**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №4**

**по дисциплине «Алгоритмы и структуры данных»**

**на тему «Исследование генетического алгоритма. Изучение различных кодировок генотипа»**

**Вариант – 11**

**Студент гр. 23Б16-пу**

**Крылов А. С.**

**Преподаватель**

**Дик А.Г.**

**Санкт-Петербург**

**2024 г.**

**Оглавление**

[Цель работы 2](#_Toc182550101)

[Описание задачи 2](#_Toc182550102)

[Теоретическая часть 2](#_Toc182550103)

[Основные шаги работы 4](#_Toc182550104)

[Блок схема программы 4](#_Toc182550105)

[Описание програмы: 4](#_Toc182550106)

[Описание функций: 5](#_Toc182550107)

[Описание переменных: 7](#_Toc182550108)

[Рекомендации пользователя 9](#_Toc182550109)

[Рекомендации программиста 9](#_Toc182550110)

[Исходный код программы 9](#_Toc182550111)

[Контрольный пример 10](#_Toc182550112)

[Вывод 13](#_Toc182550113)

# Цель работы

Целью лабораторной работы является исследование двух основных способов кодирования генотипа хромосом в генетическом алгоритме и проверка их эффективности.

# Описание задачи

1. Изучить особенности кодирования генетических алгоритмов
2. Написать программу поиска минимума функции, выбрав вариант тестовой функции из таблицы 1, согласна своему номеру в группе.
3. Протестировать программу на выбранной тестовой функции.

# Теоретическая часть

Для выполнения работы использован python с библиотеками tkinter и numpy.

Генетический алгоритм (ГА) — это метод оптимизации, основанный на принципах естественного отбора и эволюции. Основные этапы работы ГА:

* Инициализация популяции: Создание начальной группы решений (хромосом), представляющих возможные решения задачи.
* Оценка фитнеса: Вычисление целевой функции для каждого решения.
* Селекция: Выбор родителей для создания потомков.
* Кроссинговер: Комбинирование генов родителей для формирования новых решений.
* Мутация: Случайное изменение генов для поддержания разнообразия популяции.
* Создание новой популяции: Формирование нового поколения на основе результатов селекции, кроссинговера и мутации.
* Остановка: Алгоритм завершается, когда достигнуто максимальное число итераций или решение стабилизировалось.

## Основные шаги работы

* Инициализация: Задать параметры алгоритма (размер популяции, вероятность кроссинговера и мутации, количество поколений).
* Создание начальной популяции: Генерация случайных решений в заданных пределах.
* Оценка фитнес-функции: Вычисление значения целевой функции для каждой особи.
* Отбор родителей: Селективный отбор особей на основе их значений функции.
* Создание потомков: Выполнение кроссинговера и мутации для генерации нового поколения.
* Обновление популяции: Замена старой популяции новой.
* Проверка условий завершения: Если достигнуто максимальное число поколений или удовлетворительное решение, завершить работу, иначе вернуться к шагу 3.

