Розглянемо подробиці роботи кожної функції:

* read\_matrix\_from\_csv()

Функція зчитує довжину першого рядка і повертає курсор в початкове положення. Потім зчитує кожен елемент кожного рядка матриці і записує його у вкладений список.

* write\_matrix\_to\_csv()

Функція приймає вкладений список (матрицю). Кожен рядок матриці перетворює на тип даних str і звписує у файл.

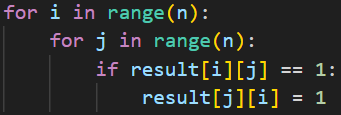
* reflexive\_closing()

Функція приймає матрицю та створює нову матрицю (вкладений список) рефлексивного замикання заданої матриці. А саме копіює початкову матрицю, зчитує кількість рядків та стовпців і кожному елементові, який має однакові номери стовпця і рядка, присвоює значення 1.



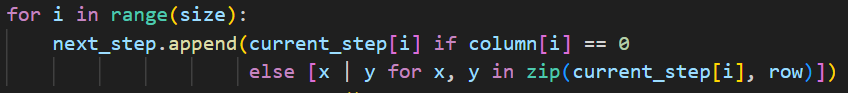
* symmetrical\_closing()

Функція приймає матрицю та створює нову матрицю (вкладений список) симетричного замикання заданої матриці. А саме копіює початкову матрицю, рахує кількість рядків і стовпців. Перебігає по всіх елементах матриці та перевіряє чи елемент матриці = 1, якщо так, то присвоює значення 1 елементові, симетричному відносно діагоналі.



* transitive\_closing()

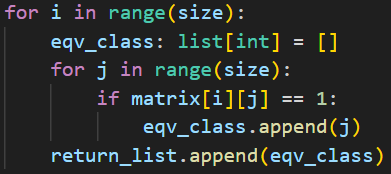
Функція приймає матрицю та створює нову матрицю (вкладений список) транзитивного замикання заданої матриці за алгоритмом Воршала. А саме рахує кількість рядків і стовпців, копіює початкову матрицю. Перебігає по всіх ітераціях алгоритму, присвоює змінній column список значень колони, а змінній row список значень рядка на цій ітерації. Перебігає по всіх елементах колони і перевіряє чи він = 0. Якщо так, то рядку вихідної матриці присвоюється початковий рядок матриці на цій ітерації. Якщо ні, то функція перебігає по відповідних елементах рядків і стовпців на цій ітерації і в рядок нової матриці записує диз’юнкцію цих елементів.



Далі для наступної ітерації перезаписується матриця на нову.

* Equivalence\_classes()

Функція приймає матрицю та виводить класи еквівалентності для цієї матриці. А саме рахує кількість рядків і стовпців матриці. Оскільки кількість класів еквівалентності рівна кількості рядків та стовпців матриці, то ми стільки разів і виконуємо програму. Далі функція перебігає по елементах кожного рядка і якщо якийсь елемент = 1, то в клас під номером рядка цього номера ми додаємо значення номера стовпця цього номера.



* check\_transitivity()

Функція перевіряє чи матриця транзитивна шляхом множення цієї матриці саму на себе. Алгоритм множення відбувається наступним чином: рахує кількість рядків та стовпців матриці та створює матрицю таких же розмірів, що й початкова, тільки заповнена нулями. Далі перебігає по кожному стовпцю матриці і для кожного стовпця перебігає по кожному рядку і повертає кон’юнкцію відповідних елементів цього рядка і стопця. Якщо хоча б одна кон’юнкція повертає 1, то в результуючу матицю з індексами порівнюваних елементів стовпця і рядка. Якщо множення цієї матриці саму на себе рівна початковій матриці, то матриця транзитивна.

