МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**«**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭКОНОМИЧЕСКИЙ УНИВЕРСИТЕТ[**»**](https://docs.google.com/document/d/1X_fw2P3HRH7P6hlazUxDCrFlY576Zx6doYUB4sMirgw/edit#heading=h.xpk852hjikfe)

Факультет информатики и прикладной математики

Кафедра прикладной математики и информатики

**КУРСОВАЯ РАБОТА**

по дисциплине:

**«Системы компьютерной математики»**

Тема: **«**Разработка стратегии прохождения гоночной трассы при помощи генетического алгоритма**»**

Направление: 01.03.02 Прикладная математика и информатика

Студент: Майко Максим Витальевич

Группа ПМ-1901 Подпись:

Проверил: Васильев Юрий Михайлович

Должность: старший преподаватель

Оценка: Дата:

Подпись:

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ**](#_6si596uc8b14) **3**

[**1. РЕШЕНИЕ ПОСТАВЛЕННОЙ ЗАДАЧИ**](#_qf3m81tx3vsj) **4**

[1.1. Определение объектов «cars»](#_9ilm9zt882kv) 4

[1.2. Генетический алгоритм](#_u4ysumya2hlu) 7

[**2. ВИЗУАЛИЗАЦИЯ РАБОТЫ АЛГОРИТМА**](#_1mqpxoexgwec) **14**

[2.1. Прорисовка путей отобранных в данном поколении индивидов](#_j3zgjbqwq6nh) 14

[2.2. Поэтапное движение лучшего индивида данного поколения](#_4docwc1bkw0z) 16

[**3. АНАЛИЗ РАБОТЫ АЛГОРИТМА**](#_iq5dd13znot3) **19**

[3.1. Проверка выдвинутой гипотезы](#_978k6n663ifd) 19

[**ЗАКЛЮЧЕНИЕ**](#_pv9zfzw71hdh) **22**

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**](#_ck34bystdp1j) **24**

# **ВВЕДЕНИЕ**

Рассматривается игра «Racetrack», в которой машинки пытаются пройти гоночную трассу, дойдя до «зоны финиша». Под машинкой подразумевается точка на плоскости, а под гоночной трассой — область, ограниченная двумя замкнутыми линиями без самопересечений, одна из которых лежит во внутренней области другой. Во внутренней области наименьшей (по площади) замкнутой линии фиксируется точка, которая будет являться «центром» гоночной трассы.

Каждая машинка является объектом, который хранит в себе информацию о своих текущих координатах, геноме, векторе инерции, значение трех сенсоров и текущей пройденной дистанции (подробнее в разделе 1.1).

По правилам игры, каждый ход машинка должна перемещаться вдоль своего текущего вектора инерции, затем сделать шаг в одном из девяти направлений. Если после хода машинка оказывается на границе гоночной трассы или вне ее пределов, то она разбивается и выбывает из игры.

Очевидно, что все машинки можно разделить на две группы: способные доехать до зоны финиша и неспособные. Машинка будет называться обученной, если она относиться к первой группе соответственно.

Основная задача: разработать стратегию прохождения гоночной трассы при помощи генетического алгоритма [1].

К рассмотрению предлагается гипотеза, что генетический алгоритм способен создать обученные машинки, начиная с произвольного набора машинок. Иными словами, генетический алгоритм способен обучить машинки достигать зоны финиша гоночной трассы.

В ходе работы будет выполнена основная задача и проверена выдвинутая гипотеза путем анализа работы представленного алгоритма.

# **1. РЕШЕНИЕ ПОСТАВЛЕННОЙ ЗАДАЧИ**

## **1.1. Определение объектов «*cars*»**

Как уже было сказано выше, каждая машинка в алгоритме будет относиться к объекту, который хранит в себе информацию о своих текущих координатах, векторе инерции, геноме, сенсорах и текущей пройденной дистанции.

Текущими координатами машинки будет являться список, содержащий два числа — значение абсциссы и ординаты точки на плоскости, соответствующей данному агенту.

Вектором инерции будем называть вектор, соединяющий точку предыдущего положения машинки с текущим. В начале каждая машинка будет иметь в качестве вектора инерции вектор . Выбор ненулевого начального вектора инерции обусловлен тем, что мы требуем от машинок движения по гоночной трассе строго в одном направлении — по часовой стрелке относительно начальной позиции машинок. Если бы в качестве начального вектора инерции был выбран нулевой вектор, мы бы не смогли гарантировать движение всех машинок в одном направлении, в связи с чем пришлось бы в дальнейшем ввести дополнительные ограничения на первый ход каждой машинки. В памяти мы будем хранить вектор инерции как список из двух чисел, модуль каждого из которых не превосходит пяти. Данное ограничение позволит зафиксировать максимально возможную скорость машинки вдоль и .

