Question 01.

```
package Question01;


public class List {
    private Student[] data;
    private int size;
    private int capacity;

    public List(int capacity) {
        this.capacity = capacity;
        this.data = new Student[capacity];
        this.size = 0;
    }

    public void add(Student student) {
        if (size < capacity) {
            data[size++] = student;
        } else {
            System.out.println("List is full. Cannot add student.");
        }
    }

    public void display() {
        if (size == 0) {
            System.out.println("List is empty.");
            return;
        }
        System.out.printf("%-15s %-15s %-8s %-5s\n", "Student Number", "Name",
"Gender", "Grade");
        System.out.println("-----------------------------------------------
");
        for (int i = 0; i < size; i++) {
```

```
            System.out.printf("%-15s %-15s %-8c %-5s\n",
                    data[i].getStudentNumber(), data[i].getName(),
                    data[i].getGender(), data[i].getGrade());
        }
    }

    public boolean isEmpty() {
        return size == 0;
    }

    public int listSize() {
        return size;
    }

    public Student retrieveList(int index) {
        if (index >= 0 && index < size) {
            return data[index];
        }
        return null;
    }

    public void sortByGrade() {
        for (int i = 0; i < size - 1; i++) {
            int minIndex = i;
            for (int j = i + 1; j < size; j++) {
                if (data[j].getGrade().compareTo(data[minIndex].getGrade()) <
0) {
                    minIndex = j;
                }
            }
            Student temp = data[minIndex];
            data[minIndex] = data[i];
```

```
            data[i] = temp;
        }
    }

    public List findStudentsByGradeBinary(String targetGrade) {
        List resultList = new List(this.size);
        int low = 0;
        int high = size - 1;
        int initialMatchIndex = -1;

        while (low <= high) {
            int mid = low + (high - low) / 2;
            int comparison = data[mid].getGrade().compareTo(targetGrade);

            if (comparison == 0) {
                initialMatchIndex = mid;
                break;
            } else if (comparison < 0) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }

        if (initialMatchIndex != -1) {
            resultList.add(data[initialMatchIndex]);

            int tempIndex = initialMatchIndex - 1;
            while (tempIndex >= 0 &&
data[tempIndex].getGrade().equals(targetGrade)) {
                resultList.add(data[tempIndex]);
                tempIndex--;
```

```
            }


            tempIndex = initialMatchIndex + 1;
            while (tempIndex < size &&
data[tempIndex].getGrade().equals(targetGrade)) {
                resultList.add(data[tempIndex]);
                tempIndex++;
            }
        }
        return resultList;
    }

    public List findStudentsByGradeSequential(String targetGrade) {
        List resultList = new List(this.size);
        for (int i = 0; i < size; i++) {
            if (data[i].getGrade().equals(targetGrade)) {
                resultList.add(data[i]);
            }
        }
        return resultList;
    }
}
```

```
package Question01;

public class Student {
    String studentNo;
    String name;
    char gender;
    char grade;

    public Student(String studentNo, String name, char gender, char grade) {
        this.studentNo = studentNo;
        this.name = name;
        this.gender = gender;
        this.grade = grade;
    }

    public String getStudentNumber() {
        return studentNo;
    }

    public String getName() {
        return name;
    }

    public char getGender() {
        return gender;
    }

    public String getGrade() {
           return String.valueOf(grade);
    }
```

```
    public void setGrade(char grade) {
        this.grade = grade;
    }

    public void setGrade(String grade) {
        if (grade != null && grade.length() == 1) {
            this.grade = grade.charAt(0);
        } else {
            System.err.println("Invalid grade format: " + grade);
        }
    }

    @Override
    public String toString() {
        return "Student{" +
                "studentNo='" + studentNo + '\'' +
                ", name='" + name + '\'' +
                ", gender=" + gender +
                ", grade=" + grade +
                '}';
    }
}
```

```
package Question01;


public class MainApp {
    public static void main(String[] args) {
        List studentDetails = new List(10);


        studentDetails.add(new Student("PS/2017/016", "Amal", 'M', 'A'));
        studentDetails.add(new Student("PS/2017/198", "Binura", 'M', 'A'));
        studentDetails.add(new Student("PS/2017/301", "Chanaka", 'M', 'A'));
        studentDetails.add(new Student("PS/2017/045", "Sarath", 'M', 'B'));
        studentDetails.add(new Student("PS/2017/149", "Nirmala", 'F', 'B'));
