Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ МОРСКОЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
(СПБГМТУ)

Факультет Морского Приборостроения

Кафедра САУиБВТ

Практическая работа

на тему: "Методы первого порядка. Генетический алгоритм "

По дисциплине «Методы оптимизации»

Специальность: Вычислительные машины, комплексы, системы и сети

Выполнил:

Студент 3 курса группы № 3370

Сергиенко Д.К.

Подпись:

Проверил:

Профессор

Рыков Сергей Алексеевич

Подпись:

Дата выполнения отчёта:

Дата сдачи отчёта:

# Метод Первого порядка

## Генетический алгоритм

Требуется найти локальный минимум функции (приспособленности)  на множестве допустимых решений ,т.е. найти такую точку , что



Стратегия решения задачи состоит в итеративном поиске оптимального решения посредством выбора одного решения с минимальным значением функции приспособленности из случайно созданного и направленно (по отношению к функции приспособленности) изменяющегося множества.

Перед первой итерацией производится генерация наборов (далее - популяции) случайных векторов (далее - особей) , где - аргументы функции приспособленности  (далее - генотип), а  значение функции с этими аргументами. Количество особей в популяции задаётся параметром Генетического алгоритма.

После генерации популяции производится отбор особей (в частности, используется метод ранжирования - вероятность выбора зависит от места в списке особей, отсортированном по значению функции приспособленности). На этапе отбора из всей популяции выбирается определённая её доля, которая останется «в живых». Есть разные способы проводить отбор. Вероятность выживания конкретной особи должна зависеть от значения функции приспособленности . Сама доля выживших обычно является параметром генетического алгоритма, и её задают заранее. По итогам отбора, некоторые войдут в итоговую популяцию, остальные особи погибают.

Следующий этап – размножение (скрещивание). Размножение в разных алгоритмах определяется по-разному — оно, конечно, зависит от представления данных. Главное требование к размножению — чтобы потомок или потомки имели возможность унаследовать черты обоих родителей, «смешав» их каким-либо способом. В частности, этап определён таким образом, что потомок имеет генотип, элементы которого равны среднему арифметическому соответствующих элементов генотипа родителей с вероятностью 50%, в иных случаях генотип ребёнка больше похож на одного из родителей.  
Размножение в генетических алгоритмах требует выбора для производства потомка нескольких родителей, обычно – двух (в работе представлена панмиксия — оба родителя выбираются случайно, каждая особь популяции имеет равные шансы быть выбранной). Особи для размножения обычно выбираются из всей популяции, а не из выживших на первом шаге элементов, поскольку главный недостаток многих генетических алгоритмов — отсутствие разнообразия в особях. Достаточно быстро выделяется один единственный генотип, который представляет собой локальный минимум, далее все элементы популяции проигрывают ему отбор, и вся популяция «забивается» копиями этой особи. Есть разные способы борьбы с таким нежелательным эффектом. Один из них — выбор для размножения вообще всех особей. Однако такой подход вынуждает хранить всех существовавших ранее особей, что увеличивает вычислительную сложность задачи. Поэтому в работе реализуется отбор всех особей из популяции, потомки которых заменяют самых неприспособленных (с большим значением функции приспособленности) особей – из конца отсортированного списка. При таком подходе для разнообразия генотипа возрастает роль мутаций.

Мутации

К мутациям относится все то же самое, что и к размножению: есть некоторая доля мутантов, являющаяся параметром генетического алгоритма, и на шаге мутаций нужно выбрать несколько особей, а затем изменить их в соответствии с заранее определёнными операциями мутации. Был выбран вариант мутации изменяющий (+ или -) каждый элемент генотипа особи на коэффициент мутации.

Критерием остановки алгоритма может быть:

* нахождение глобального, либо субоптимального решения;
* исчерпание числа поколений, отпущенных на эволюцию;
* исчерпание времени, отпущенного на эволюцию.

Таким критерием был выбран  - погрешность вычисления оптимума.

Алгоритм.

1. Задать целевую функцию (приспособленности) для особей популяции
2. Создать начальную популяцию (и провести селекцию)

Начало цикла

1. Размножение (скрещивание)
2. Мутирование
3. Формирование нового поколения (селекция)
4. Проверка условия остановки. Если выполнятся, то (конец цикла), иначе повторение цикла (Шаг 3).

Листинги программ в пакете MathCAD15 приведены в приложении А.

# Приложение A. Листинги программ Выполненных примеров решения задач с помощью генетического алгоритма в пакете MathCAD15