Каждая машинка имеет три сенсора: и. Каждый из сенсоров несет в себе число от 0 до 2 в зависимости от того, насколько далеко относительно текущей позиции машинки находятся пределы гоночной трассы. Сенсордействует вдоль текущего вектора инерции, сенсоры и — с отклонением на в левую и правую стороны относительно текущего вектора инерции соответственно. Отклонение будет производиться за счет умножения вектора инерции на соответствующие матрицы поворота (матрицу или , которые показаны на рисунке 3). Изначально каждый сенсор несет в себе число 0. Затем перед каждым ходом машинки происходит проверка на выход за пределы гоночной трассы соответствующими ей сенсорами. Если пределы относительно сенсора близко, его значение не меняется. Если пределы относительно сенсора на нормальной или большой дистанции, то его значение меняется на 1 или 2 соответственно.

Теперь определим понятие и структуру генома машинки. Геном будем называть число, определяющее траекторию хода машинки. Таким образом, геном будет называться число от 0 до 8, соответствующее одному из девяти возможных направлений движения машинки, как показано на рисунке 1. Геномом будем называть набор генов машинки. Размером генома будем называть количество генов в нем. В геноме будет содержаться 3267 генов. Зафиксировать данное число генов в геноме позволяет ограничение максимальной скорость машинки вдоль всех осей. Из-за ограничения каждая из координат вектора инерции варьируется от до , следовательно имеем различных векторов инерции для наших машинок. Имея три сенсора с тремя возможными значениями для каждого, получаем вариантов для значений сенсоров. Суммарно выходит число , и которое берется в качестве размера генома.

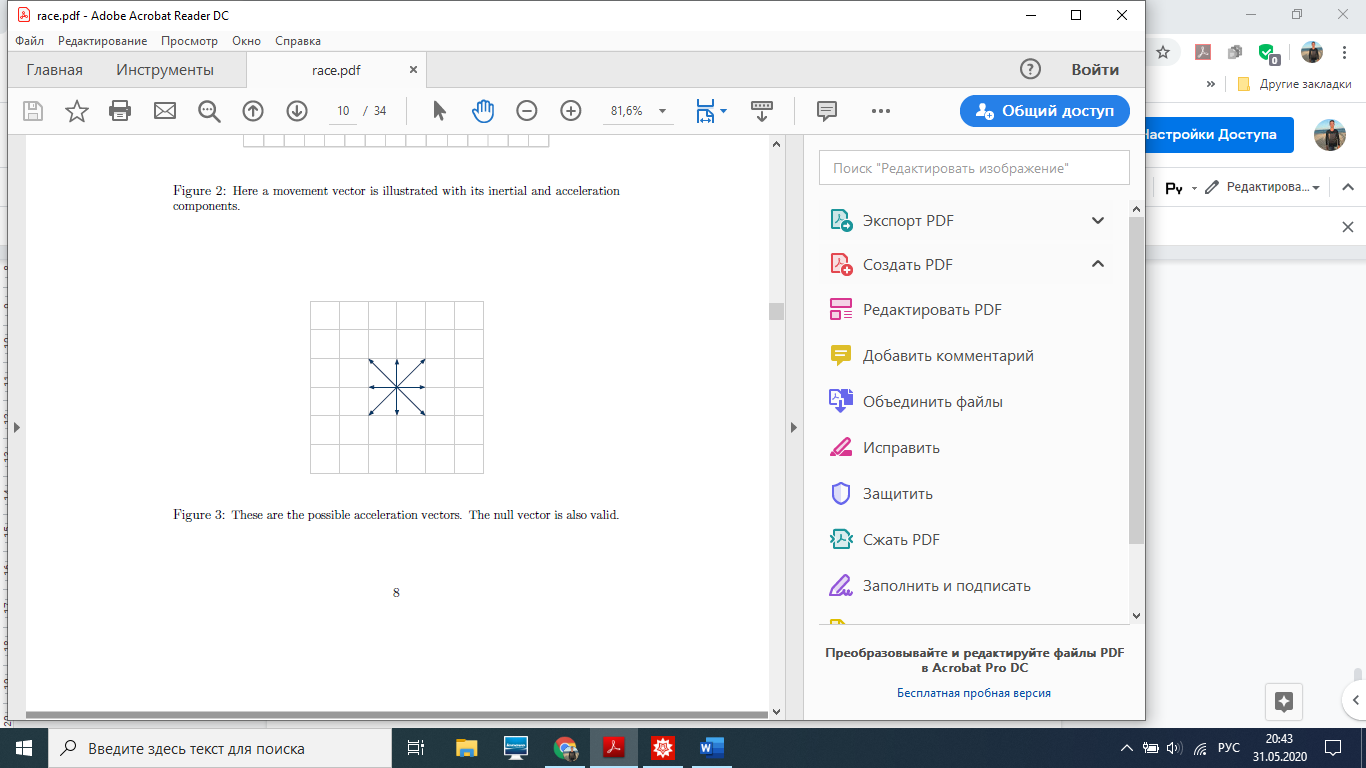


Рисунок 1 – Всевозможные векторы шага для машинки (с учетом нулевого)