        studentDetails.add(new Student("PS/2017/151", "Sithara", 'F', 'B'));
        studentDetails.add(new Student("PS/2017/280", "Kamal", 'M', 'B'));
        studentDetails.add(new Student("PS/2017/273", "Kasuni", 'F', 'C'));
        studentDetails.add(new Student("PS/2017/312", "Akila", 'F', 'C'));
        studentDetails.add(new Student("PS/2017/105", "Dasuni", 'F', 'D'));


        System.out.println("All Student Details:");
        studentDetails.display();
        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'B' (Sequential Search):");
        List studentsWithGradeBSeq =
studentDetails.findStudentsByGradeSequential("B");
        studentsWithGradeBSeq.display();
        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'A' (Sequential Search):");
        List studentsWithGradeASeq =
studentDetails.findStudentsByGradeSequential("A");
        studentsWithGradeASeq.display();
```

```
        System.out.println("\n-----------------------------------------------
--\n");


        studentDetails.sortByGrade();

        System.out.println("Student details after sorting by grade (for Binary
Search):");

        studentDetails.display();

        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'B' (Binary Search after
sorting):");

        List studentsWithGradeBBin =
studentDetails.findStudentsByGradeBinary("B");

        studentsWithGradeBBin.display();

        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'A' (Binary Search after
sorting):");

        List studentsWithGradeABin =
studentDetails.findStudentsByGradeBinary("A");

        studentsWithGradeABin.display();

        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'D' (Binary Search after
sorting):");

        List studentsWithGradeDBin =
studentDetails.findStudentsByGradeBinary("D");

        studentsWithGradeDBin.display();

        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Students with Grade 'C' (Sequential Search -
original list order):");
```

```
        List studentsWithGradeCSeq =
studentDetails.findStudentsByGradeSequential("C");

        studentsWithGradeCSeq.display();

        System.out.println("\n-------------------------------------------------
--\n");


        System.out.println("Students with Grade 'C' (Binary Search - on sorted
list):");

        List studentsWithGradeCBin =
studentDetails.findStudentsByGradeBinary("C");

        studentsWithGradeCBin.display();

        System.out.println("\n-------------------------------------------------
--\n");


    }

}
```

```
kavindus@kavindus-MacBook-Air-2 BECS-21223-Data-Structures-and-Algorithms-LAB-
ualMachines/temurin-24.jdk/Contents/Home/bin/java --enable-preview -XX:+ShowCo
avindus/Library/Application\ Support/Windsurf/User/workspaceStorage/960a05bb83
/BECS-21223-Data-Structures-and-Algorithms-LAB-07_dd24fa3/bin Question01.MainA
All Student Details:
Student Number  Name            Gender   Grade
-------------------------------------------------
PS/2017/016     Amal            M        A
PS/2017/198     Binura          M        A
PS/2017/301     Chanaka         M        A
PS/2017/045     Sarath          M        B
PS/2017/149     Nirmala         F        B
PS/2017/151     Sithara         F        B
PS/2017/280     Kamal           M        B
PS/2017/273     Kasuni          F        C
PS/2017/312     Akila           F        C
PS/2017/105     Dasuni          F        D

-------------------------------------------------

Students with Grade 'B' (Sequential Search):
Student Number  Name            Gender   Grade
-------------------------------------------------
PS/2017/045     Sarath          M        B
PS/2017/149     Nirmala         F        B
PS/2017/151     Sithara         F        B
PS/2017/280     Kamal           M        B

-------------------------------------------------

Students with Grade 'A' (Sequential Search):
Student Number  Name            Gender   Grade
-------------------------------------------------
PS/2017/016     Amal            M        A
PS/2017/198     Binura          M        A
PS/2017/301     Chanaka         M        A

-------------------------------------------------

Student details after sorting by grade (for Binary Search):
Student Number  Name            Gender   Grade
-------------------------------------------------
PS/2017/016     Amal            M        A
PS/2017/198     Binura          M        A
PS/2017/301     Chanaka         M        A
PS/2017/045     Sarath          M        B
PS/2017/149     Nirmala         F        B
PS/2017/151     Sithara         F        B
PS/2017/280     Kamal           M        B
PS/2017/273     Kasuni          F        C
PS/2017/312     Akila           F        C
PS/2017/105     Dasuni          F        D

-------------------------------------------------
```

```
--------------------------------------------------