## Пример 1 - Задача №1: найти локальный минимум функции

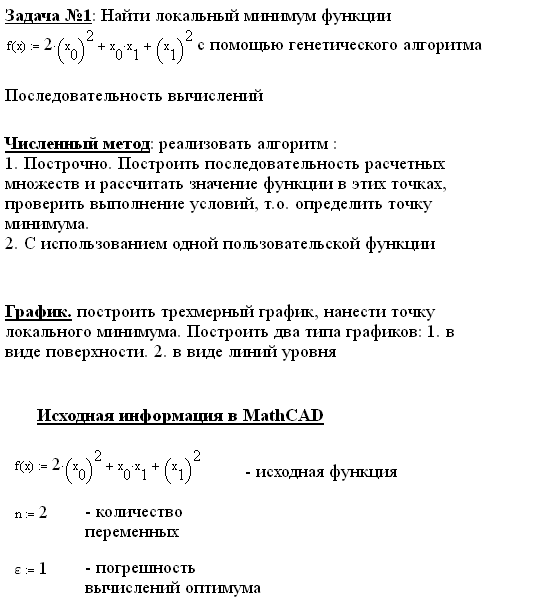


Рисунок 1 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15

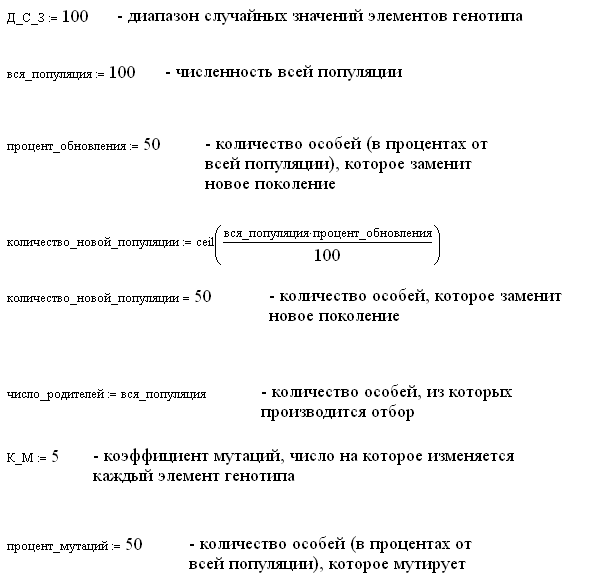


Рисунок 2 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

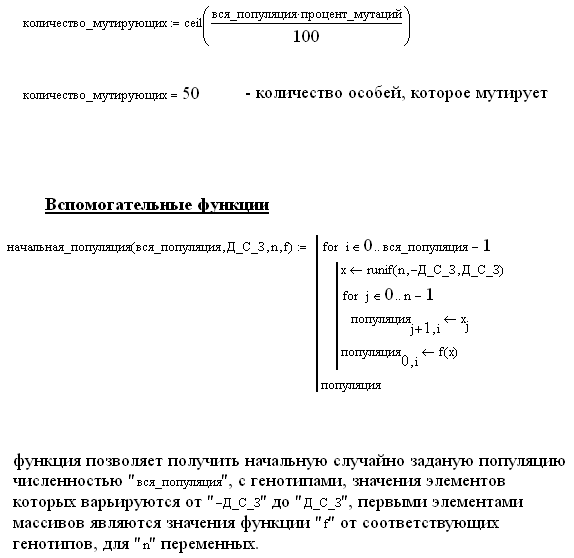


Рисунок 3 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

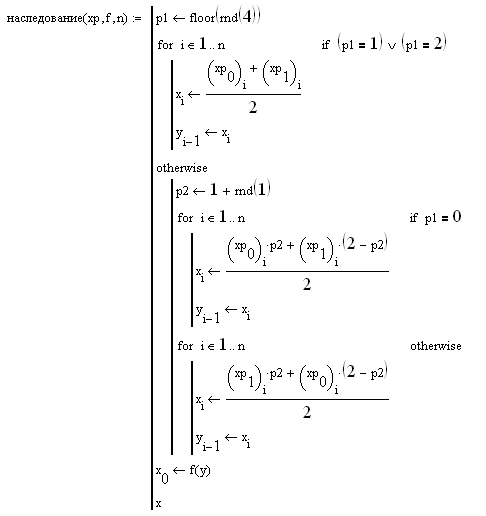


Рисунок 4 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

