```
1 class Animal {
2 void makeSound() {
            System.out.println("Some animal sound");
 4
5 }
 6
7-class Cat extends Animal {
 8
        @Overrid
        void makeSound() {
10
            System.out.println("Cat says: Bark (intentionally incorrect)");
11
12 }
13
14 public class Main {
        public static void main(String[] args) {
            Cat cat = new Cat();
cat.makeSound();
18
19 }
20
21
```

input

Cat says: Bark (intentionally incorrect)

...Program finished with exit code 0

Press ENTER to exit console.

```
1 class Vehicle {
2 -
       void drive() {
           System.out.println("Driving a vehicle");
5 }
6
7 class Car extends Vehicle {
8
       @Override
       void drive() {
9 -
           System.out.println("Repairing a car");
10
11
12 }
13
14 public class Main {
       public static void main(String[] args) {
16
           Car car = new Car();
17
           car.drive();
18
19
20
21
22
```

```
input

Repairing a car

...Program finished with exit code 0

Press ENTER to exit console.
```

```
1 class Shape {
   2 -
          double getArea() {
   3
              return 0;
   4
   5
     }
   6
   7 class Rectangle extends Shape {
          double length = \overline{5};
   8
          double width = 3;
   9
  10
          @Override
  11
  12 -
          double getArea() {
              return length * width;
  13
  14
  15 }
  16
  17 - public class Main {
  18 -
          public static void main(String[] args) {
  19
              Rectangle rect = new Rectangle();
              System.out.println("Area: " + rect.getArea());
  20
  21
  22 }
  23
  24
  25
  26
                                                                input
            O
Area: 15.0
...Program finished with exit code 0
Press ENTER to exit console.
```

```
1 class Employee {
   2 -
          void work() {
   3
              System.out.println("Employee working");
   4
   5
   6 -
          double getSalary() {
   7
              return 30000.00;
   8
   9
  10
  11 class HRManager extends Employee {
  12
          @Override
  13 -
          void work() {
              System.out.println("HR Manager managing employees");
  14
  15
  16
  17 -
          void addEmployee() {
              System.out.println("Adding a new employee");
  18
  19
  20
  21
  22 public class Main {
          public static void main(String[] args) {
  23 -
  24
              HRManager hr = new HRManager();
  25
              hr.work():
              hr.addEmployee();
  26
  27
              System.out.println("Salary: " + hr.getSalary());
                                                               input
        HR Manager managing employees
Adding a new employee
Salary: 30000.0
... Program finished with exit code 0
Press ENTER to exit console.
```

```
5
              balance += amount;
              System.out.println("Deposited: " + amount);
   6
          }
   8
   9 -
          void withdraw(double amount) {
  10
              balance -= amount;
              System.out.println("Withdrawn: " + amount);
  11
  12
     3
  13
  14
  15 class SavingsAccount extends BankAccount {
  16
          @Override
  17-
          void withdraw(double amount) {
  18 -
              if (balance - amount < 100) {
  19
                  System.out.println("Cannot withdraw. Minimum balance of 100 must be maintained.");
              } else {
  20 -
  21
                  super.withdraw(amount);
  22
  23
  24
     }
  25
  26 public class Main {
  27 -
          public static void main(String[] args) {
  28
              SavingsAccount sa = new SavingsAccount();
  29
              sa.withdraw(450);
  30
              sa.deposit(100);
  31
              sa.withdraw(400);
       *
                                                               input
               -
Deposited: 100.0
Withdrawn: 400.0
```

1 class BankAccount {

double balance = 500;

.. Program finished with exit code 0

Press ENTER to exit console.