## Блок схема программы

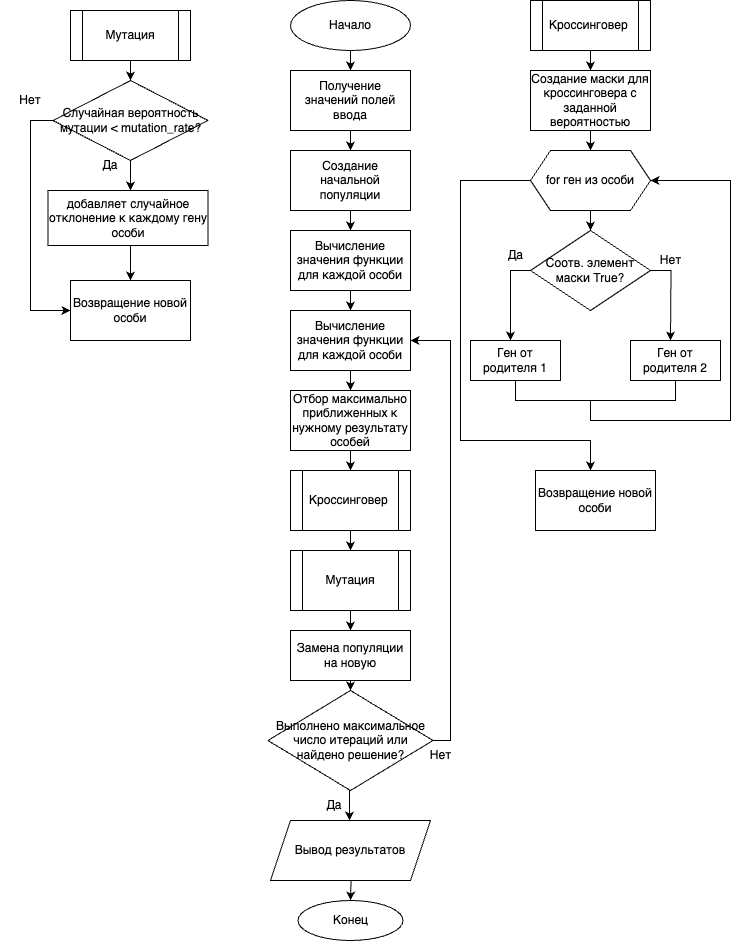


Рис 1. Блок-схема программы.

## Описание програмы:

Программная реализация написана на языке Python 3.11.9 с использованием следующих библиотек: tkinter, numpy.

## Описание функций:

Таблица 1. Описание функций

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Параметры | Описание параметров |
| ` function` | Вычисляет значение целевой функции \((x\_1 - 1)^2 + (x\_2 - 3)^2 + 4(x\_3 + 5)^2\). | `individual`: массив из трёх чисел `[x1, x2, x3]` | `individual`: переменные \(x\_1, x\_2, x\_3\), для которых вычисляется значение целевой функции. |
| `make\_population` | Создает начальную популяцию случайных особей в заданных пределах. | `size`: количество особей в популяции, `gene\_min`: минимальное значение гена, `gene\_max`: максимальное значение гена | `size`: число хромосом в популяции. `gene\_min`, `gene\_max`: границы диапазона возможных значений для каждого гена. |
| `choose\_beat\_scores` | Выбирает элитные особи с минимальными значениями целевой функции. | `population`: текущая популяция, `fitness\_scores`: массив значений целевой функции, `elite\_size`: число элитных особей | `population`: текущий набор хромосом. `fitness\_scores`: значения целевой функции для популяции. `elite\_size`: число особей, сохраняемых для следующего поколения. |
| `crossover` | Реализует кроссинговер между двумя родителями, создавая одну хромосому потомка. | `parent1`: хромосома родителя 1, `parent2`: хромосома родителя 2, `crossover\_rate`: вероятность выбора генов от первого родителя | `parent1`, `parent2`: хромосомы родителей. `crossover\_rate`: доля генов, которая будет передана от первого родителя. |
| `mutate` | Выполняет мутацию, изменяя один или несколько генов в хромосоме. | `individual`: хромосома для мутации, `mutation\_rate`: вероятность мутации, `gene\_min`: минимальное значение гена, `gene\_max`: максимальное значение гена | `individual`: хромосома для модификации. `mutation\_rate`: вероятность изменения гена. `gene\_min`, `gene\_max`: диапазон значений для генов после мутации. |
| `genetic\_algorithm\_mod` | Основная функция, реализующая генетический алгоритм. Возвращает лучшее найденное решение, его значение и данные по всем поколениям. | `population\_size`: размер популяции, `generations`: число поколений, `mutation\_rate`: вероятность мутации, `gene\_min`, `gene\_max`: диапазоны генов, `elite\_size`: число элитных особей, `crossover\_rate`: вероятность кроссинговера | `population\_size`: размер начальной популяции. `generations`: число итераций алгоритма. `mutation\_rate`: вероятность мутации. `gene\_min`, `gene\_max`: границы диапазона значений генов. `elite\_size`: число особей, переходящих в следующее поколение без изменений. `crossover\_rate`: вероятность передачи генов от одного родителя. |

