Zadanie 2

Cele główne

- Praktyczne zastosowanie kolekcji Java (List, Map, Set),
- Implementacja interfejsów Comparable i Comparator,
- Praca z typami wyliczeniowymi (enum),
- Wykorzystanie AI jako narzędzia wspomagającego programowanie,
- Rozróżnienie między samodzielnym kodowaniem a asystowanym przez AI.

Cele szczegółowe

Po zakończeniu zajęć student potrafi:

- Implementować klasy z interfejsami Comparable i Comparator,
- Wykorzystywać różne typy kolekcji Java do zarządzania danymi,
- Tworzyć i używać typy wyliczeniowe,
- Efektywnie współpracować z AI w procesie rozwijania kodu,
- Krytycznie oceniać i weryfikować kod generowany przez AI.

2. STRUKTURA ZAJĘĆ

CZĘŚĆ I: Implementacja Podstawowa (90 minut)

Zadania do Wykonania Bez AI:

Zadanie 1: Typ Wyliczeniowy EmployeeCondition

```
/**
 * Cel: Zrozumienie enum i jego zastosowań
 */
public enum EmployeeCondition {
    // TODO: Zdefiniuj stany pracownika:
    // OBECNY, DELEGACJA, CHORY, NIEOBECNY
    // Dodaj metodę toString() zwracającą polskie nazwy
}
```

Zadanie 2: Klasa Employee

```
/**
 * Cel: Praktyka programowania obiektowego i interfejsu Comparable
 */
public class Employee implements Comparable<Employee> {
    // TODO: Pola klasy
    private String firstName;
    private String lastName;
    private EmployeeCondition condition;
    private int birthYear;
    private double salary;
```

```
    // TODO: Konstruktor

    // TODO: Gettery i settery

    // TODO: Metoda printing()

    // TODO: Implementacja compareTo() - sortowanie po nazwisku

    @Override
    public int compareTo(Employee other) {
        // Implementuj porównanie po nazwisku
    }
}
```

Zadanie 3: Podstawowe Metody ClassEmployee

```
/**
 * Cel: Praca z kolekcjami i podstawowe operacje CRUD
 */
public class ClassEmployee {
    private String groupName;
    private List<Employee> employees;
    private int maxCapacity;

    // TODO: Konstruktor

    // TODO: Podstawowe metody:
    public boolean addEmployee(Employee employee) {
        // Sprawdź duplikaty i pojemność
    }

    public boolean removeEmployee(Employee employee) {
        // Usuń pracownika
    }

    public void changeCondition(Employee employee, EmployeeCondition conditio
n) {
        // Zmień stan pracownika
    }

    public void addSalary(Employee employee, double amount) {
        // Dodaj do wynagrodzenia
    }

    public void summary() {
        // Wyświetl wszystkich pracowników
    }
}
```

...

Szczegółowe wymagania zamieszczone w pliku „” dostępnym na kursie na UPEL-u.

CZĘŚĆ II: Rozszerzenia z AI (90 minut)

Wprowadzenie do pracy z AI w IntelliJ IDEA

Zadania do wykonania z AI: zaawansowane wyszukiwanie, sortowanie i statystyki, rozszerzenia funkcjonalne i inne.