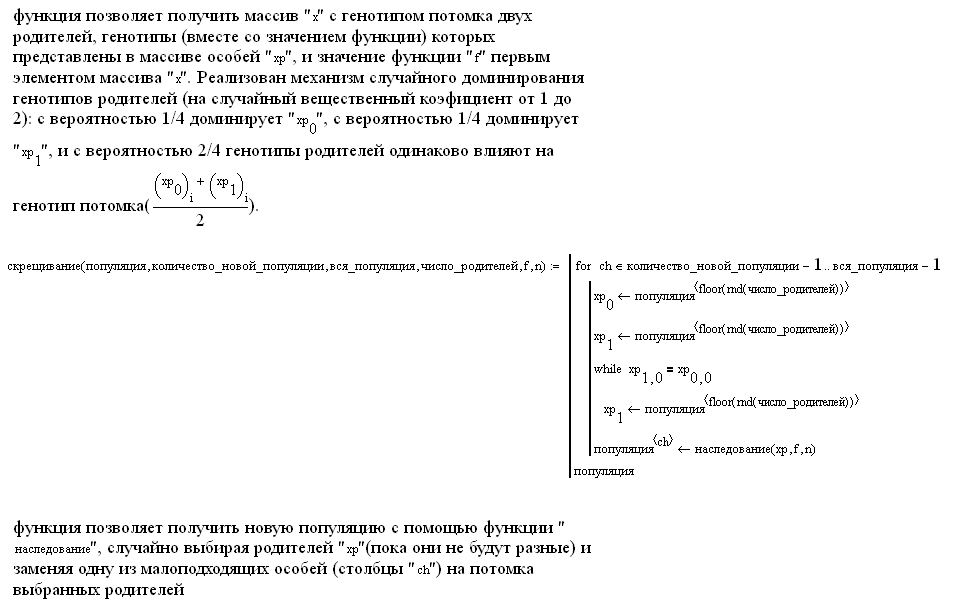


Рисунок 5 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

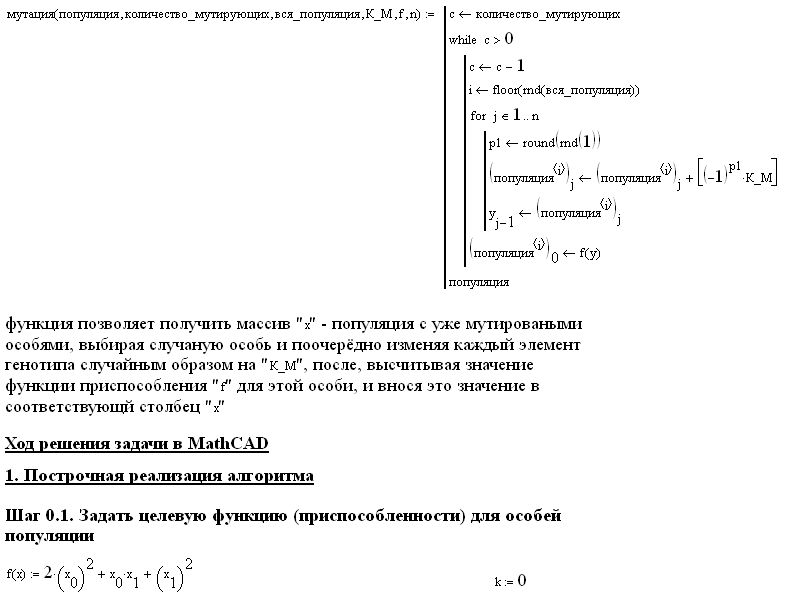


Рисунок 6 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

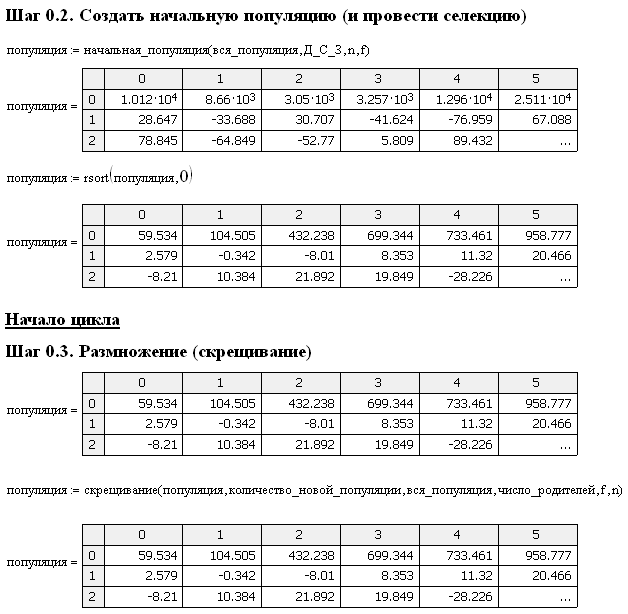


Рисунок 7 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

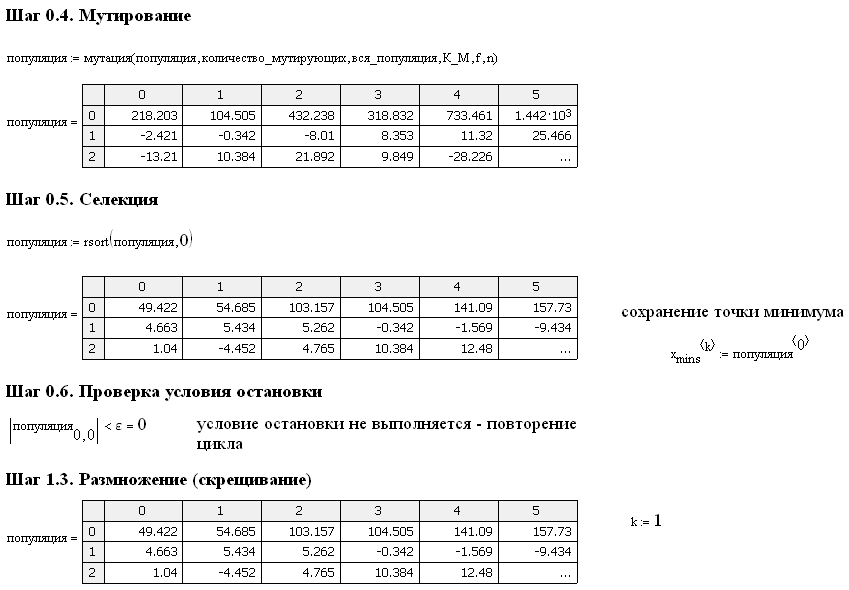


Рисунок 8 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

