ЛАБОРАТОРНА РОБОТА № 7

ДОСЛІДЖЕННЯ МУРАШИНИХ АЛГОРИТМІВ

Мета роботи: *використовуючи спеціалізовані бібліотеки та мову програмування Python навчитися дослідити метод мурашиних колоній.*

1. ТЕОРЕТИЧНІ ВІДОМОСТІ

Основні теоретичні відомості подані на лекціях. Також доцільно вивчити матеріал поданий в літературі:

Додатково деякі теоретичні відомості подано у кожному завданні окремо.

Можна використовувати Google Colab або Jupiter Notebook.

1. Принципи поведінки мурах

Мурахи належать до так званих «соціальних комах», тобто комах, що живуть в межах певного колективу — сім'ї або колонії. На Землі біля двох відсотків комах є «соціальними», половина з яких припадає на мурах. Поведінка мурах під час транспортування їжі, обминання перешкод, побудови мурашника тощо наближається до теоретично оптимальної. Основу «соціальної» поведінки мурах складає самоорганізація — сукупність динамічних механізмів, за допомогою яких система досягає глобальної мети в результаті взаємодії елементів на низькому рівні. Принциповою особливістю такої низькорівневої взаємодії є використання елементами системи лише локальної інформації, без будь-якого централізованого управління та звернення до глобального образу, який репрезентує систему у зовнішньому світі. Самоорганізація є результатом взаємодії таких чотирьох компонентів: 1) випадковість; 2) додатний зворотний зв'язок; 3) від'ємний зворотний зв'язок; 4) багатократність взаємодій.

Мурахи використовують два способи передачі інформації:

прямий — обмін харчами, мандибулярний, візуальний та хімічний контакти тощо;

непрямий — стигмержі (stigmergy). Стигмержі — це рознесений у часовому просторі тип взаємодії між елементами системи, коли один суб'єкт взаємодії змінює деяку частину оточуючого середовища, а решта використовує інформацію про її стан пізніше, коли знаходяться в її околі. Біологічно, стигмержі здійснюється через феромон (pheromone) — спеціальний секрет, який відкладається як слід під час руху мурахи. Чим вища концентрація феромону на стежці, тим більше мурах буде рухатись по

ній. Феромон з часом випаровується, що дозволяє мурахам адаптувати свою поведінку до зміни оточуючого середовища. Розподіл феромону по простору пересування мурах є своєрідною глобальною пам'яттю, яка динамічно змінюється. Кожна мураха може сприймати та змінювати лише локальну чарунку цієї глобальної пам'яті мурашника.

Розглянемо, як колективна поведінка біологічних мурах забезпечує знаходження найкоротшого шляху до їжі на прикладі експериментів на асиметричному мості [6]. Асиметричний міст (рис. 1) з'єднує гніздо мурах з джерелом їжі двома гілками різної довжини. Експерименти [6] проводилися за такою схемою: 1) будувався міст A-B-C-D; 2) відчинялися дверцята в точці A; 3) фіксувалась кількість мурах, які обрали довгий (A-C-D) та короткий (A-B-D) шляхи. На початку експериментів мурахи обирали обидві гілки з однаковою ймовірністю тому, що на мості не було феромонів. Через деякий час майже усі мурахи пересувались найкоротшим маршрутом A-B-D, що пояснюється таким чином. Мурахи, які обрали короткий маршрут A-B-D-B-A, скоріше поверталися з їжею в гніздо, залишаючи феромонні сліди на короткій гілці мосту.

При наступному виборі маршруту, мурахи віддавали перевагу короткій гілці мосту, тому що на ній вища концентрація феромонів. Таким чином, феромон швидше накопичується на гільці A-B-D, що підштовхує мурах до вибору найкоротшого маршруту.

