

```

1- class Animal {
2-     void makeSound() {
3-         System.out.println("Some animal sound");
4-     }
5- }
6-
7- class Cat extends Animal {
8-     @Override
9-     void makeSound() {
10-        System.out.println("Cat says: Bark (intentionally incorrect)");
11-    }
12- }
13-
14- public class Main {
15-     public static void main(String[] args) {
16-         Cat cat = new Cat();
17-         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() {
3         System.out.println("Driving a vehicle");
4     }
5 }
6
7 class Car extends Vehicle {
8     @Override
9     void drive() {
10        System.out.println("Repairing a car");
11    }
12 }
13
14 public class Main {
15     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 {
8     double length = 5;
9     double width = 3;
10
11     @Override
12     double getArea() {
13         return length * width;
14     }
15 }
16
17 public class Main {
18     public static void main(String[] args) {
19         Rectangle rect = new Rectangle();
20         System.out.println("Area: " + rect.getArea());
21     }
22 }
23
24
25
26

```



input

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() {
14         System.out.println("HR Manager managing employees");
15     }
16
17     void addEmployee() {
18         System.out.println("Adding a new employee");
19     }
20 }
21
22 public class Main {
23     public static void main(String[] args) {
24         HRManager hr = new HRManager();
25         hr.work();
26         hr.addEmployee();
27         System.out.println("Salary: " + hr.getSalary());
28     }

```



input

```

HR Manager managing employees
Adding a new employee
Salary: 30000.0

```

```

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

```

```

1 class BankAccount {
2     double balance = 500;
3
4     void deposit(double amount) {
5         balance += amount;
6         System.out.println("Deposited: " + amount);
7     }
8
9     void withdraw(double amount) {
10        balance -= amount;
11        System.out.println("Withdrawn: " + amount);
12    }
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.");
20         } else {
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

```

```

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

```

```

1 class Animal {
2     void move() {
3         System.out.println("Animal moves");
4     }
5 }
6
7 class Cheetah extends Animal {
8     @Override
9     void move() {
10        System.out.println("Cheetah runs fast");
11    }
12 }
13
14 public class Main {
15     public static void main(String[] args) {
16         Cheetah ch = new Cheetah();
17         ch.move();
18     }
19 }
20
21

```

input

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() {
7         return "Doe";
8     }
9 }
10
11 class Employee extends Person {
12     String getEmployeeId() {
13         return "EMP123";
14     }
15
16     @Override
17     String getLastName() {
18         return "Doe - Software Developer";
19     }
20 }
21
22 public class Main {
23     public static void main(String[] args) {
24         Employee emp = new Employee();
25         System.out.println(emp.getFirstName());
26         System.out.println(emp.getLastName());
27         System.out.println(emp.getEmployeeId());
28     }
29 }

```



input

```

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() {
16         return 2 * Math.PI * radius;
17     }
18
19     @Override
20     double getArea() {
21         return Math.PI * radius * radius;
22     }
23 }
24
25 public class Main {
26     public static void main(String[] args) {
27         Circle c = new Circle();
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.

```



```

1 class Vehicle {
2     String make, model, fuelType;
3     int year;
4
5     Vehicle(String make, String model, int year, String fuelType) {
6         this.make = make;
7         this.model = model;
8         this.year = year;
9         this.fuelType = fuelType;
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() {
27         super("Honda", "Civic", 2022, "Petrol");
28     }
29
30     @Override

```

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.

```

```

53     return 120;
54 }
55 }
56
57 class Motorcycle extends Vehicle {
58     Motorcycle() {
59         super("Yamaha", "R15", 2021, "Petrol");
60     }
61
62     @Override
63     double fuelEfficiency() {
64         return 40;
65     }
66
67     @Override
68     double maxSpeed() {
69         return 150;
70     }
71 }
72
73 public class Main {
74     public static void main(String[] args) {
75         Car car = new Car();
76         System.out.println("Car max speed: " + car.maxSpeed());
77
78         Truck truck = new Truck();
79         System.out.println("Truck distance on 10L: " + truck.distanceTraveled(10));
80
81         Motorcycle moto = new Motorcycle();
82         System.out.println("Motorcycle fuel efficiency: " + moto.fuelEfficiency());
83     }

```

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 Employee {
2     String name, address, jobTitle;
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() {
17        System.out.println(name + " has satisfactory performance.");
18    }
19 }
20
21 class Manager extends Employee {
22     Manager() {
23         super("Alice", "Mumbai", "Manager", 80000);
24     }
25
26     void manageProject() {
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.

```

30
31 class Developer extends Employee {
32     Developer() {
33         super("Bob", "Delhi", "Developer", 60000);
34     }
35
36     void writeCode() {
37         System.out.println(name + " is writing Java code.");
38     }
39 }
40
41 class Programmer extends Employee {
42     Programmer() {
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 {
52     public static void main(String[] args) {
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();

```



input

```

Bonus: 8000.0
Bob is writing Java code.
Charlie is fixing bugs.

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

```

```

1 class MyThread extends Thread {
2     public void run() {
3         for (int i = 1; i <= 5; i++) {
4             System.out.println("Number: " + i);
5         }
6     }
7 }
8
9 public class Main {
10     public static void main(String[] args) {
11         MyThread t = new MyThread();
12         t.start();
13     }
14 }
15
16
17
18
19

```



input

```

Number: 1
Number: 2
Number: 3
Number: 4
Number: 5

```

```

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

```



```
1 class MyRunnable implements Runnable {
2     public void run() {
3         System.out.println("Hello from thread!");
4     }
5 }
6
7 public class Main {
8     public static void main(String[] args) {
9         Thread t = new Thread(new MyRunnable());
10        t.start();
11    }
12 }
```



inp

Hello from thread!

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

```

1 class A extends Thread {
2     public void run() {
3         System.out.println("Thread A is running");
4     }
5 }
6
7 class B extends Thread {
8     public void run() {
9         System.out.println("Thread B is running");
10    }
11 }
12
13 public class Main {
14     public static void main(String[] args) {
15         A t1 = new A();
16         B t2 = new B();
17         t1.start();
18         t2.start();
19     }

```

Thread A is running
Thread B is running

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

```

1 public class Main extends Thread {
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
12    public static void main(String[] args) {
13        Main t = new Main();
14        t.start();
15    }
16 }
17

```

input

```

Thread sleep for 2 seconds
Thread running!

```

```

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

```

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



input

Error: Division by zero!

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

```

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

```

Exception: Underage

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

input


```

1 public class Main {
2     public static void main(String[] args) {
3         try {
4             int data = 100 / 0;
5         } catch (ArithmeticException e) {
6             System.out.println("Caught exception: " + e.getMessage());
7         } finally {
8             System.out.println("code executed");
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 {
2     MONDAY, TUESDAY, WEDNESDAY
3 }
4
5 public class Main {
6     public static void main(String[] args) {
7         Day today = Day.MONDAY;
8         System.out.println("Today is: " + today);
9     }
10 }
11
12
13
```



input

Today is: MONDAY

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

```

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

```



input

Green color

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

```

1 enum Level {
2     LOW("Low Level"),
3     MEDIUM("Medium Level"),
4     HIGH("High Level");
5
6     private String description;
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 {
18     public static void main(String[] args) {
19         Level l = Level.HIGH;
20         System.out.println("Level: " + l);
21         System.out.println("Description: " + l.getDescription());
22     }
23 }

```

Level: HIGH
Description: High Level

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