Student details after sorting by grade (for Binary Search):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/016     Amal            M        A
PS/2017/198     Binura          M        A
PS/2017/301     Chanaka         M        A
PS/2017/045     Sarath          M        B
PS/2017/149     Nirmala         F        B
PS/2017/151     Sithara         F        B
PS/2017/280     Kamal           M        B
PS/2017/273     Kasuni          F        C
PS/2017/312     Akila           F        C
PS/2017/105     Dasuni          F        D

--------------------------------------------------

Students with Grade 'B' (Binary Search after sorting):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/149     Nirmala         F        B
PS/2017/045     Sarath          M        B
PS/2017/151     Sithara         F        B
PS/2017/280     Kamal           M        B

--------------------------------------------------

Students with Grade 'A' (Binary Search after sorting):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/198     Binura          M        A
PS/2017/016     Amal            M        A
PS/2017/301     Chanaka         M        A

--------------------------------------------------

Students with Grade 'D' (Binary Search after sorting):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/105     Dasuni          F        D

--------------------------------------------------

Students with Grade 'C' (Sequential Search - original list order):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/273     Kasuni          F        C
PS/2017/312     Akila           F        C

--------------------------------------------------

Students with Grade 'C' (Binary Search - on sorted list):
Student Number  Name            Gender   Grade
--------------------------------------------------
PS/2017/273     Kasuni          F        C
PS/2017/312     Akila           F        C

--------------------------------------------------

) kavindus@kavindus-MacBook-Air-2 BECS-21223-Data-Structures-and-Algorithms-LAB-07 %
```

Question 02.

```
package Question02;

public class Employee {
    String empId;
    String name;
    String department;
    char grade;

    public Employee(String empId, String name, String department, char grade) {
        this.empId = empId;
        this.name = name;
        this.department = department;
        this.grade = grade;
    }

    public String getEmpId() {
        return empId;
    }
    public String getName() {
        return name;
    }

    public String getDepartment() {
        return department;
    }

    public String getGrade() {
        return String.valueOf(grade);
    }
}
```

```
package Question02;

public class LinkedList {
    private Node head;
    private int size;

    private class Node {
        Employee data;
        Node next;

        Node(Employee data) {
            this.data = data;
            this.next = null;
        }
    }

    public void add(Employee employee) {
        Node newNode = new Node(employee);
        if (head == null) {
            head = newNode;
        } else {
            Node current = head;
            while (current.next != null) {
                current = current.next;
            }
            current.next = newNode;
        }
        size++;
    }

    public void display() {
```

```
        if (head == null) {
            System.out.println("List is empty.");
            return;
        }
        System.out.printf("%-10s %-15s %-15s %-6s\n", "Employee ID", "Name",
"Department", "Grade");
        System.out.println("-----------------------------------------------------
-----");
        Node current = head;
        while (current != null) {
            System.out.printf("%-10s %-15s %-15s %-6s\n",
                    current.data.getEmpId(), current.data.getName(),
                    current.data.getDepartment(), current.data.getGrade());
            current = current.next;
        }
    }

    public void insertionSortByGrade() {
        if (head == null || head.next == null) {
            return;
        }

        Node sorted = null;
        Node current = head;

        while (current != null) {
            Node next = current.next;

            if (sorted == null ||
sorted.data.getGrade().compareTo(current.data.getGrade()) > 0) {
                current.next = sorted;
                sorted = current;
```

```
            } else {
                Node temp = sorted;
                while (temp.next != null &&
temp.next.data.getGrade().compareTo(current.data.getGrade()) <= 0) {
                    temp = temp.next;
                }
                current.next = temp.next;
                temp.next = current;
            }
            current = next;
        }
        head = sorted;
    }

    public LinkedList findEmployeesByGrade(String targetGrade) {
        LinkedList resultList = new LinkedList();
        Node current = head;

        while (current != null) {
            if (current.data.getGrade().equals(targetGrade)) {