void deposit(double amount) {

2

3 4 -

```
1 class Animal {
 2 -
       void move() {
            System.out.println("Animal moves");
 3
 4
 5
   }
7 class Cheetah extends Animal {
       @Override
 8
       void move() {
9 -
            System.out.println("Cheetah runs fast");
10
11
12 }
13
14 public class Main {
       public static void main(String[] args) {
15 -
16
            Cheetah ch = new Cheetah();
           ch.move();
17
18
19
20
21
```

```
Cheetah runs fast

...Program finished with exit code 0

Press ENTER to exit console.
```

```
1 class Person {
   2 -
          String getFirstName() {
   3
              return "John";
   4
   5
   6 -
          String getLastName() {
              return "Doe";
   7
   8
   9
     }
  10
  11 class Employee extends Person {
  12 -
          String getEmployeeId() {
  13
              return "EMP123";
  14
  15
  16
          @Override
  17 -
          String getLastName() {
              return "Doe - Software Developer";
  18
  19
  20
  21
  22 public class Main {
          public static void main(String[] args) {
  23 -
  24
              Employee emp = new Employee();
  25
              System.out.println(emp.getFirstName());
              System.out.println(emp.getLastName());
  26
              System.out.println(emp.getEmployeeId());
  27
  28
  29
                                                               input
        14
John
Doe - Software Developer
EMP123
...Program finished with exit code 0
Press ENTER to exit console.
```

```
1 class Shape {
 2 -
        double getPerimeter() {
 3
            return 0;
 4
 5
 6 -
        double getArea() {
 7
            return 0;
8
9
   }
10
11 class Circle extends Shape {
12
        double radius = 5;
13
14
       @Override
15 -
        double getPerimeter() {
            return 2 * Math.PI * radius;
16
17
18
19
       @Override
20 -
        double getArea() {
            return Math.PI * radius * radius;
21
22
23 }
24
25 public class Main {
26 -
        public static void main(String[] args) {
            Circle c = new Circle();
27
28
            System.out.println("Perimeter: " + c.getPerimeter());
29
            System.out.println("Area: " + c.getArea());
                                                            input
```

```
Perimeter: 31.41592653589793
Area: 78.53981633974483

...Program finished with exit code 0
Press ENTER to exit console.
```

```
3
          int year;
   4
   5 -
          Vehicle(String make, String model, int year, String fuelType) {
   6
              this.make = make;
              this.model = model;
   8
              this.year = year;
              this.fuelType = fuelType;
   9
  10
  11
  12 -
          double fuelEfficiency() {
  13
              return 0;
  14
  15
  16 -
          double maxSpeed() {
  17
              return 0;
  18
  19
  20 -
          double distanceTraveled(double fuelUsed) {
  21
              return fuelEfficiency() * fuelUsed;
  22
  23 }
  24
  25 class Car extends Vehicle {
  26 -
          Car() {
              super("Honda", "Civic", 2022, "Petrol");
  27
  28
  29
         @Override
v 🧷 🛅 🖎 🗴
                                                              input
Car max speed: 180.0
Truck distance on 10L: 80.0
Motorcycle fuel efficiency: 40.0
...Program finished with exit code 0
Press ENTER to exit console.
```

1 class Vehicle {

String make, model, fuelType;

2

```
Motorcycle() {
    super("Yamaha", "R15", 2021, "Petrol");
  58 -
  59
  60
  61
  62
          @Override
  63 -
          double fuelEfficiency() {
  64
               return 40;
  65
  66
  67
          @Override
          double maxSpeed() {
  68 -
  69
              return 150;
          }
  70
  71 }
  72
  73 public class Main {
  74 -
          public static void main(String[] args) {
               Car car = new Car();
  76
               System.out.println("Car max speed: " + car.maxSpeed());
  77
  78
               Truck truck = new Truck();
               System.out.println("Truck distance on 10L: " + truck.distanceTraveled(10));
  79
  80
               Motorcycle moto = new Motorcycle();
  81
               System.out.println("Motorcycle fuel efficiency: " + moto.fuelEfficiency());
  82
                                                                  input
Car max speed: 180.0
Truck distance on 10L: 80.0
Motorcycle fuel efficiency: 40.0
```

54

55 } 56 }

57 class Motorcycle extends Vehicle {

...Program finished with exit code O Press ENTER to exit console.

```
3
          double salary;
   4
   5 -
          Employee(String name, String address, String jobTitle, double salary) {
   6
              this.name = name;
   7
              this.address = address;
   8
              this.jobTitle = jobTitle;
   9
              this.salary = salary;
  10
  11
  12 -
          double calculateBonus() {
  13
              return salary * 0.10;
  14
  15
  16 -
          void performanceReport() {
              System.out.println(name + " has satisfactory performance.");
  17
  18
  19 }
  20
  21 class Manager extends Employee {
  22 -
          Manager() {
  23
              super("Alice", "Mumbai", "Manager", 80000);
  24
  25
          void manageProject() {
  26 -
  27
              System.out.println(name + " is managing a project.");
  28
  29 }
  30
 input
Bonus: 8000.0
Bob is writing Java code.
Charlie is fixing bugs.
...Program finished with exit code 0
Press ENTER to exit console.
```

class Employee {

String name, address, jobTitle;

2

```
30
31 class Developer extends Employee {
32 -
        Developer() {
            super("Bob", "Delhi", "Developer", 60000);
33
34
35
36 -
       void writeCode() {
           System.out.println(name + " is writing Java code.");
37
38
39
40
41 class Programmer extends Employee {
        Programmer() {
42 -
43
            super("Charlie", "Bangalore", "Programmer", 50000);
44
45
46 -
       void fixBugs() {
47
            System.out.println(name + " is fixing bugs.");
48
49 }
50
51 public class Main {
        public static void main(String[] args) {
52 -
53
            Manager mgr = new Manager();
54
            mgr.performanceReport();
55
            mgr.manageProject();
56
            System.out.println("Bonus: " + mgr.calculateBonus());
57
58
            Developer dev = new Developer();
59
            dev.writeCode();
```

```
Bonus: 8000.0

Bob is writing Java code.

Charlie is fixing bugs.

...Program finished with exit code 0

Press ENTER to exit console.
```

```
2 -
          public void run() {
   3 -
              for (int i = 1; i \le 5; i++) {
   4
                  System.out.println("Number: " + i);
   5
   6
   7
     7
   8
   9 public class Main {
          public static void main(String[] args) {
  10 -
  11
              MyThread t = new MyThread();
  12
              t.start();
  13
  14
  15
  16
  17
  18
  19
                                                               input
        D S
Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
... Program finished with exit code 0
Press ENTER to exit console.
```