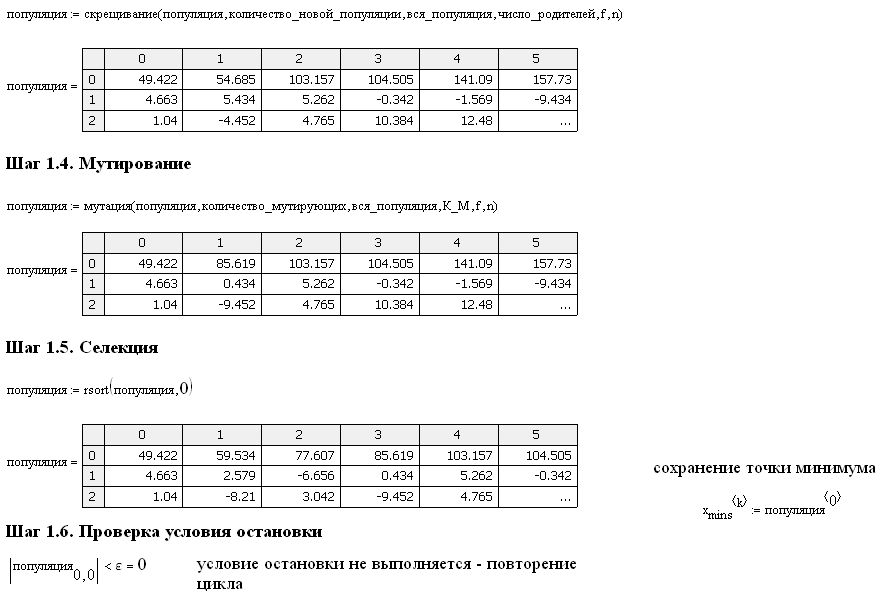


Рисунок 9 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

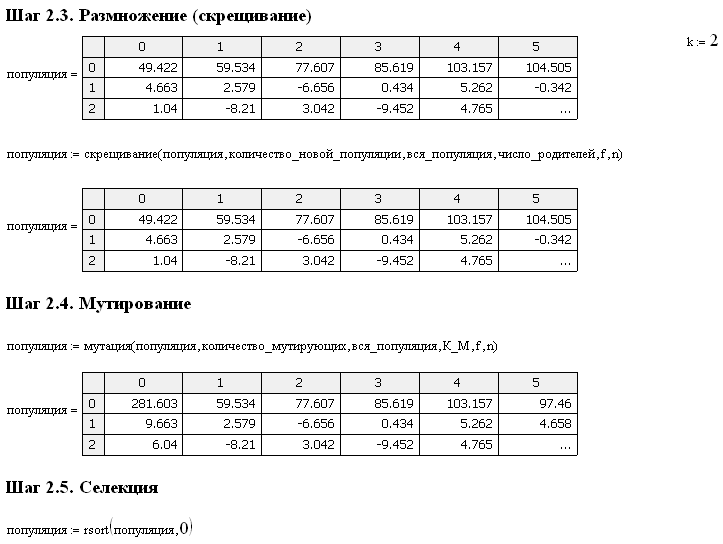


Рисунок 10 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

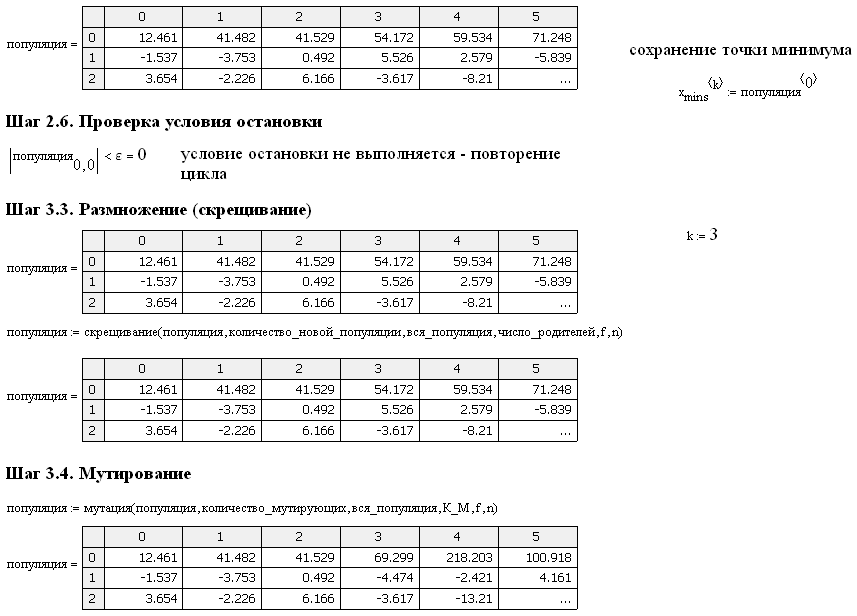


Рисунок 11 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

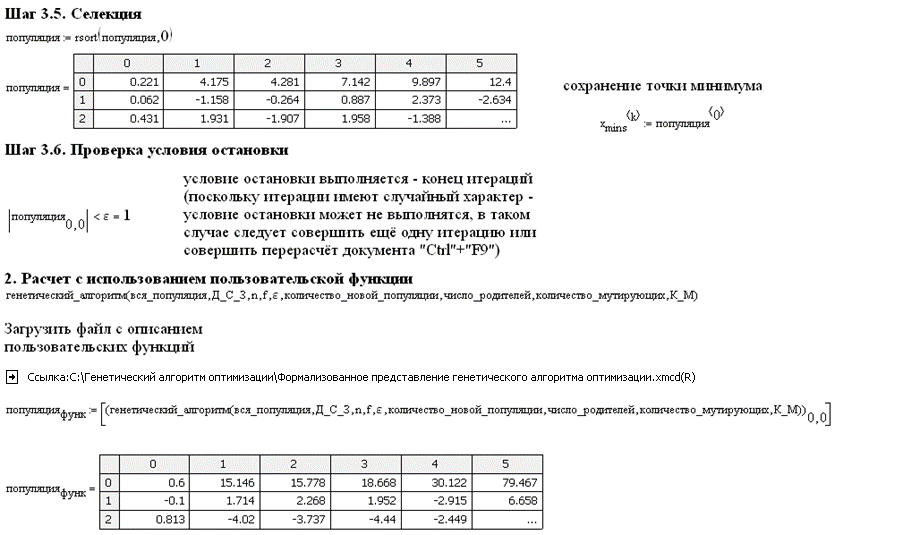


Рисунок 12 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

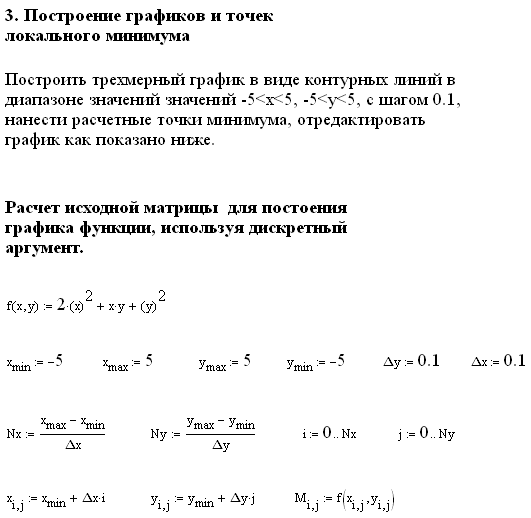


