- **AllocatedComputers Class:**

```
package LabMid;

import java.util.List;

class AllocatedComputers implements ComputerList {
   List<String> computerList;

   public AllocatedComputers(List<String> computerList) {
      this.computerList = computerList;
   }

   @Override
   public ComputerIterator createIterator() {
      return new ComputerListIterator(computerList);
   }
}
```

- **Computer Class**

```
package LabMid;

import java.util.List;

public class Computer {

   int computerNo;
   String OS;
   List<String> tools;

   public Computer(int computerNo, String OS, List<String> tools) {
```

```
        this.computerNo = computerNo;
        this.OS = OS;
        this.tools = tools;
    }

    @Override
    public String toString() {
        return "Computer{" +
                "computerNo=" + computerNo +
                ", OS='" + OS + '\'' +
                ", tools=" + tools +
                '}';
    }
}
```

- **ComputerAllocationTemplate**

```
package LabMid;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

abstract class ComputerAllocationTemplate {

    public void allocateComputers(List<Student> students, Map<String, Computer>
computers) {
        List<String> linuxList = new ArrayList<>();
        List<String> windowsList = new ArrayList<>();

        for (Student student : students) {
            Computer computer = allocateComputer(student, computers);
```

```
            if (computer.OS.equals("Linux")) {
                linuxList.add(student.name + " (" + student.registrationNo + "): " + computer);
            } else {
                windowsList.add(student.name + " (" + student.registrationNo + "): " + computer);
            }
        }

        printAllocationLists(linuxList, windowsList);
    }

    protected abstract Computer allocateComputer(Student student, Map<String, Computer> computers);

    protected abstract void printAllocationLists(List<String> linuxList, List<String> windowsList);
}
```

- **ComputerIterator**

```
package LabMid;

interface ComputerIterator {

    boolean hasNext();

    Object next();
}
```

```
package LabMid;

interface ComputerList {
```

```
    ComputerIterator createIterator();
}
```

- **ComputerListIterator**

```
package LabMid;

import java.util.List;

class ComputerListIterator implements ComputerIterator {
    List<String> computerList;
    int index = 0;

    public ComputerListIterator(List<String> computerList) {
        this.computerList = computerList;
    }

    @Override
    public boolean hasNext() {
        return index < computerList.size();
    }

    @Override
    public Object next() {
        if (this.hasNext()) {
            return computerList.get(index++);
        }
        return null;
    }
}
```

- **OddEvenComputerAllocation**

```
package LabMid;


import java.util.List;

import java.util.Map;


class OddEvenComputerAllocation extends ComputerAllocationTemplate {


   @Override

   protected Computer allocateComputer(Student student, Map<String, Computer>
computers) {

      int registrationNoLastDigit =
Integer.parseInt(String.valueOf(student.registrationNo.charAt(student.registrationNo.length(
) - 1)));

      return (registrationNoLastDigit % 2 == 1) ? computers.get("Linux") :
computers.get("Windows");

   }


   @Override

   protected void printAllocationLists(List<String> linuxList, List<String> windowsList) {

      System.out.println("Linux List:");

      for (String studentInfo : linuxList) {

         System.out.println(studentInfo);

      }


      System.out.println("\nWindows List:");

      for (String studentInfo : windowsList) {

         System.out.println(studentInfo);

      }

   }

}
```

- **Student**

```
package LabMid;

public class Student {

    String name;
    String registrationNo;
    String description;
    int semester;

    public Student(String name, String registrationNo, String description, int semester) {
        this.name = name;
        this.registrationNo = registrationNo;
        this.description = description;
        this.semester = semester;
    }
}
```

- **Demo**

```
package LabMid;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;

public class Demo {

    public static void main(String[] args) {
```

```
List<Student> students = Arrays.asList(
        new Student("Haleema", "005", "Computer Science", 3),
        new Student("Marwa", "006", "Electrical Engineering", 4),
        new Student("Fatima", "004", "English", 5),
        new Student("Maryum", "008", "Civil Engineering", 4),
        new Student("Ali", "003", "Computer Engineering", 2),
        new Student("Hamza", "002", "Electrical Engineering", 4),
        new Student("Umar", "001", "Artificial Intelligence", 7),
        new Student("Tooba", "007", "Computer Science", 4),
        new Student("Hafsa", "009", "Psychology", 7)
);


Map<String, Computer> computers = new HashMap<>();
computers.put("Linux", new Computer(1, "Linux", Arrays.asList("gcc", "vim", "python")));
computers.put("Windows", new Computer(2, "Windows", Arrays.asList("Visual Studio", "Notepad++")));
computers.put("Windows", new Computer(3, "Windows", Arrays.asList("Visual Studio", "Notepad++")));
computers.put("Linux", new Computer(4, "Linux", Arrays.asList("gcc", "vim", "python")));
computers.put("Linux", new Computer(5, "Linux", Arrays.asList("gcc", "vim", "python")));
computers.put("Linux", new Computer(6, "Linux", Arrays.asList("gcc", "vim", "python")));
computers.put("Windows", new Computer(7, "Windows", Arrays.asList("Visual Studio", "Notepad++")));
computers.put("Linux", new Computer(8, "Linux", Arrays.asList("gcc", "vim", "python")));
computers.put("Windows", new Computer(9, "Windows", Arrays.asList("Visual Studio", "Notepad++")));


ComputerAllocationTemplate allocationStrategy = new OddEvenComputerAllocation();
```

```
        allocationStrategy.allocateComputers(students, computers);


        Set<Integer> assignedComputerNumbers = new HashSet<>();
        List<String> linuxList = new ArrayList<>();
        List<String> windowsList = new ArrayList<>();


        for (Student student : students) {
            Computer computer = allocateComputer(student, computers,
assignedComputerNumbers);
            if (computer != null) {
                String allocationInfo = student.name + " (" + student.registrationNo + "): " +
computer;
                if (computer.OS.equals("Linux")) {
                    linuxList.add(allocationInfo);
                } else {
                    windowsList.add(allocationInfo);
                }
            }
        }
    }


    private static Computer allocateComputer(Student student, Map<String, Computer>
computers, Set<Integer> assignedComputerNumbers) {
        int registrationNoLastDigit =
Integer.parseInt(String.valueOf(student.registrationNo.charAt(student.registrationNo.length(
) - 1)));
        Computer computer = (registrationNoLastDigit % 2 == 1) ? computers.get("Linux") :
computers.get("Windows");
        if(computer != null && assignedComputerNumbers.add(computer.computerNo)){
            return computer;
        } else {
            return null;
```

```
        }
    }

    private static void printAllocationList(String header, List<String> list) {
        System.out.println(header);
        ComputerListIterator iterator = new ComputerListIterator(list);
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }
        System.out.println();
    }
}
```