МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

ІНСТИТУТ ПРИКЛАДНОЇ МАТЕМАТИКИ ТА ФУНДАМЕНТАЛЬНИХ НАУК

Кафедра прикладної математики

**ЗВІТ**

про виконання лабораторної роботи №5 із дисципліни:

“**Математичні основи штучного інтелекту”**

Виконав:

студент групи ПМ-31

Матолін Марко

Прийняв:

доцент кафедри ПМ

Пабирівський В.В.

Львів ‒ 2023

**Тема:** «Генетичні алгоритми»

# Завдання

Розробити та реалізувати генетичний алгоритм для розв’язання задачі оптимізації на прикладі складання розкладу занять для початкової школи. Передбачити таке:

## • кількість різних уроків становить порядку 6–8 при загальній кількості усіх занять порядку 23–25 на тиждень; • кількість різних вчителів (за спеціалізацією уроків) становить 3–4, причому один вчитель (класний керівник) веде щонайменше половину уроків для свого класу; • кількість класів має бути у межах 2–3; • кількість уроків на день не повинна перевищувати 5; • розклад уроків не повинен мати «вікон» (пропусків уроків серед дня); • для проведення деяких уроків (фізвиховання, хореографія, музика) вимагається використання спеціалізованих приміщень відповідного типу (у школі є не більше одного приміщення такого типу);

• кількість усіх навчальних приміщень рівна кількості класів плюс кількість спеціалізованих приміщень.

## Примітка: Для зручності повторного використання передбачити запис / читання згенерованих

даних (перелік уроків, вчителів, класів тощо) у файл / з файлу.

## Етапи виконання завдання

1. Розробити структуру хромосоми для розв’язання задачі оптимізації.
2. Реалізувати допоміжну функцію для обчислення значення цільової функції.
3. Реалізувати допоміжну функцію ініціалізації початкової «популяції».
4. Реалізувати допоміжну функцію «мутації».

## 5. Реалізувати допоміжні функції одно-точкового та багато-точкового

«схрещування» (кросовера).

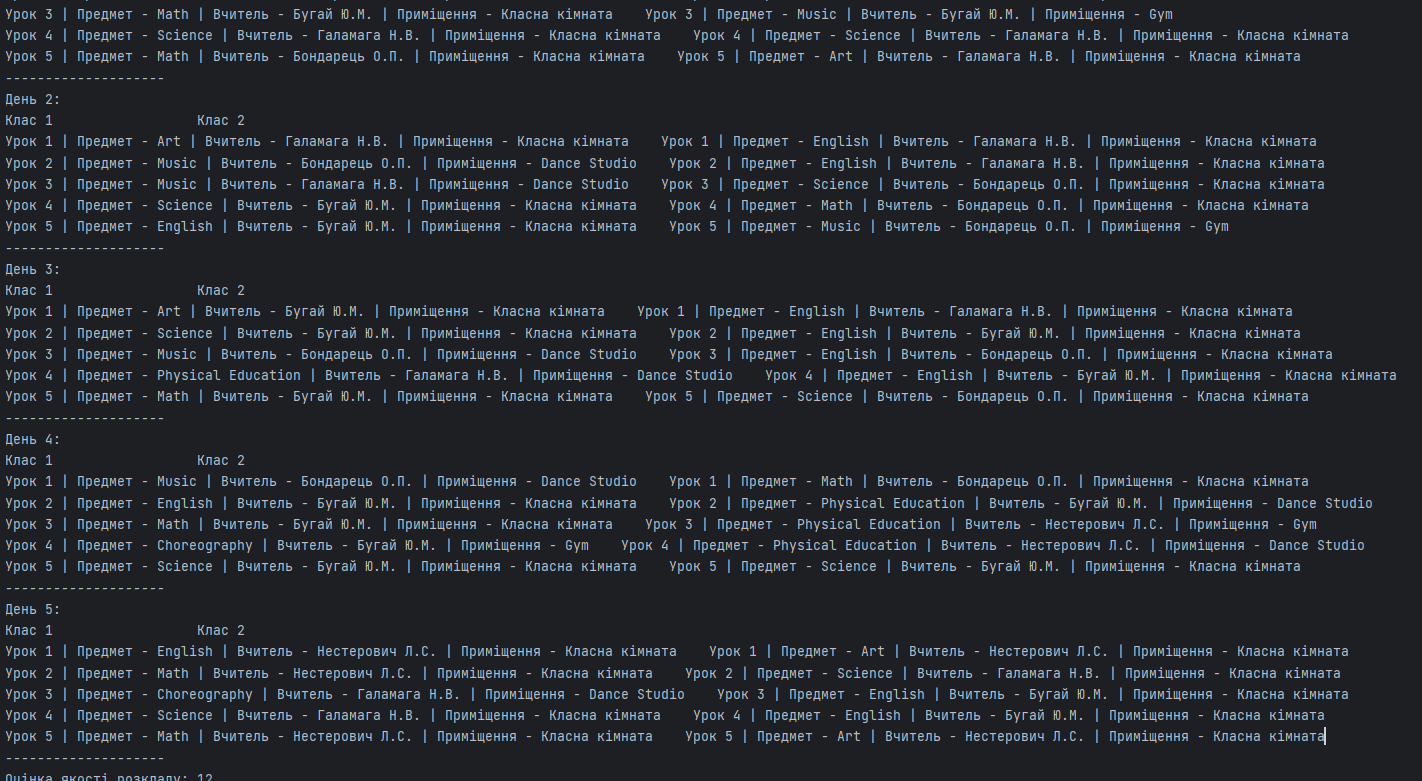
6. Безпосередньо реалізувати генетичний алгоритм оптимізації розкладу занять згідно завдання.