Рисунок 13 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

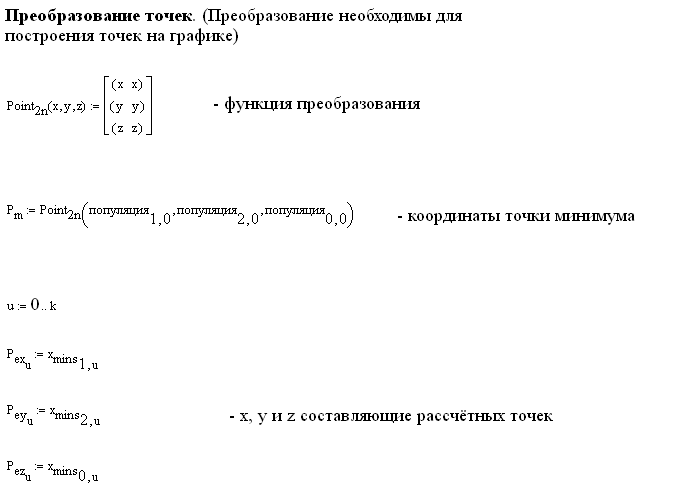


Рисунок 14 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

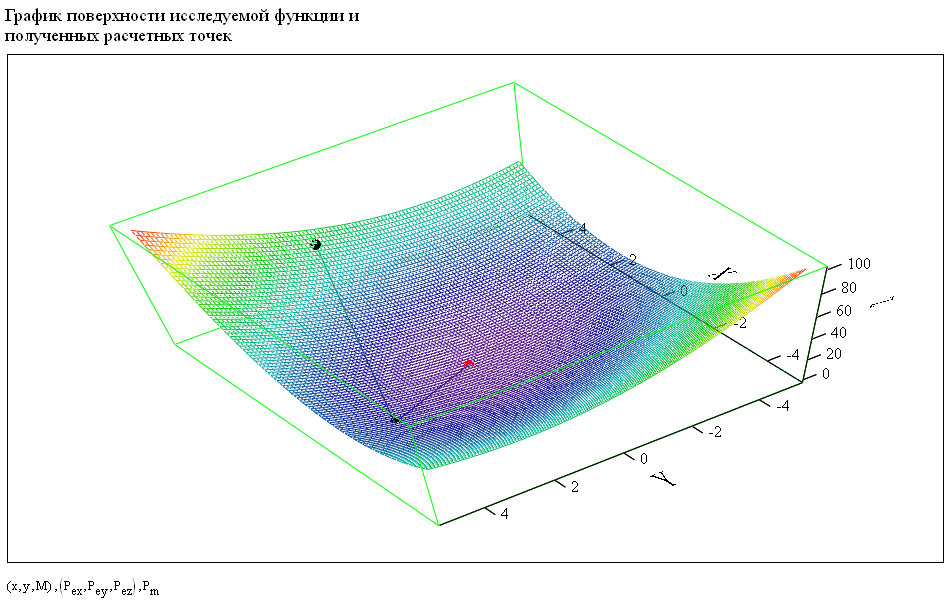


Рисунок 15 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

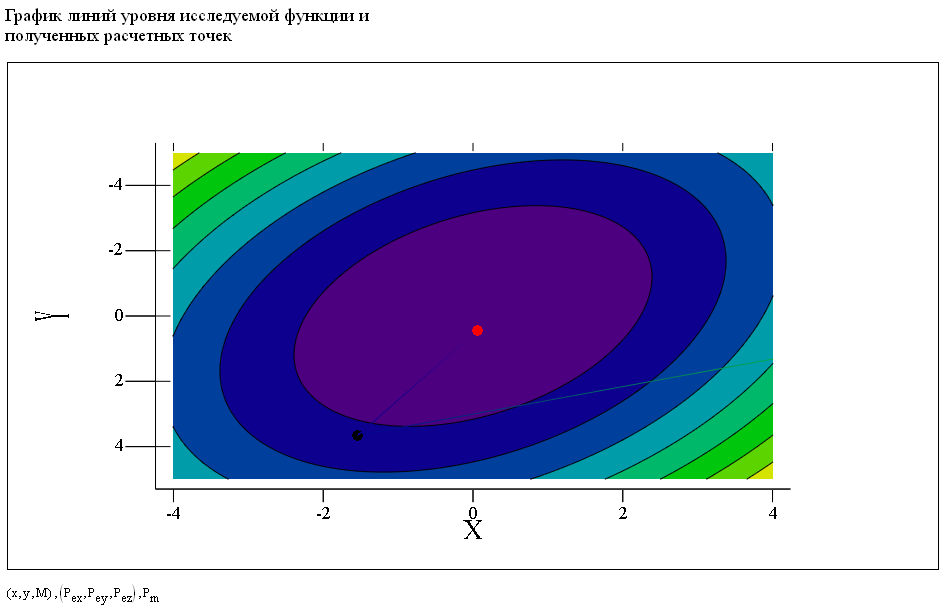


Рисунок 16 - Листинг программы выполнения задачи №1: найти локальный минимум функции с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

