# 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  () |
| positions | Масив координат стовпців |
| searchTree | Дерево станів системи |
| solutionStatus | Ознака вирішеності задачі (false –задачу не вирішено,  true – задачу вирішено) |

## Загальний алгоритм

1. ПОЧАТОК
2. Отримати інформацію про вибір методу вирішення, заданий користувачам
3. Зчитати поточне положення фігур на шаховій дошці
4. Перевірити правильність розстановки фігур solutionStatus відповідно до критерію вирішення задачі
   1. ЯКЩО фігури розставлені правильно (solutionStatus == true), ТО вивести відповідне повідомлення та завершити роботу програми
   2. ІНАКШЕ вирішити задачу програмно
5. Перевірити чи кожна фігура займає лише один рядок checkRows

5.1. ЯКЩО декілька фігур займають один рядок, ТО змінити їх положення так, щоб кожна королева займала окремий рядок (rowsInsert)

5.2. Записати координати фігур у вектор positions (matrixPositions -> positions)

6. Побудувати searchTree

6.1. Ініціалізувати rootNode: rootNode = currentState

6.2 Створити нащадків rootNode: rootNode.children – усі можливі перестановки 1-ї фігури

(queenIndex = 0, freePositions = range (0 -8) exept current) (generateChildren (queenIndex, freePositions) if queenIndex > 7 -> break)

6.3 ПОВТОРИТИ для кожного rootNode.child y rootNode

(freePositions = range (0 – 8))

Знайти позиції у наступному рядку (рядку під номером queenIndex), які знаходяться під атакою фігури поставленої у currentNode (із номером queenIndex) та виключити їх із можливих позицій;

queenIndex++;

generateChildren (queenIndex, freePositions)

1. Почати пошук рішення searchSolution за заданим алгоритмом
2. Зупинити роботу програми при знайденому рішенні
   1. ІНАКШЕ вивести повідомлення про помилку