Java OOPS - Code Snippets test – 1.1

March-24/ java/001.1 Time: 02:00hrs

Java OOPS - Code Snippets

Diploma in Advance Computing

March 2024

Q1. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q2. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q3. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q4. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q5. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q6. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q7. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q8. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q9. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q10. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q11. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q12. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q13. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q14. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q15. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q16. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q17. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q18. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q19. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q20. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q21. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q22. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q23. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q24. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q25. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q26. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q27. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q28. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q29. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q30. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q31. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q32. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q33. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q34. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q35. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q36. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q37. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q38. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q39. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q40. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q41. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q42. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q43. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q44. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q45. Which OOPS concept is demonstrated by the following code?

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

A. Inheritance

**B. Abstraction**

C. Encapsulation

D. Interface

Q46. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

Q47. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

A. 10

**B. 12**

C. 11

D. 13

Q48. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

A. Class A

**B. Class B**

C. Compilation Error

D. Runtime Error

Q49. Which OOPS concept is demonstrated by the following code?

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

**A. Inheritance**

B. Abstraction

C. Encapsulation

D. Interface

Q50. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

A. static

B. private

**C. final**

D. abstract

* Q51. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q52. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q53. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q54. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q55. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q56. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q57. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q58. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q59. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q60. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q61. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q62. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q63. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q64. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q65. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q66. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q67. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q68. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q69. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q70. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q71. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q72. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q73. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q74. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q75. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q76. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q77. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q78. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q79. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q80. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q81. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q82. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q83. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q84. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q85. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q86. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q87. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q88. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q89. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q90. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q91. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q92. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q93. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q94. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q95. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q96. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B

* Q97. Which OOPS concept is demonstrated by the following code?

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

* A. Inheritance
* B. Abstraction
* C. Encapsulation
* D. Interface

✅ Answer: A

* Q98. Which keyword is used to prevent method overriding?

class Parent {  
 final void show() {  
 System.out.println("Final Method");  
 }  
}

* A. static
* B. private
* C. final
* D. abstract

✅ Answer: C

* Q99. What will be the output?

class Test {  
 public static void main(String[] args) {  
 int x = 5;  
 int y = ++x \* 2;  
 System.out.println(y);  
 }  
}

* A. 10
* B. 12
* C. 11
* D. 13

✅ Answer: B

* Q100. What will be the output of the following code?

class A {  
 void show() { System.out.println("Class A"); }  
}  
class B extends A {  
 void show() { System.out.println("Class B"); }  
}  
public class Main {  
 public static void main(String[] args) {  
 A obj = new B();  
 obj.show();  
 }  
}

* A. Class A
* B. Class B
* C. Compilation Error
* D. Runtime Error

✅ Answer: B