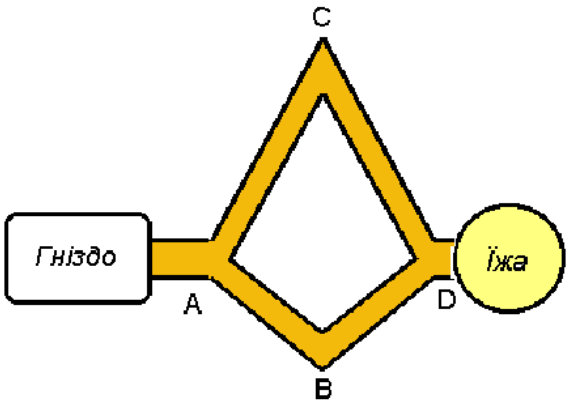


Рис. 1. Асиметричний міст

2. Основні поняття

Мурашині алгоритми серйозно досліджуються європейськими вченими із середини 1990-х років. На сьогоднішній день вже отримано гарні результати для оптимізації таких складних комбінаторних задач,як задача комівояжера, задача оптимізації маршрутів вантажівок, задача розфарбування графа, квадратична задача про призначення, задача оптимізації сіткових графіків, задача календарного планування і т. ін. Особливо ефективні мурашині алгоритми при динамічній оптимізації

процесів у розподілених нестаціонарних системах, наприклад, графіків у телекомунікаційних мережах.

Метод мурашиних колоній заснований на взаємодії декількох *мурах* (програмних агентів, що є членами великої колонії) і використовується для вирішення різних оптимізаційних задач. Агенти спільно вирішують проблему й допомагають іншим агентам у подальшій оптимізації розв’язку.

Методи мурахи (Ant algorithms), або оптимізація за принципом мурашиної колонії, мають специфічні особливості, властиві мурахам, і використовують їх для орієнтації у фізичному просторі. Природа пропонує різні методики для оптимізації деяких процесів. Методи мурашиних колоній особливо цікаві тому, що їх можна використовувати для вирішення не тільки статичних, але й динамічних задач, наприклад, задач маршрутизації в мінливих мережах.

Базова ідея методу мурашиних колоній полягає у вирішенні оптимізаційної задачі шляхом застосування непрямого зв'язку між автономними агентами.

У методі мурашиних колоній передбачається, що навколишнє середовище являє собою закриту двовимірну мережу – групу вузлів, з'єднаних за допомогою граней. Кожна грань має вагу, що позначається як відстань між двома вузлами, з'єднаними нею. Граф – двонаправлений, тому агент може подорожувати по грані в будь-якому напрямку.

Агент забезпечується набором простих правил, які дозволяють йому вибирати шлях у графі. Він підтримує список табу *tList* – список вузлів, які він вже відвідав. Таким чином, агент повинен проходити через кожний вузол тільки один раз.

Вузли в списку “поточної подорожі” *tList* розташовуються в тому порядку, у якому агент відвідував їх. Пізніше список використовується для визначення довжини шляху між вузлами.

Справжня мураха під час переміщення по шляху залишає за собою деяку кількість *феромону*. У методі мурашиних колоній агент залишає феромон на гранях мережі після завершення подорожі.

3. Мурашиний підхід до розв’язання задачі комівояжера

Задача комівояжера полягає у виборі найкоротшого замкнутого шляху, що проходить через усі міста рівно один раз. Розглянемо, як реалізувати чотири основних компоненти самоорганізаційної поведінки мурах під час оптимізації маршруту комівояжера.

Багатократність взаємодії реалізується ітераційним пошуком маршруту комівояжера одночасного декількома мурахами.

Додатний зворотний зв’язок реалізується як імітація природної поведінки мурах типу «залишання слідів – пересування по слідах». Чим більше слідів залишено на стежці — ребрі графу, тим більше мурах буде рухатись по ній. При цьому на стежці з’являються нові сліди, які

приваблюють додаткових мурах. Для задачі комівояжера додатний зворотний зв'язок реалізується таким стохастичним правилом: «ймовірність включення ребра графу в маршрут мурахи пропорційна кількості феромону на ній».

Використання цього стохастичного правила забезпечує реалізацію і іншого компоненту поведінки мурах – випадковості. Кількість феромону, який відкладає мураха на ребрі графа, є зворотно пропорційною величиною до довжини відповідного маршруту комівояжера. Чим коротший маршрут комівояжера знайшла мураха, тим більше феромону буде відкладене на відповідних ребрах графу.

Використання лише додатного зворотного зв'язку призводить до передчасної збіжності алгоритму, тобто до випадку, коли усі мурахи рухаються одним і тим же субоптиальним маршрутом. Для уникнення цього використовується від'ємний зворотний зв'язок – випаровування феромону. Час випаровування феромону не повинен бути дуже великим, бо при цьому виникає загроза збігання маршрутів усіх мурах до одного субоптимального розв'язку. З іншого боку, час випаровування не повинен бути і малим, щоб не призвести до некооперативної поведінки мурах через втрату пам'яті колонії.