## Пример 2 - Задача №2: найти локальный минимум функции Розенброка

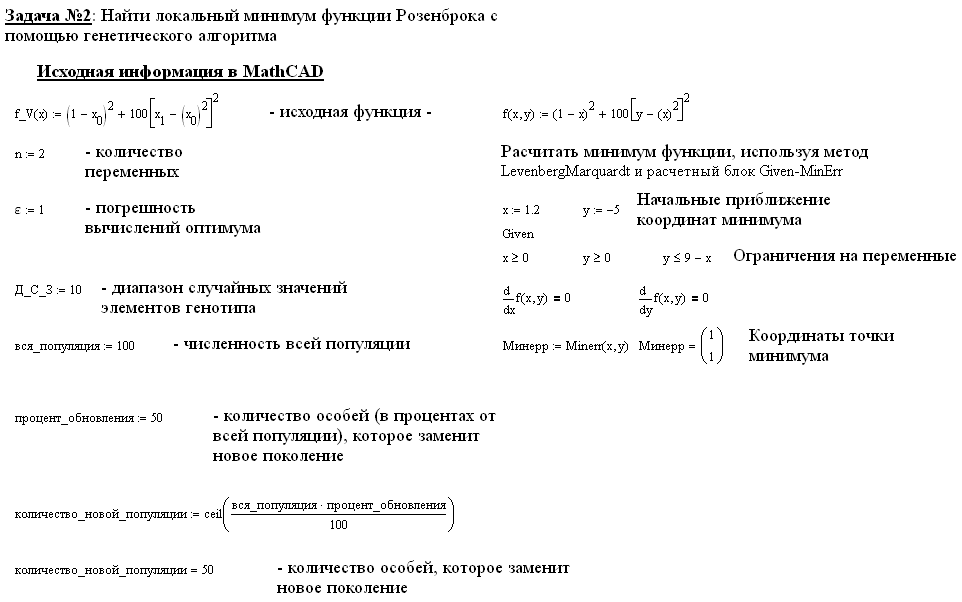


Рисунок 1 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15

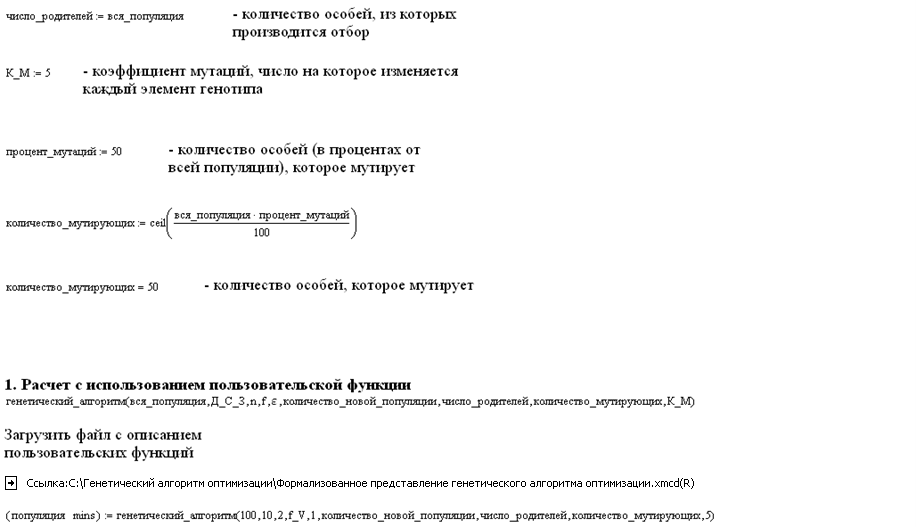


Рисунок 2 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

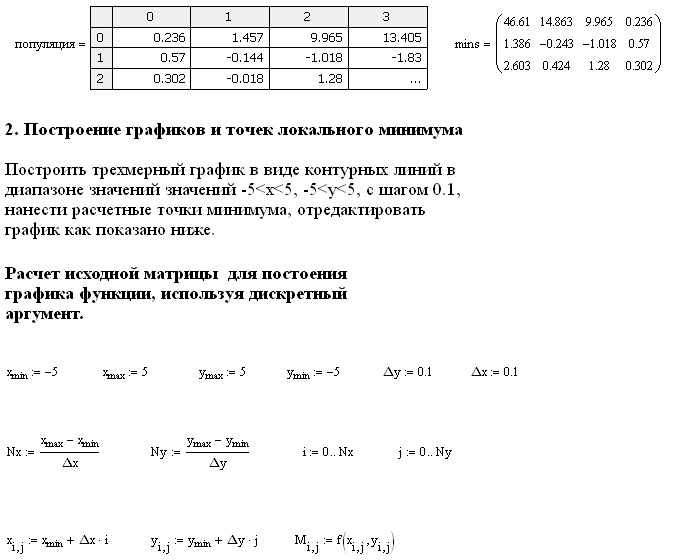


Рисунок 3 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

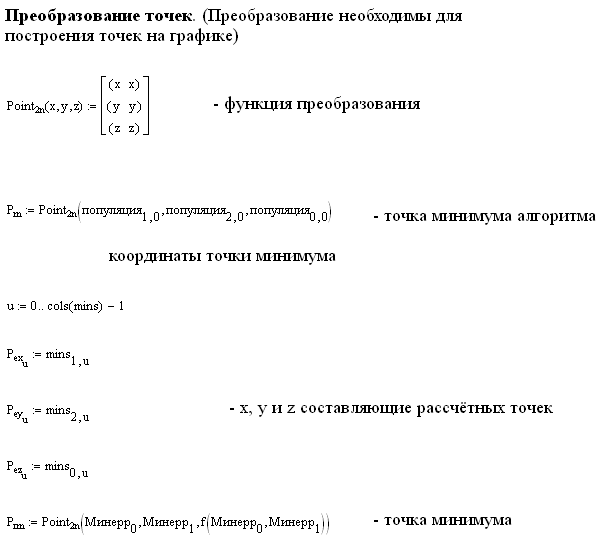


Рисунок 4 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

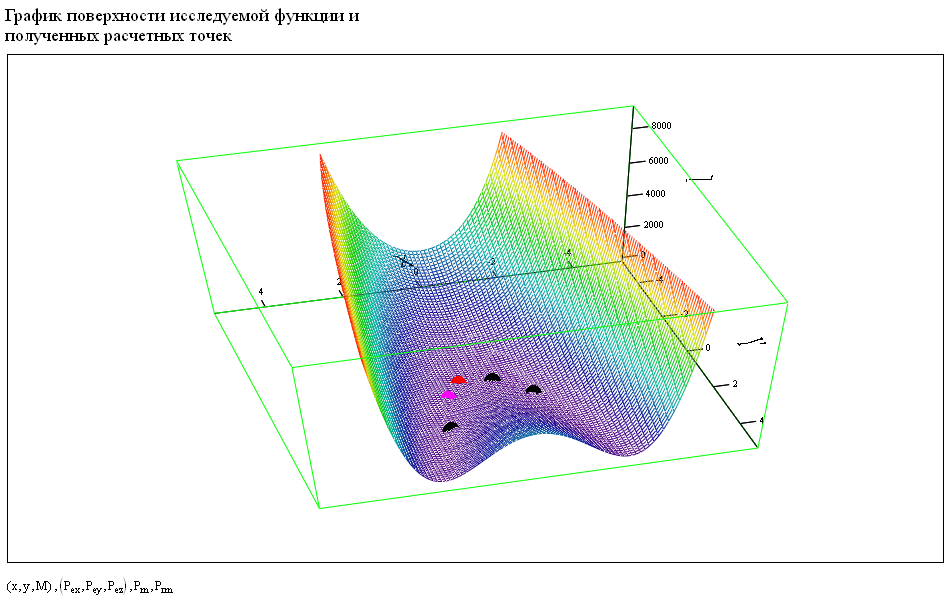


Рисунок 5 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

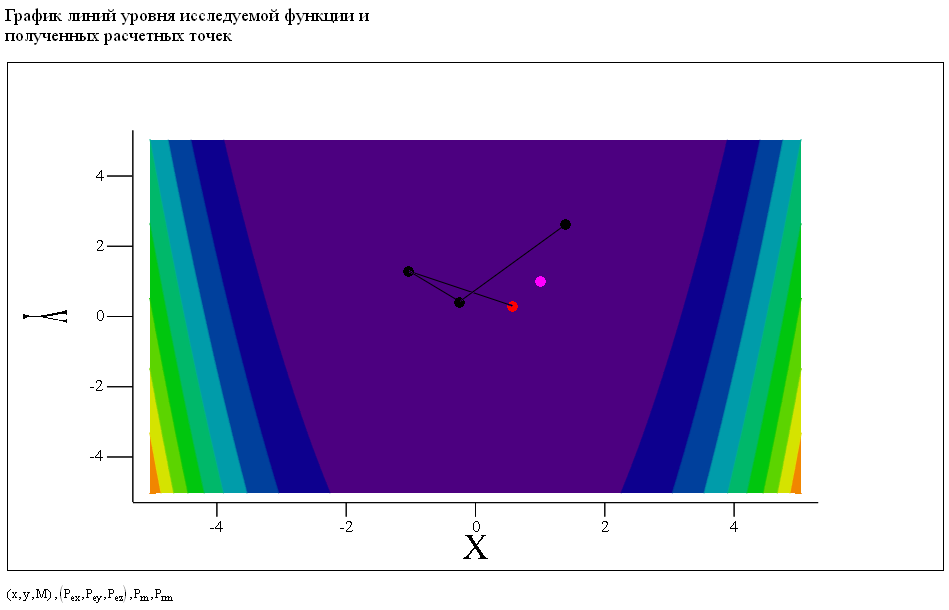


Рисунок 6 - Листинг программы выполнения задачи №2: найти локальный минимум функции Розенброка с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

