

Code:

import java.util.ArrayList;

import java.util.List;

// LabComputer class

class LabComputer {

private String compile;

private String os;

private String tools;

public LabComputer(String compile, String os, String tools) {

this.compile = compile;

this.os = os;

this.tools = tools;

}

}

// Student class

class Student {

private String name;

private String regNo;

private String discipline;

private int semester;

public Student(String name, String regNo, String discipline, int semester) {

this.name = name;

this.regNo = regNo;

this.discipline = discipline;

this.semester = semester;

}

}

// Iterator interface

interface Iterator {

boolean hasNext();

Object next();

}

// ConcreteIterator class

class ConcreteIterator implements Iterator {

private int index;

private List<LabComputer> computerList;

public ConcreteIterator(List<LabComputer> computers) {

this.index = 0;

this.computerList = computers;

}

@Override

public boolean hasNext() {

return index < computerList.size();

}

@Override

public Object next() {

if (hasNext()) {

return computerList.get(index++);

}

return null; // or throw NoSuchElementException

}

}

// AllocationStrategy interface

interface AllocationStrategy {

void allocateComputers(ComputerList computerList);

}

// LinuxAllocationStrategy class

class LinuxAllocationStrategy implements AllocationStrategy {

@Override

public void allocateComputers(ComputerList computerList) {

// Implementation for Linux allocation

System.out.println("Allocating computers for Linux OS to odd-numbered students.");

}

}

// WindowsAllocationStrategy class

class WindowsAllocationStrategy implements AllocationStrategy {

@Override

public void allocateComputers(ComputerList computerList) {

// Implementation for Windows allocation

System.out.println("Allocating computers for Windows OS to even-numbered students.");

}

}

// ComputerList class

class ComputerList {

private List<LabComputer> computers;

private Iterator iterator;

private AllocationStrategy strategy;

public ComputerList() {

this.computers = new ArrayList<>();

this.iterator = new ConcreteIterator(computers);

}

public void addComputer(LabComputer computer) {

computers.add(computer);

}

public List<LabComputer> getComputers() {

return computers;

}

public void setStrategy(AllocationStrategy strategy) {

this.strategy = strategy;

}

public void performStrategy() {

if (strategy != null) {

strategy.allocateComputers(this);

} else {

System.out.println("No strategy set.");

}

}

}

public class Main {

public static void main(String[] args) {

// Create some students and lab computers

Student student1 = new Student("John", "123", "Computer Science", 1);

Student student2 = new Student("Alice", "456", "Physics", 2);

LabComputer computer1 = new LabComputer("gcc", "Linux", "vim");

LabComputer computer2 = new LabComputer("visual studio", "Windows", "notepad");

// Create a computer list

ComputerList computerList = new ComputerList();

// Add computers to the list

computerList.addComputer(computer1);

computerList.addComputer(computer2);

// Set Linux allocation strategy

computerList.setStrategy(new LinuxAllocationStrategy());

// Perform the strategy

computerList.performStrategy();

// Set Windows allocation strategy

computerList.setStrategy(new WindowsAllocationStrategy());

// Perform the strategy

computerList.performStrategy();

}

}