Перехід мурахи з міста i в місто j на ітерації t алгоритму залежить від трьох складових: табу-списка, видимості та віртуального сліду феромону.

Табу-список (tList) — це перелік міст, які вже відвідані мурахою і заходити в які ще раз заборонено. Табу-список збільшується зі здійсненням маршруту та заповнюється нулями на початку кожної ітерації алгоритму. Позначимо через J_i^k список міст, які ще потрібно відвідати мурасі k, що перебуває у місті i. Зрозуміло, що об'єднання цих списків дає множину усіх міст з маршруту комівояжера.

Видимість – це величина обернена до відстані: $\eta_{ij} = 1/D_{ij}$, де D_{ij} — відстань між містами i та j. Видимість базується тільки на локальній інформації і являє собою «евристичну бажаність» вибору міста j, під час перебування у місті i. Чим ближче міста i та j, тим більше бажання відвідати їх.

Віртуальний слід феромону на ребрі (i – j) являє собою «бажаність, підкріплену досвідом» переходу в місто i з міста j. Інформація, яку несе слід феромону, змінюється підчас оптимізації і відображає набутий мурахами досвід. Кількість віртуального феромону на ребрі (i – j) на t-й ітерації алгоритму позначимо як $\tau_{ij}(t)$.

Ймовірність переходу k-ї мурахи з міста i у місто j на t-й ітерації розраховується за випадково-пропорційним правилом:

$$\begin{cases} P_{ij}^k(t) = \dfrac{\left[\tau_{ij}(t)\right]^\alpha \left[\eta_{ij}\right]^\beta}{\sum\limits_{l \in J_i^k} \left[\tau_{il}(t)\right]^\alpha \left[\eta_{il}\right]^\beta}, & \text{якщо } j \in J_i^k; \\ P_{ij}^k(t) = 0, & \text{якщо } j \notin J_i^k. \end{cases}$$

де α і β – два регульовані параметри, які є вагами інтенсивності сліду феромону та видимості. Якщо $\alpha = 0$, то найвірогіднішим буде перехід у найближчі міста. У класичній теорії оптимізації це відповідає, так званому,

пожадливому алгоритму. Якщо β = 0, тоді працює лише феромоне підсилення, що призводить до швидкого завершення роботи алгоритму через збігання маршрутів усіх мурах до одного субоптимального розв'язку.

Після завершення маршруту кожна мураха k відкладає на ребро (i – j) таку кількість феромону:

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L^{k}(t)}, & \text{якщо } (i,j) \in T^{k}(t) \\ 0, & \text{якщо } (i,j) \notin T^{k}(t) \end{cases},$$

де $T^{k}(t)$ – маршрут, зроблений мурахою k на ітерації t; $L^{k}(t)$ — його довжина; Q — регульований параметр, значення якого обирають одного порядку з довжиною оптимального маршруту.

Для забезпечення можливості експлуатації простору рішень потрібно забезпечити випаровування феромону — зменшення кількості відкладеного на попередніх ітераціях феромону. Інтенсивність випаровування феромону задається за допомогою коефіцієнта випаровування $p \in [0, 1]$. Кінцеве правило оновлення феромону, яке стосується всіх ребер, приймає вигляд

$$\tau_{ij}(t+1) \leftarrow (1-p)\,\tau_{ij}(t) + \Delta\tau_{ij}(t),$$

де $\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t)$; m – кількість мурах в колонії.

На початку оптимізації кількість феромону на ребрах приймається за малу додатну константу τ_0. Загальна кількість мурах в колонії приймається постійною на весь час розв'язання задачі. Забагато мурах призводить до швидкого підсилення субоптимальних маршрутів. Коли ж мурах замало, виникає небезпека втрати кооперативної поведінки через швидке випаровування феромону. Зазвичай кількість мурах приймають рівною числу міст — кожна мураха починає маршрут з окремого міста.

4. Послідовність виконання методу

Метод мурашиних колоній виконується в наступній послідовності кроків.