## Пример 3 - Задача №3: найти локальный минимум функции Химмельблау

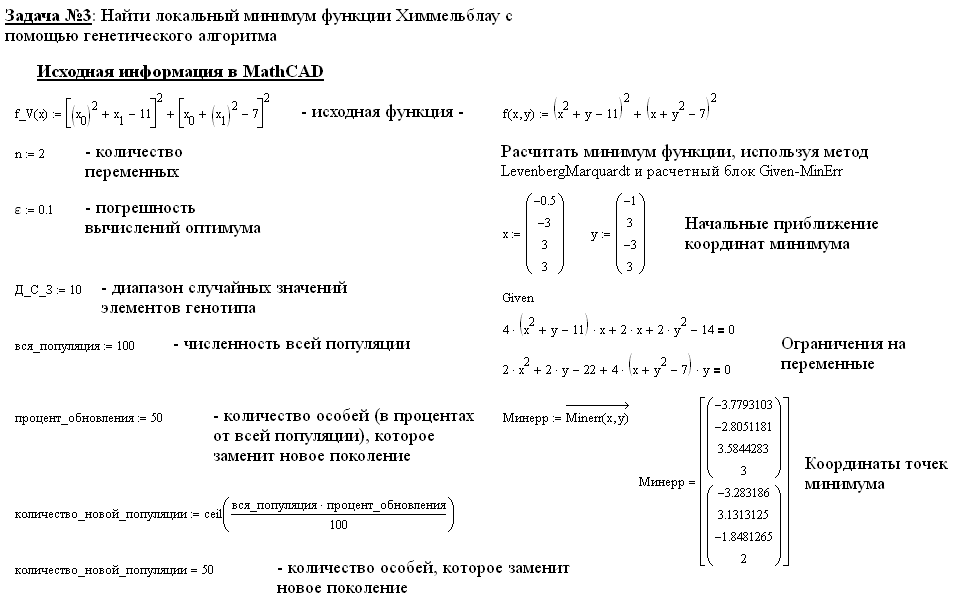


Рисунок 1 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15

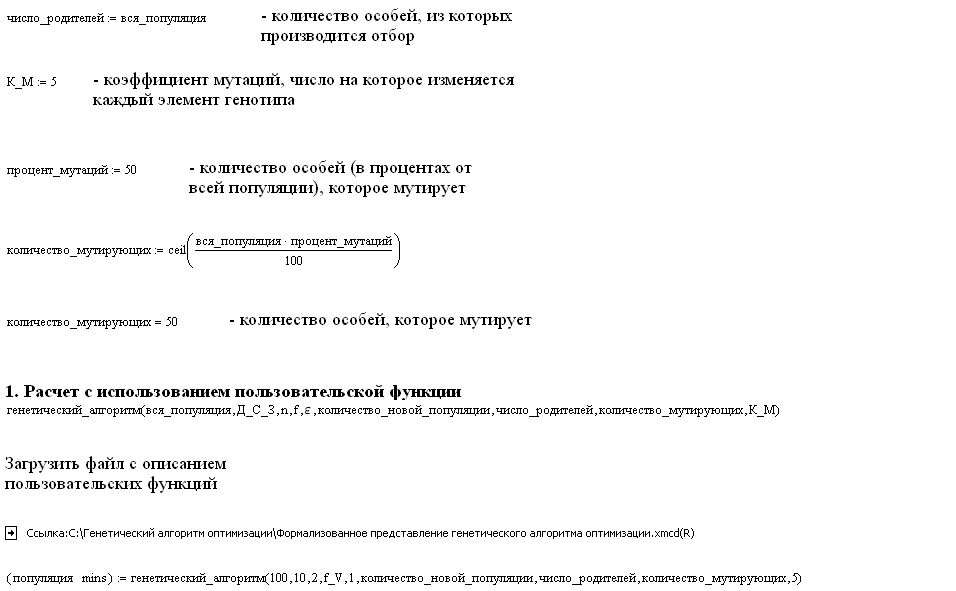


Рисунок 2 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

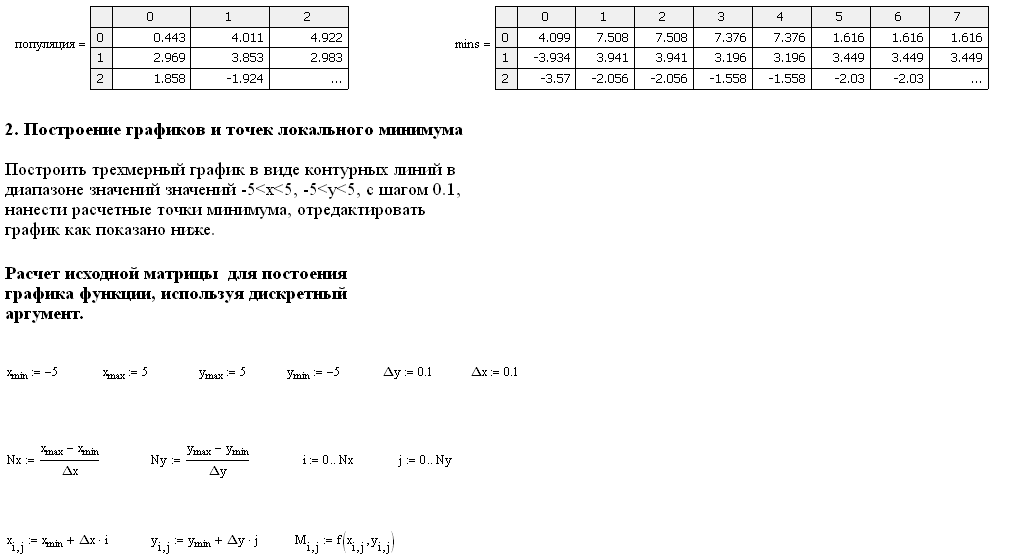


Рисунок 3 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