## Описание переменных:

Таблица 2. Описание переменных

|  |  |  |
| --- | --- | --- |
| Название | Тип | Описание |
| individual | list[float] | Вектор переменных задачи, содержащий три элемента: \(x\_1, x\_2, x\_3\). |
| population | ndarray | Текущая популяция хромосом, где каждая строка — это одна особь (хромосома). |
| fitness\_scores | ndarray | Массив значений целевой функции для всех особей в популяции. |
| elite\_size | int | Количество элитных особей, сохраняемых в новое поколение. |
| parent1, parent2 | list[float] | Родительские хромосомы, участвующие в кроссинговере. |
| child | list[float] | Потомок, созданный в результате кроссинговера и мутации. |
| mutation\_rate | float | Вероятность изменения гена при мутации (от 0 до 1). |
| crossover\_rate | float | Вероятность выбора гена от первого родителя в процессе кроссинговера (от 0 до 1). |
| gene\_min, gene\_max | float | Границы диапазона значений для генов особей. |
| population\_size | int | Общее количество особей в популяции. |
| generations | int | Количество итераций, выполняемых алгоритмом (число поколений). |
| best\_solution | list[float] | Лучшее решение, найденное алгоритмом. |
| best\_fitness | float | Наименьшее значение целевой функции, соответствующее лучшему решению. |
| generation\_data | list[tuple] | История поколений: номер поколения, значение фитнес-функции и координаты лучшего решения. |

# Рекомендации пользователя

* Для запуска программы, убедитесь, что у вас установлен Python и необходимые библиотеки. Запустите код в среде разработки или командной для его выполнения.
* Запуск производится файлом laba4.py.
* Периодически проверяйте корректность данных перед использованием программы.

# Рекомендации программиста

* Поддерживайте актуальность используемых библиотек и версии Python для сохранения актуальности и работоспособности кода на современных системах.
* Периодически проводите тестирование на различных входных данных для обеспечения надежности и корректности программы.

# Исходный код программы

[**https://github.com/akryloff/spbu-algorithms-and-data-structures**](https://github.com/akryloff/spbu-algorithms-and-data-structures)