Крок 1. Задати параметри методу: α – коефіцієнт, що визначає відносну значимість шляху (кількість феромона на шляху); β – параметр, що означає пріоритет відстані над кількістю феромона; ρ – коефіцієнт кількості феромона, що агент залишає на шляху, де (1–ρ) показує коефіцієнт випару феромона на шляху після його завершення; Q– константу, що відноситься до кількості феромона, яку було залишено на шляху.

Крок 2. Ініціалізація методу. Створення популяції агентів. Після створення популяція агентів нарівно розподіляється по вузлах мережі. Необхідно рівномірний розподіл агентів між вузлами, щоб всі вузли мали однакові шанси стати відправною точкою. Якщо всі агенти почнуть рух з

однієї точки, це буде означати, що дана точка вважається оптимальною для старту, а насправді вона такою може і не бути.

Крок 3. Рух агентів. Якщо агент ще не закінчив шлях, тобто не відвідав всі вузли мережі, для визначення наступної грані шляху використовується формула:

$$P = \frac{\tau_{ru}(t)^{a} \cdot \eta_{ru}(t)^{\beta}}{\sum\limits_{k \in j} \tau_{ru}(t)^{a} \cdot \eta_{ru}(t)^{\beta}},$$

де J – множина граней, ще не відвіданих агентом; $\tau_{ru}(t)$ – інтенсивність феромона на грані між вузлами r і u, які утворять k-y грань, у момент часу t; $\eta_{ru}(t)$ – функція, що представляє собою зворотну величину відстані грані.

Агент подорожує тільки по вузлах, які ще не були відвідані ним,тобто по вузлах, яких не має у списку табу *tList*. Тому ймовірність розраховується тільки для граней, які ведуть до ще не відвіданих вузлів.

Крок 4. Пройдений агентом шлях відображається, коли агент відвідає всі вузли мережі. Цикли заборонені, оскільки в метод включений список табу *tList*. Після завершення може бути розрахована *довжина шляху*. Вона дорівнює сумі всіх граней, по яких подорожував агент. *Кількість феромону*, що було залишено на кожній грані шляху *i*-го агенту, визначається за формулою:

$$\Delta\tau_{ru}^{i}(t) = \frac{Q}{L^{i}(t)},$$

де $L^{i}(t)$ – довжина шляху *i*-го агенту.

Результат є засобом вимірювання шляху: короткий шлях характеризується високою концентрацією феромону, а більш довгий шлях – більш низкою. Потім отриманий результат використовується для збільшення кількості феромону уздовж кожної грані пройденого *i*-им агентом шляху:

$$\tau_{ru}(t+1) = \tau_{ru}(t) + (\Delta\tau_{ru}^{i}(t) \cdot \rho),$$

де r, u – вузли, що утворюють грані, які відвідав *i*-ий агент.

Дана формула застосовується до всього шляху, при цьому кожна грань позначається феромоном пропорційно довжині шляху. Тому варто дочекатися, поки агент закінчить подорож і тільки потім оновити рівні феромону, у противному випадку дійсна довжина шляху залишиться невідомою. Константа ρ приймає значення між 0 і 1.

$$\tau_{ru}(t) = \tau_{ru}(t) \cdot (1-\rho)$$

Тому для випаровування феромону використовується зворотний коефіцієнт відновлення шляху $(1-\rho)$.

Крок 5. Перевірка на досягнення оптимального результату. Перевірка може виконуватися для постійної кількості шляхів або до моменту, коли протягом декількох запусків не було отримано повторних змін у виборі найкращого шляху. Якщо перевірка дала позитивний результат, то

відбувається закінчення роботи методу (перехід до кроку 7), у противному випадку – перехід до кроку 6.

Крок 6. Повторний запуск. Після того як шлях агента завершений, грані оновлені відповідно до довжини шляху, й відбулося випаровування феромону на всіх гранях, метод виконується повторно. Список табу очищується, і довжина шляху прирівнюється нулю. Після цього виконується перехід до кроку 3.

Крок 7. Кінець. Визначається кращий шлях, що і є розв'язком.