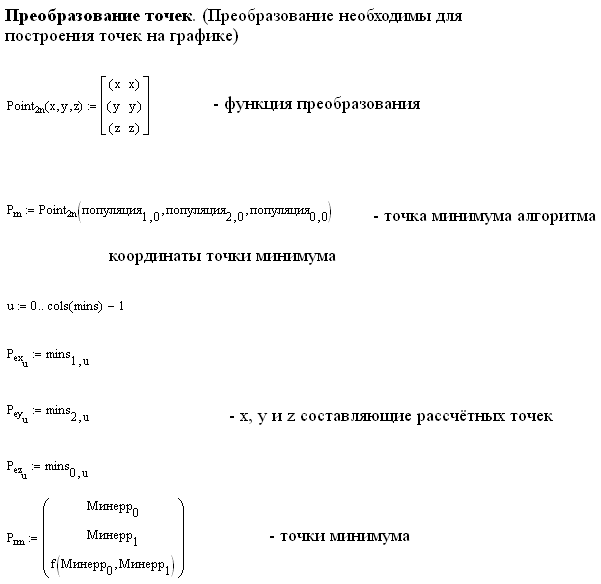


Рисунок 4 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

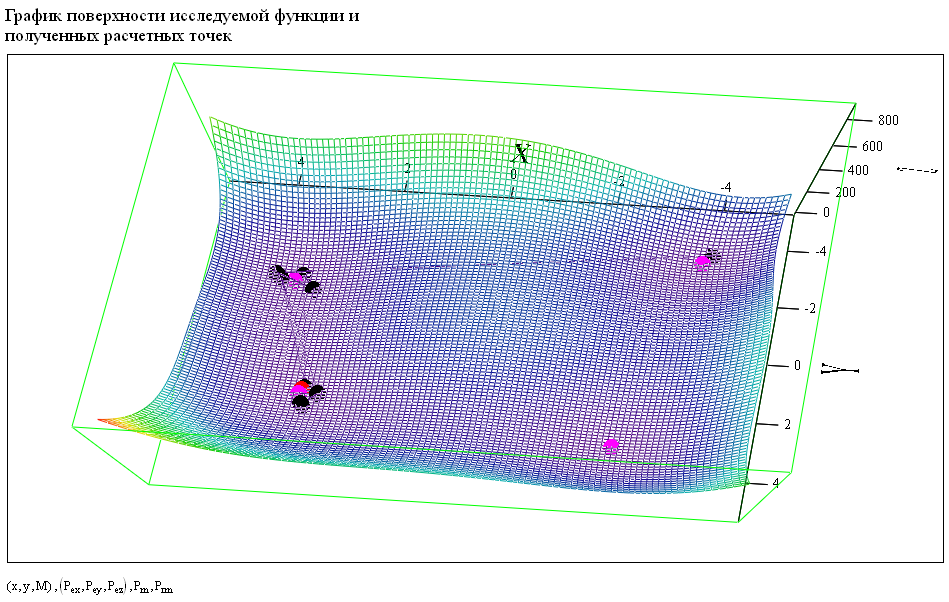


Рисунок 5 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15 (продолжение)

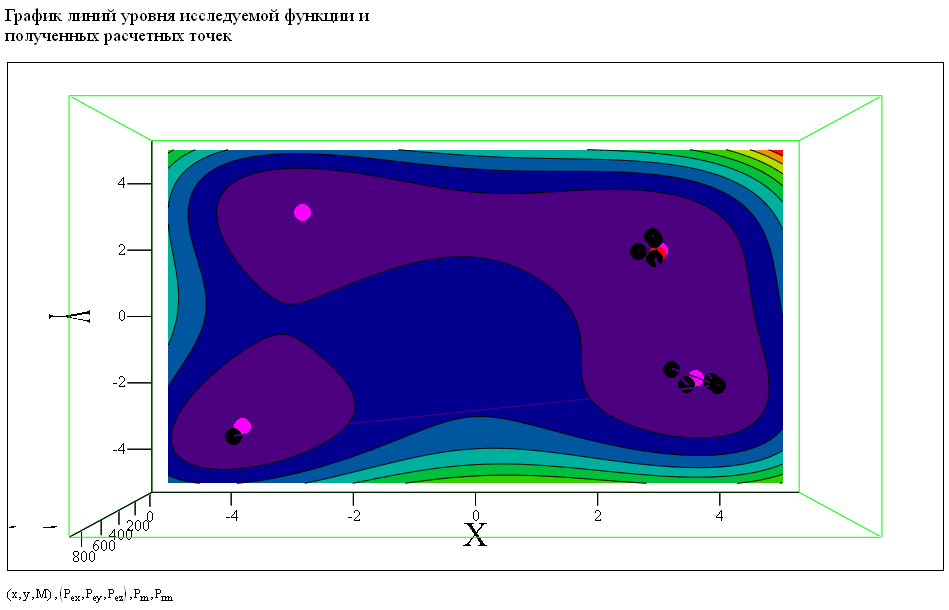


Рисунок 6 - Листинг программы выполнения задачи №2: найти локальный минимум функции Химмельблау с помощью генетического алгоритма в пакете MathCAD15 (продолжение)