# Варіант-3

Код програми:

import random  
  
# Параметри задачі  
num\_classes = random.randint(2, 3)  
num\_teachers = random.randint(3, 4)  
num\_lessons\_per\_week = random.randint(23, 25)  
num\_lesson\_types = random.randint(6, 8)  
num\_days = 5  
  
lesson\_types = ['Math', 'Science', 'English', 'Art', 'Music', 'Physical Education', 'Choreography']  
special\_rooms = ['Gym', 'Dance Studio', 'Music Room']  
class\_teachers = ['Галамага Н.В.', 'Бондарець О.П.', 'Нестерович Л.С.', 'Бугай Ю.М.']  
teachers = ['Галамага Н.В.', 'Бондарець О.П.', 'Нестерович Л.С.', 'Бугай Ю.М.']  
  
# Генерація початкової популяції  
def create\_initial\_population(population\_size):  
 population = []  
 for \_ in range(population\_size):  
 schedule = []  
 for class\_id in range(num\_classes):  
 class\_schedule = []  
 for day in range(num\_days):  
 for lesson in range(1, num\_lessons\_per\_week // num\_days + 1):  
 lesson\_type = random.randint(0, num\_lesson\_types - 1)  
 teacher\_ids = random.sample(range(num\_teachers), num\_teachers)  
 room\_id = random.randint(0, num\_classes - 1)  
 class\_schedule.append((lesson + day \* (num\_lessons\_per\_week // num\_days), lesson\_type, teacher\_ids[class\_id], room\_id))  
 schedule.append(class\_schedule)  
 population.append(schedule)  
 return population  
  
  
# Оцінка пристосованості розкладу  
def evaluate\_fitness(schedule):  
 fitness = 0  
 for class\_schedule in schedule:  
 # Перевірка кількості уроків на день  
 daily\_lessons = [0] \* num\_days  
 for lesson, \_, \_, \_ in class\_schedule:  
 day = (lesson - 1) % num\_days  
 daily\_lessons[day] += 1  
 if any(num\_lessons > num\_lessons\_per\_week // num\_days for num\_lessons in daily\_lessons):  
 fitness += 1  
  
 # Перевірка "вікон" у розкладі  
 num\_windows = sum(1 for i in range(num\_lessons\_per\_week - 1) if class\_schedule[i][0] + 1 != class\_schedule[i + 1][0])  
 fitness += num\_windows  
  
 # Перевірка використання спеціалізованих приміщень  
 used\_special\_rooms = set()  
 for lesson, lesson\_type, teacher\_id, room\_id in class\_schedule:  
 if lesson\_type >= num\_lesson\_types - len(special\_rooms):  
 if room\_id < num\_classes or room\_id >= num\_classes + len(special\_rooms):  
 fitness += 1  
 elif room\_id in used\_special\_rooms:  
 fitness += 1  
 else:  
 used\_special\_rooms.add(room\_id)  
 else:  
 if room\_id >= num\_classes:  
 fitness += 1  
  
 return fitness  
  
  
# Схрещування двох розкладів  
def crossover(schedule1, schedule2):  
 child\_schedule = []  
 for class\_id in range(num\_classes):  
 child\_class\_schedule = []  
 for lesson, lesson\_type, teacher\_id, room\_id in schedule1[class\_id]:  
 if random.random() < 0.5:  
 child\_class\_schedule.append((lesson, lesson\_type, teacher\_id, room\_id))  
 else:  
 child\_class\_schedule.append((lesson, lesson\_type, teacher\_id, room\_id))  
 child\_schedule.append(child\_class\_schedule)  
 return child\_schedule  
  