                resultList.add(current.data);
            }
            current = current.next;
        }
        return resultList;
    }
}
```

```
package Question02;


public class List {
    private Employee[] data;
    private int size;
    private int capacity;

    public List(int capacity) {
        this.capacity = capacity;
        this.data = new Employee[capacity];
        this.size = 0;
    }

    public void add(Employee employee) {
        if (size < capacity) {
            data[size++] = employee;
        } else {
            System.out.println("List is full. Cannot add employee.");
        }
    }

    public void display() {
        if (size == 0) {
            System.out.println("List is empty.");
            return;
        }
        System.out.printf("%-10s %-15s %-15s %-6s\n", "Employee ID", "Name",
"Department", "Grade");
        System.out.println("----------------------------------------------------
-----");
        for (int i = 0; i < size; i++) {
            System.out.printf("%-10s %-15s %-15s %-6s\n",
```

```
                    data[i].getEmpId(), data[i].getName(),
                    data[i].getDepartment(), data[i].getGrade());
        }
    }

    public void insertionSortByGrade() {
        for (int i = 1; i < size; i++) {
            Employee key = data[i];
            int j = i - 1;

            while (j >= 0 && data[j].getGrade().compareTo(key.getGrade()) > 0)
{
                data[j + 1] = data[j];
                j = j - 1;
            }
            data[j + 1] = key;
        }
    }

    public List findEmployeesByGradeBinary(String targetGrade) {
        List resultList = new List(this.size);
        int low = 0;
        int high = size - 1;
        int initialMatchIndex = -1;

        while (low <= high) {
            int mid = low + (high - low) / 2;
            int comparison = data[mid].getGrade().compareTo(targetGrade);

            if (comparison == 0) {
                initialMatchIndex = mid;
                break;
```

```
            } else if (comparison < 0) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }

        if (initialMatchIndex != -1) {
            resultList.add(data[initialMatchIndex]);

            int tempIndex = initialMatchIndex - 1;
            while (tempIndex >= 0 &&
data[tempIndex].getGrade().equals(targetGrade)) {
                resultList.add(data[tempIndex]);
                tempIndex--;
            }

            tempIndex = initialMatchIndex + 1;
            while (tempIndex < size &&
data[tempIndex].getGrade().equals(targetGrade)) {
                resultList.add(data[tempIndex]);
                tempIndex++;
            }
        }
        return resultList;
    }
}
```

```
package Question02;


public class MainApp {
    public static void main(String[] args) {
        // Array implementation
        System.out.println("Array Implementation:");
        System.out.println("====================\n");

        List employeeList = new List(10);

        employeeList.add(new Employee("EMP001", "Anjali", "HR", 'B'));
        employeeList.add(new Employee("EMP002", "Roshan", "Finance", 'A'));
        employeeList.add(new Employee("EMP003", "Meera", "IT", 'C'));
        employeeList.add(new Employee("EMP004", "Hiran", "HR", 'A'));
        employeeList.add(new Employee("EMP005", "Sanjay", "Marketing", 'B'));
        employeeList.add(new Employee("EMP006", "Vimukthi", "Finance", 'D'));
        employeeList.add(new Employee("EMP007", "Dilani", "IT", 'C'));
        employeeList.add(new Employee("EMP008", "Tharindu", "Marketing", 'A'));
        employeeList.add(new Employee("EMP009", "Ishara", "HR", 'B'));
        employeeList.add(new Employee("EMP010", "Lahiru", "IT", 'D'));

        System.out.println("All Employee Details:");
        employeeList.display();
        System.out.println("\n-----------------------------------------------
--\n");

        System.out.println("Sorting by Grade (Insertion Sort on Array):");
        employeeList.insertionSortByGrade();
        employeeList.display();
        System.out.println("\n-----------------------------------------------
--\n");
```

```
        System.out.println("Employees with Grade 'B' (Binary Search on sorted
array):");
        List gradeBEmployees = employeeList.findEmployeesByGradeBinary("B");
        gradeBEmployees.display();
        System.out.println("\n-----------------------------------------------
--\n");


        // Linked List implementation
        System.out.println("Linked List Implementation:");
        System.out.println("==========================\n");

        LinkedList employeeLinkedList = new LinkedList();

        employeeLinkedList.add(new Employee("EMP001", "Anjali", "HR", 'B'));
        employeeLinkedList.add(new Employee("EMP002", "Roshan", "Finance",
'A'));
        employeeLinkedList.add(new Employee("EMP003", "Meera", "IT", 'C'));
        employeeLinkedList.add(new Employee("EMP004", "Hiran", "HR", 'A'));
