Set Interface:

Sets stores unique elements and does not allow duplicate elements.

Two common implementations – HashSet & LinkedHashSet

1. HashSet: HashSet (Java SE 21 & JDK 21)
   - It is a class which implements the set interface.
   - It allows null values and stores elements in unordered way.

Characteristics:

- Does not allow any order of elements
- Best suited for quick adding, removing and contains operation

```
package setEx;

import java.util.HashSet;

public class HashSetExample {
    public static void main(String[] args) {
        //Creating hashset
        HashSet<String> fruits = new HashSet<>();

        //Adding elements to the hashset
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Kiwi");
        fruits.add("Orange");
        fruits.add(null);

        //displaying the hashset
        System.out.println("HashSet: "+ fruits);

        //Checking the element exist or not
        System.out.println("Does hashset contains
grapes? "+fruits.contains("grapes"));
```

```
        //Removing an element
        fruits.remove("Orange");
        System.out.println("After removal :"+fruits);

        //To check the size of the set
        System.out.println("Size of HashSet: "+
fruits.size());

        //To remove all elements from the set
        fruits.clear();
        System.out.println("Is HashSet Empty?
:"+fruits.isEmpty());
    }
}
```

o/p:

HashSet: [null, Apple, Kiwi, Orange, Banana]

Does hashset contains grapes? false

After removal :[null, Apple, Kiwi, Banana]

Size of HashSet: 4

Is HashSet Empty? :true

LinkedHashSet: LinkedHashSet (Java SE 21 & JDK 21)

It is similar to hashset but maintains the doubly linkedlist across all elements.

Characteristics:

1. Maintains the insertion order
2. Slightly slower than hashset because it maintains the order

```
package setEx;

import java.util.LinkedHashSet;

public class LinkedHashSetExample {
    public static void main(String[] args) {
        //Creating LHS
        LinkedHashSet<String> cities = new
LinkedHashSet<>();

        //Adding elements to the lHS
        cities.add("Pune");
        cities.add("Sambhajinagar");
        cities.add("Mumbai");
        cities.add("Nagpur");

        //Displaying the LHS
        System.out.println("LHS: "+cities);

        //To check LHS contains an element or not
        System.out.println("Does LHS contain Mumbai? "+
cities.contains("Mumbai"));
    }
}
```

o/p:

LHS: [Pune, Sambhajinagar, Mumbai, Nagpur]

Does LHS contain Mumbai? True

Employee Management System

Step1: Employee Class

```
package ems;

public class Employee {
    private int id;
    private String name;
    private String designation;
    private double salary;

    //Constructor
    public Employee(String designation, int id, String name,
double salary) {
        this.designation = designation;
        this.id = id;
        this.name = name;
        this.salary = salary;
    }

    //Getters and Setters
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDesignation() {
        return designation;
    }

    public void setDesignation(String designation) {
        this.designation = designation;
    }
```

```
    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    @Override
    public String toString() {
        return "Employee{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", designation='" + designation + '\'' +
                ", salary=" + salary +
                '}';
    }
}
```

Step2:

```
package ems;

public interface EmployeeManagement {
    void addEmployee(Employee employee);
    void viewAllEmployees();
    void updateEmployee(int empId);
    void deleteEmployee(int empId);
    Employee findEmployeeById(int empId);
}
```

Step3:

```
package ems;

import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class EmployeeManagementImpl implements
EmployeeManagement{

   private List<Employee> employeeList = new ArrayList<>();

   //Adding an employee to the list
    @Override
```

```
    public void addEmployee(Employee employee) {
        employeeList.add(employee);
        System.out.println("Employee Added Successfully!!");
    }

    //To view all employees
    @Override
    public void viewAllEmployees() {
        if (employeeList.isEmpty()){
            System.out.println("No Employees Found!!");
        }else {
            System.out.println("List of Employees: ");
            for (Employee employee: employeeList){
                System.out.println(employee);
            }
        }
    }

    @Override
    public void updateEmployee(int empId) {
        Employee employee = findEmployeeById(empId);
        if (employee != null){
            Scanner scanner = new Scanner(System.in);
            System.out.println("Enter New Name: ");
            employee.setName(scanner.nextLine());
            System.out.println("Enter New Designation: ");
            employee.setDesignation(scanner.nextLine());
            System.out.println("Enter Employee Salary: ");
            employee.setSalary(scanner.nextDouble());
            System.out.println("Employee Updated
Successfully!!");
        }else {
            System.out.println("Employee Not Found");
        }
    }

    @Override
    public void deleteEmployee(int empId) {
        Employee employee = findEmployeeById(empId);
        if (employee != null){
            employeeList.remove(employee);
            System.out.println("Employee Deleted
Successfully!!!");
        } else {
            System.out.println("Employee Not Found!!");
        }
    }

    @Override
    public Employee findEmployeeById(int empId) {
        for (Employee employee: employeeList){
```

```
            if (employee.getId() == empId){
                return employee;
            }
        }
        return null;
    }
}
```

Step4:

```
package ems;

import java.util.Scanner;

public class EmployeeManagementSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        EmployeeManagementImpl management = new
EmployeeManagementImpl();
        boolean exit = false;

        System.out.println("Welcome!!!!!!");
        while(!exit){
            System.out.println("\n Menu: ");
            System.out.println("1. Add Employee ");
            System.out.println("2. View All Employees");
            System.out.println("3. Update Employee");
            System.out.println("4. Delete Employee");
            System.out.println("5. Exit ");
            System.out.println("Enter your choice: ");
            int choice = scanner.nextInt();

            switch (choice){
                case 1:
                    System.out.println("Enter Employee ID:
");
                    int id = scanner.nextInt();
                    scanner.nextLine();  //Consumes the new
line

                    System.out.println("Enter Employee Name:
");
                    String name = scanner.nextLine();

                    System.out.println("Enter Employee
Designation: ");
                    String designation = scanner.nextLine();
```

```
                    System.out.println("Enter Employee Salary:
");
                    double salary=  scanner.nextDouble();

                    Employee employee = new
Employee(designation,id,name,salary);
                    management.addEmployee(employee);
                    break;

                case 2:
                    management.viewAllEmployees();
                    break;

                case 3:
                    System.out.println("Enter Employee Id to
update: ");
                    int updateId = scanner.nextInt();
                    management.updateEmployee(updateId);
                    break;

                case 4:
                    System.out.println("Enter employee Id to
delete: ");
                    int deleteId = scanner.nextInt();
                    management.deleteEmployee(deleteId);
                    break;

                case 5:
                    exit = true;
                    System.out.println("Come Again!!!!");
                    break;

                default:
                    System.out.println("Invalid Choice!!");


            }
        }
    }
}
```

Product Management System

productId, productName, productPrice, expiryDate*