  
# Мутація розкладу  
def mutate(schedule):  
 mutated\_schedule = []  
 for class\_id in range(num\_classes):  
 mutated\_class\_schedule = []  
 for lesson, lesson\_type, teacher\_id, room\_id in schedule[class\_id]:  
 if random.random() < 0.1: # ймовірність мутації  
 lesson\_type = random.randint(0, num\_lesson\_types - 1)  
 teacher\_ids = random.sample(range(num\_teachers), num\_teachers)  
 room\_id = random.randint(0, num\_classes - 1)  
 mutated\_class\_schedule.append((lesson, lesson\_type, teacher\_ids[class\_id], room\_id))  
 else:  
 mutated\_class\_schedule.append((lesson, lesson\_type, teacher\_id, room\_id))  
 mutated\_schedule.append(mutated\_class\_schedule)  
 return mutated\_schedule  
  
  
# Генетичний алгоритм  
def genetic\_algorithm(population\_size, num\_generations):  
 population = create\_initial\_population(population\_size)  
  
 for generation in range(num\_generations):  
 fitness\_scores = [evaluate\_fitness(schedule) for schedule in population]  
  
 sorted\_population = [x for \_, x in sorted(zip(fitness\_scores, population), key=lambda pair: pair[0])]  
 selected\_population = sorted\_population[:population\_size // 2]  
  
 new\_population = []  
  
 # Схрещування і мутація для створення наступної популяції  
 while len(new\_population) < population\_size:  
 parent1, parent2 = random.sample(selected\_population, 2)  
 child = crossover(parent1, parent2)  
 mutated\_child = mutate(child)  
 new\_population.append(mutated\_child)  
  
 population = new\_population  
  
 best\_schedule = max(population, key=evaluate\_fitness)  
 return best\_schedule  
  
  
# Параметри генетичного алгоритму  
population\_size = 50  
num\_generations = 100  
  
# Запуск генетичного алгоритму  
best\_schedule = genetic\_algorithm(population\_size, num\_generations)  
  
# Виведення оптимального розкладу  
print("Оптимальний розклад:")  
  
for day in range(num\_days):  
 print(f"День {day + 1}:")  
 print("Клас 1".ljust(20) + " " \* 4 + "Клас 2")  
 class\_1\_schedule = best\_schedule[0][day \* (num\_lessons\_per\_week // num\_days):(day + 1) \* (num\_lessons\_per\_week // num\_days)]  
 class\_2\_schedule = best\_schedule[1][day \* (num\_lessons\_per\_week // num\_days):(day + 1) \* (num\_lessons\_per\_week // num\_days)]  
  
 for i in range(num\_lessons\_per\_week // num\_days):  
 class\_1\_lesson = class\_1\_schedule[i]  
 class\_2\_lesson = class\_2\_schedule[i]  
  
 class\_1\_lesson\_str = f"Урок {i+1} | Предмет - {lesson\_types[class\_1\_lesson[1]]} | " \  
 f"Вчитель - {teachers[class\_1\_lesson[2]]} | " \  
 f"Приміщення - {special\_rooms[class\_1\_lesson[3]] if class\_1\_lesson[1] >= num\_lesson\_types - len(special\_rooms) else 'Класна кімната'}"  
  
 class\_2\_lesson\_str = f"Урок {i+1} | Предмет - {lesson\_types[class\_2\_lesson[1]]} | " \  
 f"Вчитель - {teachers[class\_2\_lesson[2]]} | " \  
 f"Приміщення - {special\_rooms[class\_2\_lesson[3]] if class\_2\_lesson[1] >= num\_lesson\_types - len(special\_rooms) else 'Класна кімната'}"  
  
 print(class\_1\_lesson\_str.ljust(20) + " " \* 4 + class\_2\_lesson\_str)  
  
 print("--------------------")  
  
fitness = evaluate\_fitness(best\_schedule)  
print(f"Оцінка якості розкладу: {fitness}")

**Результат виконання:**



**Висновок:**

На лабораторній роботі я ознайомилась з генетичним алгоритмом та реалізувала його на практиці для розв’язання задачі оптимізації на прикладі складання розкладу занять.