2024-2학기 자바프로그래밍2 13주차 실습

문제 1

```java
package week13;

class MyArrayAlg {
    public static <T extends Comparable<T>> T getMin(T[] arr) {
        if (arr == null || arr.length == 0) {
            return null;
        }
        T min = arr[0];
        for (T e : arr) {
            if (e.compareTo(min) < 0) {
                min = e;
            }
        }
        return min;
    }

    public static <T> boolean contains(T[] arr, T element) {
        if (arr == null || arr.length == 0) {
            return false;
        }
        for (T e : arr) {
            if (e.equals(element)) {
                return true;
            }
        }
        return false;
    }

    public static <T> void reverse(T[] arr) {
        if (arr == null || arr.length <= 1) {
            return;
        }
        for (int i = 0; i < arr.length / 2; i++) {
            T temp = arr[i];
            arr[i] = arr[arr.length - 1 - i];
            arr[arr.length - 1 - i] = temp;
        }
    }

    public static <T extends Comparable<T>> void sort(T[] arr) {
        if (arr == null || arr.length <= 1) {
            return;
        }
        for (int i = 0; i < arr.length; i++) {
            for (int j = i+1; j < arr.length; j++) {
                if (arr[i].compareTo(arr[j]) > 0) {
                    T temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
    }

    public static <T> void printArray(T[] arr) {
        for (T e : arr) {
            System.out.print(e + " ");
        }
        System.out.println();
    }

}
```

```java
public class Exercise01 {

    public static void main(String[] args) {
        Integer[] iArray = { 5, 16, 10, 8, 9 };
        Double[] dArray = { 1.5, 6.6, 7.4, 3.3, 1.5 };
        Character[] cArray = { 'J', 'A', 'V', 'A' };
        String[] sArray = { "Java", "Programming", "Python", "C" };

        System.out.print("Min: ");
        System.out.print(MyArrayAlg.getMin(iArray) + " ");
        System.out.print(MyArrayAlg.getMin(dArray) + " ");
        System.out.print(MyArrayAlg.getMin(cArray) + " ");
        System.out.println(MyArrayAlg.getMin(sArray) + "\n");

        System.out.print("Contains: ");
        System.out.print(MyArrayAlg.contains(iArray, 8) + " ");
        System.out.print(MyArrayAlg.contains(dArray, 5.5) + " ");
        System.out.print(MyArrayAlg.contains(cArray, 'A') + " ");
        System.out.println(MyArrayAlg.contains(sArray, "C++") + "\n");

        System.out.println("Reverse: ");
        MyArrayAlg.reverse(iArray);
        MyArrayAlg.reverse(dArray);
        MyArrayAlg.reverse(cArray);
        MyArrayAlg.reverse(sArray);
        MyArrayAlg.printArray(iArray);
        MyArrayAlg.printArray(dArray);
        MyArrayAlg.printArray(cArray);
        MyArrayAlg.printArray(sArray);
        System.out.println();

        System.out.println("Sort: ");
        MyArrayAlg.sort(iArray);
        MyArrayAlg.sort(dArray);
        MyArrayAlg.sort(cArray);
        MyArrayAlg.sort(sArray);
        MyArrayAlg.printArray(iArray);
        MyArrayAlg.printArray(dArray);
        MyArrayAlg.printArray(cArray);
        MyArrayAlg.printArray(sArray);

    }

}
```

문제 2

```java
package week13;

import java.util.ArrayList;

class BankManager {
    // 동일 계좌가 없으면 추가
    public static <T extends Account> void addAccount(List<T> accounts, T a) {
        if(!accounts.contains(a))
            accounts.add(a);
    }

    // 전체 계좌 출력
    public static void printAccounts(List<? extends Account> accounts) {
        for (Account account : accounts) {
            System.out.println(account);
        }
    }

    // 저축 계좌만 출력
    public static void printSavingsAccounts(List<? super SavingsAccount> accounts) {
        List<SavingsAccount> savingsAccounts = new ArrayList<>();
        for (Object obj : accounts) {
            if (obj instanceof SavingsAccount) {
                savingsAccounts.add((SavingsAccount) obj);
            }
        }

        savingsAccounts.sort((a, b) -> Double.compare(b.getInterestRate(), a.getInterestRate()));

        for (SavingsAccount account : savingsAccounts) {
            System.out.println(account);
        }
    }

    // 입출금 계좌만 출력
    public static void printCheckingAccounts(List<? super CheckingAccount> accounts) {
        List<CheckingAccount> checkingAccounts = new ArrayList<>();
        for (Object obj : accounts) {
            if (obj instanceof CheckingAccount) {
                checkingAccounts.add((CheckingAccount) obj);
            }
        }

        checkingAccounts.sort((a, b) -> Integer.compare(b.getBalance(), a.getBalance()));

        for (CheckingAccount account : checkingAccounts) {
            System.out.println(account);
        }
    }

    // 총 잔액 계산
    public static int calculateTotalBalance(List<? extends Account> accounts) {
        int total = 0;
        for (Account account : accounts) {
            total += account.getBalance();
        }
        return total;
    }
}

public class Exercise02 {

    public static void main(String[] args) {
        List<Account> accounts = new ArrayList<>();
        BankManager.addAccount(accounts, new SavingsAccount("홍길동", "20240001-1", 1000000, 0.05));
        BankManager.addAccount(accounts, new CheckingAccount("홍길동", "20240001-2", 100000, 100000));
        BankManager.addAccount(accounts, new CheckingAccount("홍길동", "20240001-3", 500000, 100000));
        BankManager.addAccount(accounts, new SavingsAccount("홍길동", "20240001-3", 5000, 0.07));
        BankManager.addAccount(accounts, new SavingsAccount("홍길동", "20240001-4", 1000000, 0.08));


        System.out.println("전체 계좌:");
        BankManager.printAccounts(accounts);
        System.out.println();

        System.out.println("저축 계좌 (이자율 높은 순):");
        BankManager.printSavingsAccounts(accounts);
        System.out.println();

        System.out.println("입출금 계좌 (잔액 높은 순):");
        BankManager.printCheckingAccounts(accounts);
        System.out.println();

        int totalBalance = BankManager.calculateTotalBalance(accounts);
        System.out.println("전체 계좌 총 잔액: " + totalBalance + "원");
        System.out.println();

    }

}
```