# Контрольный пример

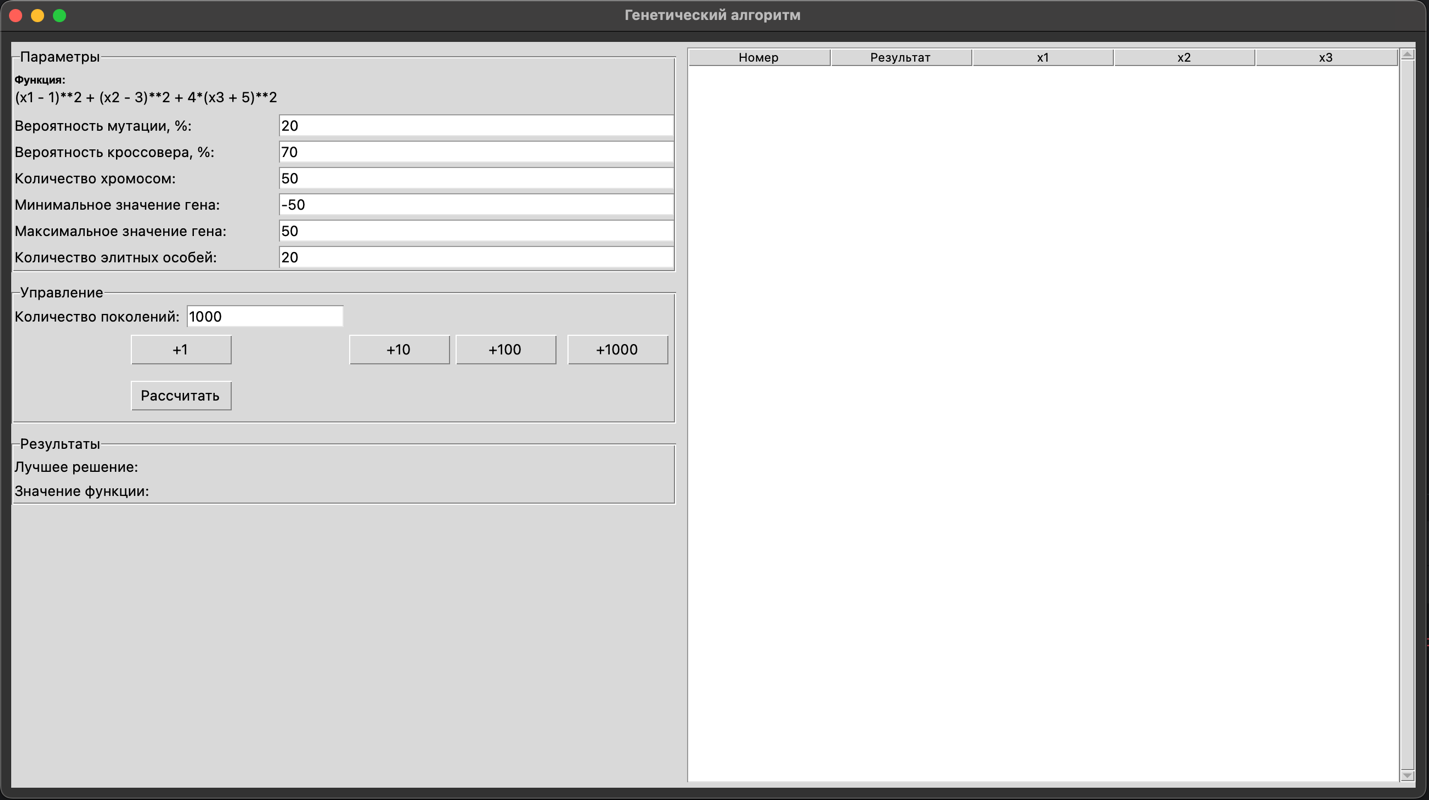


Рис 2. Интерфейс программы

Описание интерфейса:

1. В левой части интерфейса, в разделе "Параметры", находятся поля для ввода основных параметров алгоритма:
   1. Функция: отображает целевую функцию, которая будет минимизироваться. Функция задана и не может быть изменена.
   2. Вероятность мутации, %: Поле для ввода процента вероятности мутации. Определяет вероятность изменения генов в процессе мутации.
   3. Вероятность кроссинговера, %: Поле для ввода процента вероятности кроссинговера. Определяет долю генов, передаваемых от одного родителя.
   4. Количество хромосом: Поле для ввода количества особей в популяции.
   5. Минимальное значение гена: Устанавливает нижнюю границу диапазона значений для каждого гена в хромосоме.
   6. Максимальное значение гена: Устанавливает верхнюю границу диапазона значений для каждого гена.
   7. Количество элитных особей: Указывает, сколько лучших особей сохраняется без изменений в каждом поколении.
2. Управление поколениями
3. Раздел "Управление" включает:
   1. Количество поколений: Поле для ввода числа итераций (поколений) генетического алгоритма.
   2. Кнопки для увеличения количества поколений:
      1. +1: добавляет 1 к числу поколений.
      2. +10: добавляет 10.
      3. +100: добавляет 100.
      4. +1000: добавляет 1000.
   3. Эти кнопки позволяют быстро настраивать число поколений для выполнения алгоритма.
4. Кнопка "Рассчитать"
5. Кнопка "Рассчитать" запускает генетический алгоритм с текущими значениями параметров.
6. При нажатии этой кнопки алгоритм выполняет указанные итерации, а в правой части окна отображаются результаты.
7. Результаты
8. После завершения расчета результаты выводятся в левой части интерфейса, в разделе "Результаты":
   1. Лучшее решение: координаты x1,x2,x3x\_1, x\_2, x\_3x1​,x2​,x3​ оптимального решения, найденного алгоритмом.
   2. Значение функции: значение целевой функции для лучшего решения.
9. Отображение прогресса
10. Правая часть окна содержит таблицу, где отображается прогресс выполнения алгоритма:
    1. Колонки таблицы:
       1. Номер: номер поколения.
       2. Результат: значение целевой функции для лучшего решения в поколении.
       3. x1, x2, x3: координаты решения для данного поколения.
11. По завершении каждого поколения новые данные добавляются в таблицу, позволяя пользователю отслеживать прогресс.

Использование интерфейса

1. Настройка параметров: заполните поля параметров и укажите нужные значения.
2. Указание числа поколений: укажите число поколений вручную или используйте кнопки для его увеличения.
3. Запуск алгоритма: нажмите "Рассчитать" для выполнения алгоритма.