1 class MyThread extends Thread {

```
1 class MyRunnable implements Runnable {
          public void run() {
   2 -
   3
              System.out.println("Hello from thread!");
   4
   5 }
   7 public class Main {
          public static void main(String[] args) {
   8 -
   9
              Thread t = new Thread(new MyRunnable());
  10
              t.start();
  11
  12
  13
  14
  15
  16
  17
  18
                                                              ing
Hello from thread!
...Program finished with exit code 0
Press ENTER to exit console.
```

```
1 class A extends Thread {
          public void run() {
   2 -
              System.out.println("Thread A is running");
   3
   4
   5 }
   6
   7 class B extends Thread {
          public void run() {
   8 -
              System.out.println("Thread B is running");
   9
  10
  11 }
  12
  13 - public class Main {
          public static void main(String[] args) {
  14 -
  15
              A t1 = new A();
  16
              B t2 = new B();
  17
              t1.start();
  18
              t2.start();
  19
            Ø.
                                                               inp
Thread A is running
Thread B is running
... Program finished with exit code 0
Press ENTER to exit console.
```

```
2 -
          public void run() {
   3 •
              try {
   4
                  System.out.println("Thread sleep for 2 seconds");
   5
                  Thread.sleep(2000);
   6
                  System.out.println("Thread running!");
   7 -
              } catch (InterruptedException e) {
   8
                  System.out.println("Thread interrupted");
   9
  10
  11
          public static void main(String[] args) {
  12 -
              Main t = new Main();
  13
  14
              t.start();
  15
  16
     }
  17
   / IP 🌣 👊
                                                              input
Thread sleep for 2 seconds
Thread running!
... Program finished with exit code 0
Press ENTER to exit console.
```

1 public class Main extends Thread {

```
1 public class Main {
        public static void main(String[] args) {
 2 -
 3 -
            try {
                int result = 10 / 0;
 4
            } catch (ArithmeticException e) {
 5 -
                System.out.println("Error: Division by zero!");
 6
7
 8
9
10
11
```

```
input

Error: Division by zero!

...Program finished with exit code 0

Press ENTER to exit console.
```

```
public class Main {
        static void checkAge(int age) throws Exception {
 2 -
 3
            if (age < 18)
                throw new Exception("Underage");
 4
 5
            else
                System.out.println("Allowed");
 6
 7
 8
        public static void main(String[] args) {
 9 -
10 -
            try {
                checkAge(16);
11
            } catch (Exception e) {
12 -
                System.out.println("Exception: " + e.getMessage());
13
14
15
16
17
```

```
Exception: Underage

...Program finished with exit code 0

Press ENTER to exit console.
```

```
1 public class Main {
          public static void main(String[] args) {
   2 -
   3 -
              try [
                  int data = 100 / 0;
   4
              } catch (ArithmeticException e) {
   5 -
                  System.out.println("Caught exception: " + e.getMessage());
   6
   7 -
              } finally {
                  System.out.println("code executed");
   8
   9
  10
  11 }
  12
  13
✓ ,* 
                                                              input
Caught exception: / by zero
code executed
...Program finished with exit code 0
Press ENTER to exit console.
```

```
1 enum Day {
          MONDAY, TUESDAY, WEDNESDAY
   2
   3
     }
   4
   5 public class Main {
          public static void main(String[] args) {
   6 -
   7
              Day today = Day.MONDAY;
              System.out.println("Today is: " + today);
   8
   9
  10
     }
  11
  12
  13
                                                              input
Today is: MONDAY
...Program finished with exit code 0
Press ENTER to exit console.
```

```
1 enum Color {
          RED, GREEN, BLUE
   2
   3 }
   4
   5 public class Main {
          public static void main(String[] args) {
   7
              Color c = Color.GREEN;
   8
   9 -
              switch (c) {
  10
                  case RED:
                      System.out.println("Red color");
  11
  12
                      break:
  13
                  case GREEN:
  14
                      System.out.println("Green color");
  15
                      break;
  16
                  case BLUE:
                      System.out.println("Blue color");
  17
  18
                      break;
 19
        input
           0
               1
Green color
```

```
Green color

...Program finished with exit code 0

Press ENTER to exit console.
```

```
1 enum Level {
 2
        LOW("Low Level"),
        MEDIUM("Medium Level"),
 3
        HIGH("High Level");
 4
 5
        private String description;
 6
 7
 8 -
        Level(String desc) {
 9
             this.description = desc;
10
11
12 -
        public String getDescription() {
13
             return description;
14
15 }
16
17 public class Main {
        public static void main(String[] args) {
             Level l = Level.HIGH;
19
             System.out.println("Level: " + 1);
System.out.println("Description: " + 1.getDescription());
20
21
22
23 }
```

```
Level: HIGH
Description: High Level

...Program finished with exit code 0
Press ENTER to exit console.
```