2. ЗАВДАННЯ НА ЛАБОРАТОРНУ РОБОТУ ТА МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО ЙОГО ВИКОНАННЯ

Завдання 2.1. Дослідження мурашиного алгоритму на прикладі рішення задачі комівояжера

У файлі Відстані між обласними центрами України.docx міститься таблиця відстаней між обласними центрами України. Міста пронумеровані по алфавіту. ***Ваш номер за журналом групи (або номер за табличкою у рейтингу) повинен відповідати місту з якого ви будете починати виїзд.*** Наприклад: за рейтинговою таблицею - № 6 Драк Тарас Сергійович починає з № 6. Івано-Франківськ. Виїзд починається і закінчується в цьому місті.

Використовуючи мову Python, розробити програму, що реалізує метод мурашиних колоній, для рішення задачі комівояжера, який їздить по містах України.

РЕКОМЕНДАЦІЇ ДО ВИКОНАННЯ

1. Ознайомитися з літературою та основними теоретичними відомостями за темою роботи.

2. Розробити програму, що реалізує метод мурашиних колоній. При програмній реалізації методу дотримуватися вимог, наведених в теоретичних відомостях.

2.1. Для гнучкості програмного забезпечення всі параметри методу (наприклад, кількість агентів, коефіцієнт випаровування і т. ін.) зберігати в змінних або константах.

2.2. Для реалізації агентів, що моделюють поведінку мурах,використовувати структуру, що має наступні поля:

– поточна позиція, в якій знаходиться агент;

– наступна позиція;

– список табу, в якому зберігаються ті пункти, в яких вже побував агент;

– кількість пунктів, що вже відвідав агент;

– масив подорожі, де зберігається послідовність пунктів, в яких побував агент;

– довжина шляху, що пройшов агент, розраховується після закінчення подорожі.

2.3. Виділити основні етапи методу в окремі функції:

– ініціалізація методу;

– моделювання переміщення агентів;

– вибір наступного пункту;

– оновлення шляхів;

– перезавантаження агентів.

Також необхідно реалізувати окрему функцію розрахунку довжини шляху.

2.4. Передбачити можливість наглядного (графічного) відображення змін у поточних результатах роботи методу.

3. Виконати тестування розробленого програмного забезпечення за допомогою вирішення конкретних прикладів задачі комівояжера. Задачі (не менше трьох) для виконання тестування програми сформулювати самостійно. Вибір тестових задач обґрунтувати.

4. Порівняти одержані результати вирішення різних прикладів задачі комівояжера. Результати порівняльного аналізу звести до таблиці, попередньо розробивши систему критеріїв порівняння результатів вирішення задачі комівояжера.

5. Оформити звіт з роботи.

Приклади реалізації алгоритму на інших мовах.

Для MATLAB на рис. 2 подано базовий алгоритм

```
<Ввод матрицы расстояний D>
<Инициализация параметров алгоритма α (alpha),
β (beta), e, Q, τ0 (tau0)>
m=n % Количество муравьев равно числу городов
For i=1:n
   For j=1:n % Для каждого ребра
       If i~=j
       eta(i,j)=1/D(i,j); % Видимость
       tau(i,j)= tau0; % Феромон
       Else tau(i,j)=0;
% Переход из одного города в тот же самый
% невозможен
       End
   End
End
For k=1:m
<Разместить муравья k в случайно выбранный
город>
End
<Выбрать условно-кратчайший маршрут T+ и
рассчитать его длину L+>
```

```
% Основной цикл
For t=1:tmax
% tmax - количество итераций алгоритма
       For k=1:m % Для каждого муравья
<Построить маршрут Tk(t) по правилу (1) и
рассчитать длину Lk(t)>
       End
If < "Лучшее решение найдено?" >
<Обновить T+ и L+ >
End
For i=1:n
For j=1:n % Для каждого ребра
<Обновить следы феромона по правилам (2) и
(3).>
End
End
End
<Вывести кратчайший маршрут T+ и его длину L+>
```

