
    }
}
```

✓ SECTION 6: Packages

---

27. Explain packages and advantages.

Used to group related classes.

Advantages:

- Organization
- Avoid name conflict
- Access control

---

28. Write program using java.util.ArrayList.

```
import java.util.ArrayList;

public class ListDemo {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>();
        list.add(10);
        list.add(20);
        System.out.println(list);
    }
}
```

✓ SECTION 7: Threads

---

29. Explain thread life cycle.

New → Runnable → Running → Waiting → Terminated.

---

30. Create thread using Thread class.

```
class MyThread extends Thread {
    public void run(){
        System.out.println("Thread Running");
```

```
    }
}
```

---

31. Explain synchronization with example.

```
class Counter {
    int count=0;
    synchronized void increment(){
        count++;
    }
}
```

✓ SECTION 8: Exception Handling

---

32. Explain exception handling mechanism with example.

```
try{
    int x = 10/0;
}
catch(ArithmeticException e){
    System.out.println("Error");
}
finally{
    System.out.println("Done");
}
```

---

33. Explain multiple catch blocks with example.

```
try{
    int arr[]=new int[2];
    arr[5]=10;
}
catch(ArithmeticException e){
    System.out.println("Arithmetic");
}
catch(ArrayIndexOutOfBoundsException e){
    System.out.println("Array Error");
}
```

---

34. Explain user-defined exception.

```
class MyException extends Exception {
    MyException(String msg){
        super(msg);
    }
}
```

35. Explain debugging and its importance.

Debugging is the process of detecting and correcting errors.

Importance:

- Ensures correct output
- Improves reliability
- Removes logical mistakes