**Class Test**

**Scenario: Health Monitoring System Using OOP Principles**

You are tasked with designing a Health Monitoring System using Java and object-oriented programming principles. The system will monitor various health-tracking devices such as HeartRateMonitor, StepCounter, and SleepTracker. You will create a common interface to define shared behaviors, implement this interface across all device types, and use polymorphism to manage them uniformly.

**a. Construct the Interface**

Create a Java interface named HealthDevice with the following method declarations:

startMonitoring() – Begins data collection and returns a message (e.g., "Heart rate monitoring started").

stopMonitoring() – Ends data collection and returns a message.

getReading() – Returns a simulated reading value (e.g., “Heart Rate: 76 bpm”).

**b. Implement the Interface**

Implement the HealthDevice interface in the following classes:

HeartRateMonitor

Simulates monitoring heartbeats.

Returns heart rate in bpm.

StepCounter

Simulates counting daily steps.

Returns step count data.

SleepTracker

Simulates sleep tracking.

Returns sleep duration or quality**.**

Each class should provide custom logic and messages relevant to its functionality.

**c. Apply Polymorphism**

Create an ArrayList<HealthDevice> to hold instances of HeartRateMonitor, StepCounter, and SleepTracker.

Iterate through the list and call startMonitoring(), getReading(), and stopMonitoring() on each device.

Demonstrate how each device responds based on its own implementation while accessed through a common interface.

**d. Develop Extended Functionality**

Add a method deviceType() to the HealthDevice interface:

This method should return a string identifying the device type (e.g., "Step Counter").

Implement deviceType() in each class.

During iteration, print the output of deviceType() to display device identity polymorphically.