        employeeLinkedList.add(new Employee("EMP005", "Sanjay", "Marketing",
'B'));
        employeeLinkedList.add(new Employee("EMP006", "Vimukthi", "Finance",
'D'));
        employeeLinkedList.add(new Employee("EMP007", "Dilani", "IT", 'C'));
        employeeLinkedList.add(new Employee("EMP008", "Tharindu", "Marketing",
'A'));
        employeeLinkedList.add(new Employee("EMP009", "Ishara", "HR", 'B'));
        employeeLinkedList.add(new Employee("EMP010", "Lahiru", "IT", 'D'));

        System.out.println("All Employee Details:");
        employeeLinkedList.display();
        System.out.println("\n-----------------------------------------------
--\n");
```

```
        System.out.println("Sorting by Grade (Insertion Sort on Linked
List):");

        employeeLinkedList.insertionSortByGrade();

        employeeLinkedList.display();

        System.out.println("\n-----------------------------------------------
--\n");


        System.out.println("Employees with Grade 'A' (Search on linked
list):");

        LinkedList gradeAEmployees =
employeeLinkedList.findEmployeesByGrade("A");

        gradeAEmployees.display();

    }

}
```

```
kavindus@kavindus-MacBook-Air-2 BECS-21223-Data-Structures
-Data-Structures-and-Algorithms-LAB-07 ; /usr/bin/env /Lib
n/java --enable-preview -XX:+ShowCodeDetailsInExceptionMes
rf/User/workspaceStorage/960a05bb83905651cd538de11a632512/
LAB-07_dd24fa3/bin Question02.MainApp
Array Implementation:
====================

All Employee Details:
Employee ID Name            Department      Grade
-----------------------------------------------------
EMP001      Anjali          HR              B
EMP002      Roshan          Finance         A
EMP003      Meera           IT              C
EMP004      Hiran           HR              A
EMP005      Sanjay          Marketing       B
EMP006      Vimukthi        Finance         D
EMP007      Dilani          IT              C
EMP008      Tharindu        Marketing       A
EMP009      Ishara          HR              B
EMP010      Lahiru          IT              D

-----------------------------------------------

Sorting by Grade (Insertion Sort on Array):
Employee ID Name            Department      Grade
-----------------------------------------------------
EMP002      Roshan          Finance         A
EMP004      Hiran           HR              A
EMP008      Tharindu        Marketing       A
EMP001      Anjali          HR              B
EMP005      Sanjay          Marketing       B
EMP009      Ishara          HR              B
EMP003      Meera           IT              C
EMP007      Dilani          IT              C
EMP006      Vimukthi        Finance         D
EMP010      Lahiru          IT              D

-----------------------------------------------

Employees with Grade 'B' (Binary Search on sorted array):
Employee ID Name            Department      Grade
-----------------------------------------------------
EMP005      Sanjay          Marketing       B
EMP001      Anjali          HR              B
EMP009      Ishara          HR              B

-----------------------------------------------
```

```
--------------------------------------------------

Linked List Implementation:
==========================

All Employee Details:
Employee ID Name            Department      Grade
--------------------------------------------------------
EMP001      Anjali          HR              B
EMP002      Roshan          Finance         A
EMP003      Meera           IT              C
EMP004      Hiran           HR              A
EMP005      Sanjay          Marketing       B
EMP006      Vimukthi        Finance         D
EMP007      Dilani          IT              C
EMP008      Tharindu        Marketing       A
EMP009      Ishara          HR              B
EMP010      Lahiru          IT              D

--------------------------------------------------

Sorting by Grade (Insertion Sort on Linked List):
Employee ID Name            Department      Grade
--------------------------------------------------------
EMP002      Roshan          Finance         A
EMP004      Hiran           HR              A
EMP008      Tharindu        Marketing       A
EMP001      Anjali          HR              B
EMP005      Sanjay          Marketing       B
EMP009      Ishara          HR              B
EMP003      Meera           IT              C
EMP007      Dilani          IT              C
EMP006      Vimukthi        Finance         D
EMP010      Lahiru          IT              D

--------------------------------------------------

Employees with Grade 'A' (Search on linked list):
Employee ID Name            Department      Grade
--------------------------------------------------------
EMP002      Roshan          Finance         A
EMP004      Hiran           HR              A
EMP008      Tharindu        Marketing       A
kavindus@kavindus-MacBook-Air-2 BECS-21223-Data-Structures-an
```