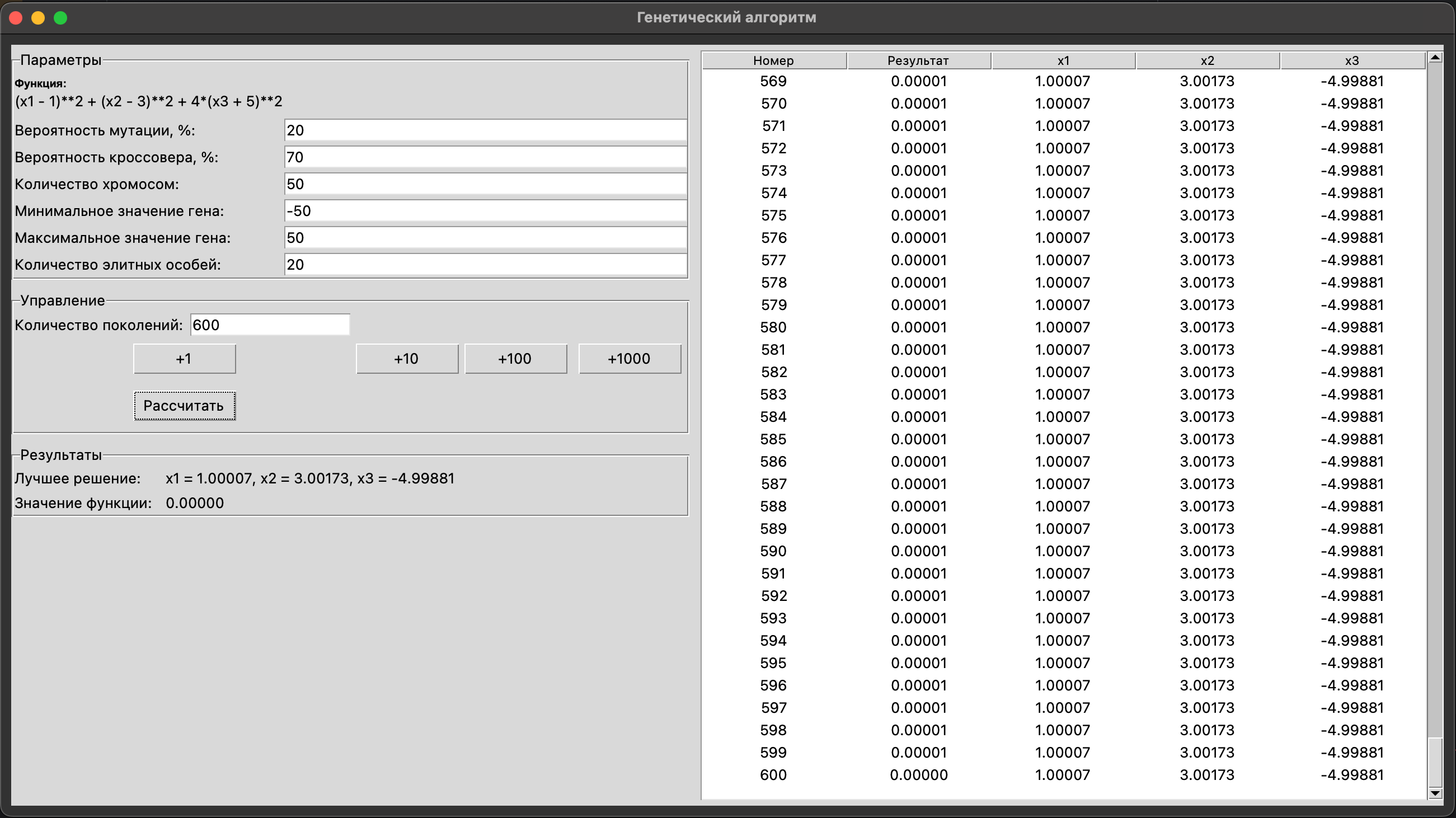


Рис 3. Результат выполнения программы

# Вывод

В рамках данной работы были исследованы генетические алгоритмы и их применение, разработан генетический алгоритм для конкретной функции и удобный интерфе йс взаимодействия с программой. Реализованный алгоритм успешно минимизировал тестовую функцию, подтвердив свою эффективность.