Рис.2

Аналогічний алгоритм виконання програми подано на рис.3

```
Ініціалізація параметрів алгоритму α, β, e, Q, τ0.
m = n          {Кількість мурах дорівнює кількості міст}
For i = 1 to n
   For j = 1 to n       {Для кожного ребра}
      If i <> j
            η(i, j) =1/D(i, j)    {Видимість}
            τ(i, j) = τ0     {Феромон}
      Else t(i, j) = 0
      End
   End
End
For k = 1 to m
  Розмістити мураху k у випадково обране місто.
End
Обрати умовно найкоротший маршрут T+ та обчислити його довжину L+.
{Головна програма}
For t = 1 to tmax          {Кількість ітерацій алгоритму}
   For k = 1 to m      {Для кожної мурахи}
      Побудувати маршрут Tk(t) за правилом (1) та обчислити його довжину Lk(t).
   End
   If «Кращий розв'язок знайдено»
     Оновити T+ та L+.
   End
   For i = 1 to n
      For j = 1 to n        {Для кожного ребра}
         Оновити сліди феромону за правилом (2).
      End
   End
End
Вивести найкоротший маршрут T+ та його довжину L+.
```

Рис.3

При тестуванні для формування матриці відстаней використовуйте таблицю відстаней між містами України що подана у відповідному документі. У результатах вказуйте маршрут відповідно до вашого варіанту.

Зміст звіту

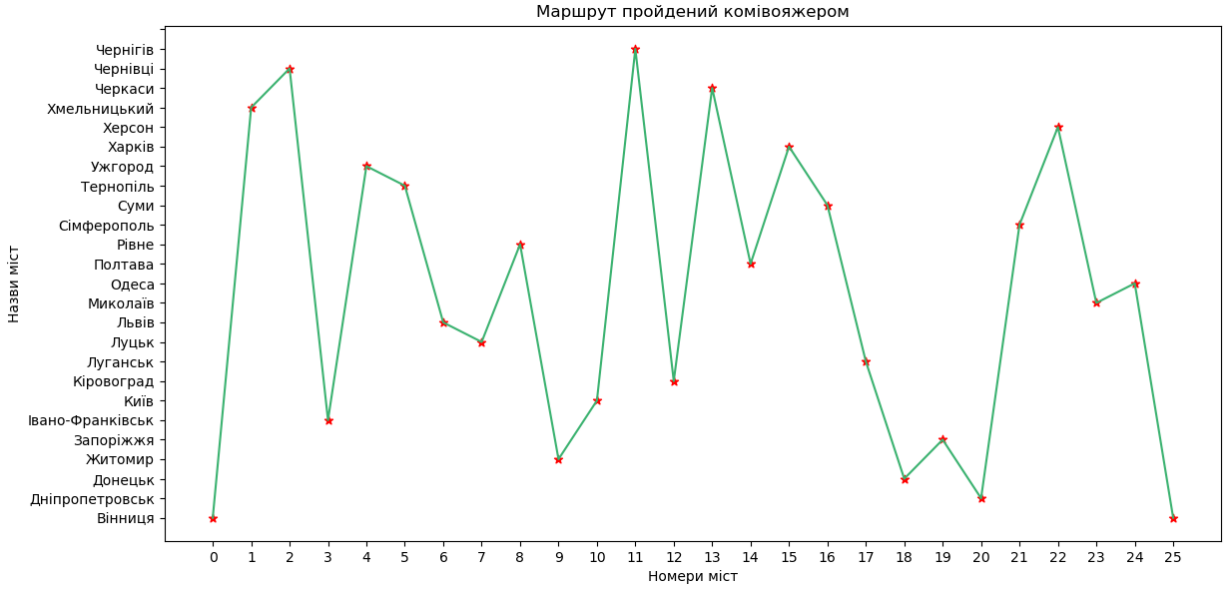
Тема та мета роботи.

Номер групи, ПІБ студента, номер варіанта.

Текст розробленого програмного забезпечення з коментарями.

Результати роботи програмного забезпечення (таблиця порівняльного аналізу та графіки з поясненнями).

Графік найкоротшого маршруту з початком у місті згідно вашого варіанту подаєте в кінці. Наприклад:



Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Збережіть код робочої програми під назвою LR_7_task_1.py
Код програми, результати у вигляді графіків занесіть у звіт.
Зробіть висновок

Коди комітити на GitHab. У кожному звіті повинно бути посилання на GitHab.

Назвіть бланк звіту СШІ-ЛР-7-NNN-XXXXX.doc
де NNN – позначення групи
XXXXX – позначення прізвища студента.

Переконвертуйте файл звіту в СШІ-ЛР-7-NNN-XXXXX.pdf
Надішліть чи представте звіт викладачу.