# Опис алгоритмів

Перелік всіх основних змінних та їхнє призначення наведено в таблиці 3.1.

Таблиця 3.1 – Основні змінні та їхні призначення

|  |  |
| --- | --- |
| Змінна | Призначення |
|  | Точність з якою необхідно знайти корені розв’язків  () |
| Matrix | Матриця системи |
| b | Стовпець вільних членів |
| norm | Норма поточного розв’язку системи |
| flag | Ознака сходимості методу (false – метод не сходиться,  true – метод сходиться) |

## Загальний алгоритм

1. ПОЧАТОК
2. Зчитати розмірність системи.
3. Зчитати матрицю системи та стовпець вільних членів:
   1. Зчитати матрицю системи:
      1. Цикл проходу по всіх рядках матриці системи (:
         1. Цикл проходу всіх стовпцях матриці системи   
            (:
            1. ЯКЩО поточний елемент матриці – вірно записане число, ТО записати його в відповідну комірку *matrix*. ІНАКШЕ видати повідомлення про помилку та перейти до пункту 8.
   2. Зчитати стовпець вільних членів:
      1. Цикл проходу по всіх елементах стовпця вільного членів:
         1. Якщо поточний елемент стовпцю вільних членів – вірно записане число, ТО записати його в відповідну комірку *free*. ІНАКШЕ видати повідомлення про помилку та перейти до пункту 8.
4. ЯКЩО обраний метод Якобі, ТО обробити дані згідно алгоритму методу Якобі (пункт 3.2)
5. ЯКЩО обраний метод Гауса-Зейделя, ТО обробити дані згідно алгоритму методу Гауса-Зейделя (пункт 3.3)
6. ЯКЩО обраний метод градієнтного спуску, ТО обробити дані згідно алгоритму методу градієнтного спуску (пункт 3.4).
7. ЯКЩО обраний метод сходиться для вхідних даних, ТО:
   1. ЯКЩО обрана система на дві невідомих, ТО побудувати та вивести графік системи.
   2. Вивести рішення системи.
   3. Записати систему та її рішення у файл.
8. КІНЕЦЬ

## Алгоритм методу Якобі

1. ПОЧАТОК
2. Обчислити визначник матриці (*Matrix*) (**det**):
   1. ЯКЩО матриця має одиничну розмірність, ТО позначити як визначник даної матриці єдиний елемент цієї матриці. ІНАКШЕ обчислити визначник розкладом за першим стовпцем:
      1. Ініціалізація визначника поточної матриці (*Matrix*) *detRes* = 0.
      2. ЦИКЛ проходу по всіх елементах першого стовпця матриці (*Matrix*) :
         1. Обчислити алгебраїчне доповнення поточного елементу :
            1. Утворити доповнювальний мінор для поточного елементу (**marixCut**):

Ініціалізація нової матриці (*newMatrix*) розмірності ().

ЦИКЛ проходу по всіх елементах матриці (*Matrix*):

ЯКЩО поточний елемент не знаходиться на першому стовпці АБО   
(*i-тому*) рядку, ТО добавити його в нову матрицю (*newMatrix*).

* + - 1. Знайти визначник доповнювального мінору (тобто знайти визначник матриці *newMatrix*) (**det**) (аналогічно пункту 2).
      2. Добавити до *detRes* значення виразу  
          .

1. Присвоїти *flag* = true.
2. ЯКЩО визначник не дорівнює нулю, ТО перевірити умови сходимості методу Якобі (**isSolution**):
   1. Приведення матриці до канонічного (діагонального) вигляду (**stableMatrix**):
      1. Ініціалізація масиву *element* (*element*[*i*] = *false*, *i* =1, 2, …, ).
      2. ЦИКЛ проходу по всіх рядках матриці (*Matrix*):
         1. Знайти номер індексу найбільшого за абсолютним значенням елементу поточного рядка (*index*).
         2. ЯКЩО такий індекс (*index*), вже був знайдений (*element[index]* = true), ТО позначити, що для даної матриці (*Matrix*) метод не сходиться (присвоїти *flag* = false) та перейти до пункту 6, ІНАКШЕ присвоїти *element[index]* = true.
      3. Ініціалізація нової матриці (*newMatrix*) такої ж розмірності, що й поточна матриця (*Matrix*).
      4. ЦИКЛ проходу по всіх рядках матриці (*Matrix*):
         1. Знайти номер індексу найбільшого за абсолютним значенням елементу поточного рядка (*index*) матриці (*Matrix*).
         2. Присвоїти поточний рядок матриці (*Matrix*) *index*-рядку матриці (*newMatrix*) (тобто *newMatrix*[*index*]= *Matrix*[*i*]).
      5. Присвоїти *Matrix* = *newMatrix*.
   2. ЦИКЛ проходу по всіх рядках матриці (*Matrix*):
      1. Знайти суму абсолютних значень всіх елементів поточного рядка (*sum*):
         1. Інціціалізація *sum* = 0.
         2. ЦИКЛ проходу по всіх елементах ( поточного рядка:
            1. Додати до *sum* абсолютне значення поточного ементу .
      2. Знайти найбільший за абсолютним значенням елемент поточного рядка (*max*):
         1. Інціціалізація .
         2. ЦИКЛ проходу по всіх елементах ( поточного рядка:
            1. ЯКЩО ТО .
      3. ЯКЩО *max(sum – max)*, ТО для даної матриці (*Matrix*) метод не сходиться (присвоїти *flag* = false).

ІНАКШЕ перейти до пункту 6.

1. ЯКЩО метод сходиться (), ТО розв’язати СЛАР методом Якобі (**Jacobi**):
   1. Задати попереднє наближення розв’язку (.
   2. Присвоїти *norm* = .
   3. ПОКИ :
      1. Обчислити наступне наближення (:
         1. ЦИКЛ проходу по всіх елементах поточного наближення  
            ():
            1. Присвоїти .
            2. ЦИКЛ проходу по всіх елементах переднього наближення ():

ЯКЩО , ТО присвоїти  
 .

* + - * 1. Присвоїти .
    1. Обчислити поточне значення норми розв’язку (*norm*) (**normCalc**):
       1. Присвоїти *norm =* .
       2. ЦИКЛ проходу по всіх елементах поточного наближення :

ЯКЩО , ТО *norm =.*

* + 1. Присвоїти .

1. КІНЕЦЬ