문제 3

```java
package week13;

public class Student implements Comparable {
    private String id;
    private String name;
    private String department;
    private double gpa;

    public Student(String id, String name, String department, double gpa) {
        this.id = id;
        this.name = name;
        this.department = department;
        this.gpa = gpa;
    }

    public double getGpa() {
        return gpa;
    }

    @Override
    public String toString() {
        return String.format("학번: %s, 이름: %s, 학과: %s, 성적: %.2f", id, name, department, gpa);
    }

    @Override
    public int compareTo(Object other) {
        Student student = (Student) other;
        return Double.compare(student.gpa, gpa);
    }

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getDepartment() {
        return department;
    }
}
```

```java
package week13;

import java.util.*;

public class Exercise03 {
    public static void main(String[] args) {
        Map<String, Student> studentMap = new HashMap<>();
        Map<String, List<Student>> departmentMap = new HashMap<>();
        Scanner input = new Scanner(System.in);

        System.out.println("======menu======");
        System.out.println("0. 학생 추가");
        System.out.println("1. 학번으로 조회");
        System.out.println("2. 학과별 조회");
        System.out.println("3. 성적 상위 3명 조회");
        System.out.println("4. 종료");
        System.out.println("================");

        while (true) {
            System.out.print("메뉴를 선택하세요: ");
            int choice = input.nextInt();
            input.nextLine(); // 버퍼 정리

            switch (choice) {
                case 0: // 학생 추가
                    System.out.print("학번: ");
                    String id = input.nextLine();
                    if (studentMap.containsKey(id)) {
                        System.out.println("해당 학번의 학생이 이미 존재합니다.");
                        break;
                    }
                    System.out.print("이름: ");
                    String name = input.nextLine();
                    System.out.print("학과명: ");
                    String department = input.nextLine();
                    System.out.print("성적: ");
                    double gpa = input.nextDouble();

                    Student student = new Student(id, name, department, gpa);
                    studentMap.put(id, student);

                    departmentMap.putIfAbsent(department, new ArrayList<>());
                    departmentMap.get(department).add(student);

                    System.out.println(name + " 학생이 추가되었습니다.");
                    break;

                case 1: // 학번으로 학생 검색
                    System.out.print("학번: ");
                    String searchId = input.nextLine();
                    Student stu = studentMap.get(searchId);
                    if (stu != null) {
                        System.out.println(stu);
                    } else {
                        System.out.println("해당 학번의 학생을 찾을 수 없습니다.");
                    }
                    break;

                case 2: // 학과별 학생 조회
                    System.out.print("학과명: ");
                    String searchDept = input.nextLine();
                    List<Student> deptStudents = departmentMap.get(searchDept);
                    if (deptStudents != null && !deptStudents.isEmpty()) {
                        for (Student s : deptStudents) {
                            System.out.println(s);
                        }
                    } else {
                        System.out.println("해당 학과에 등록된 학생이 없습니다.");
                    }
                    break;

                case 3: // 성적이 높은 학생 3명 조회
                    PriorityQueue<Student> maxHeap = new PriorityQueue<>(
                        (s1, s2) -> Double.compare(s2.getGpa(), s1.getGpa())
                    );
                    maxHeap.addAll(studentMap.values());

                    for (int i = 0; i < 3 && !maxHeap.isEmpty(); i++) {
                        System.out.println(maxHeap.remove());
                    }
                    break;

                case 4: // 종료
                    System.out.println("프로그램 종료");
                    input.close();
                    return;

                default:
                    System.out.println("잘못된 입력입니다. 다시 선택하세요.");
            }
            System.out.println();
        }
    }
}
```