Текущей пройденной дистанцией (или ) будем называть величину угла в радианах между вектором, проведенным из зафиксированного заранее центра гоночной трассы в текущее положение машинки, и вектором, проведенным из зафиксированного заранее центра гоночной трассы в точку старта всех машинок. Угол будет измеряться против часовой стрелки. Изначально равен нулю.

Теперь имеется вся необходимая база для определения объектов . Будет сформировано две категории: и . К первой категории будут принадлежать машинки из начального поколения, ко второй — машинки, которые будут участвовать в генерации новых поколений, то есть машинки-потомки. Каждую из категорий определим в виде ассоциаций с отложенным присваиванием в системе компьютерной алгебры Wolfram Mathematica [2], как показано на рисунке 2.

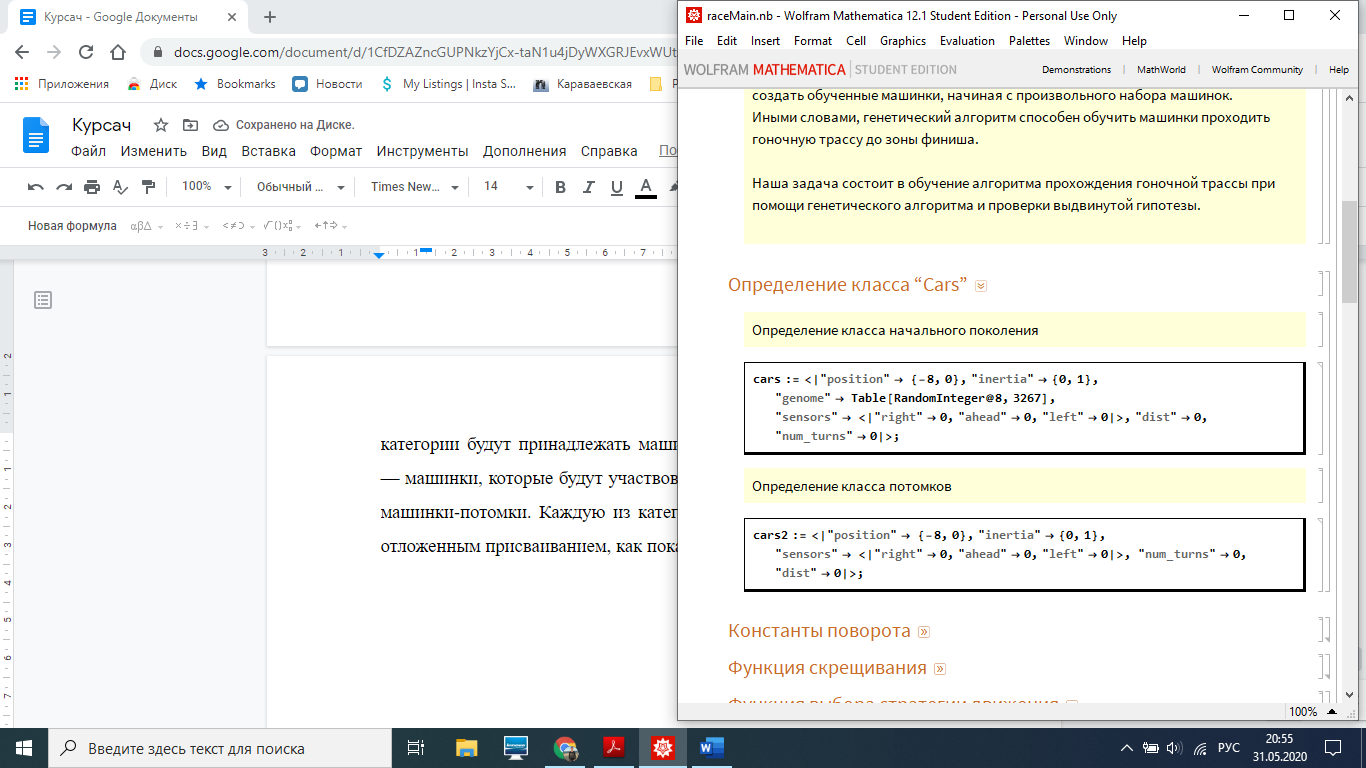


Рисунок 2 – Определение двух категорий машинок

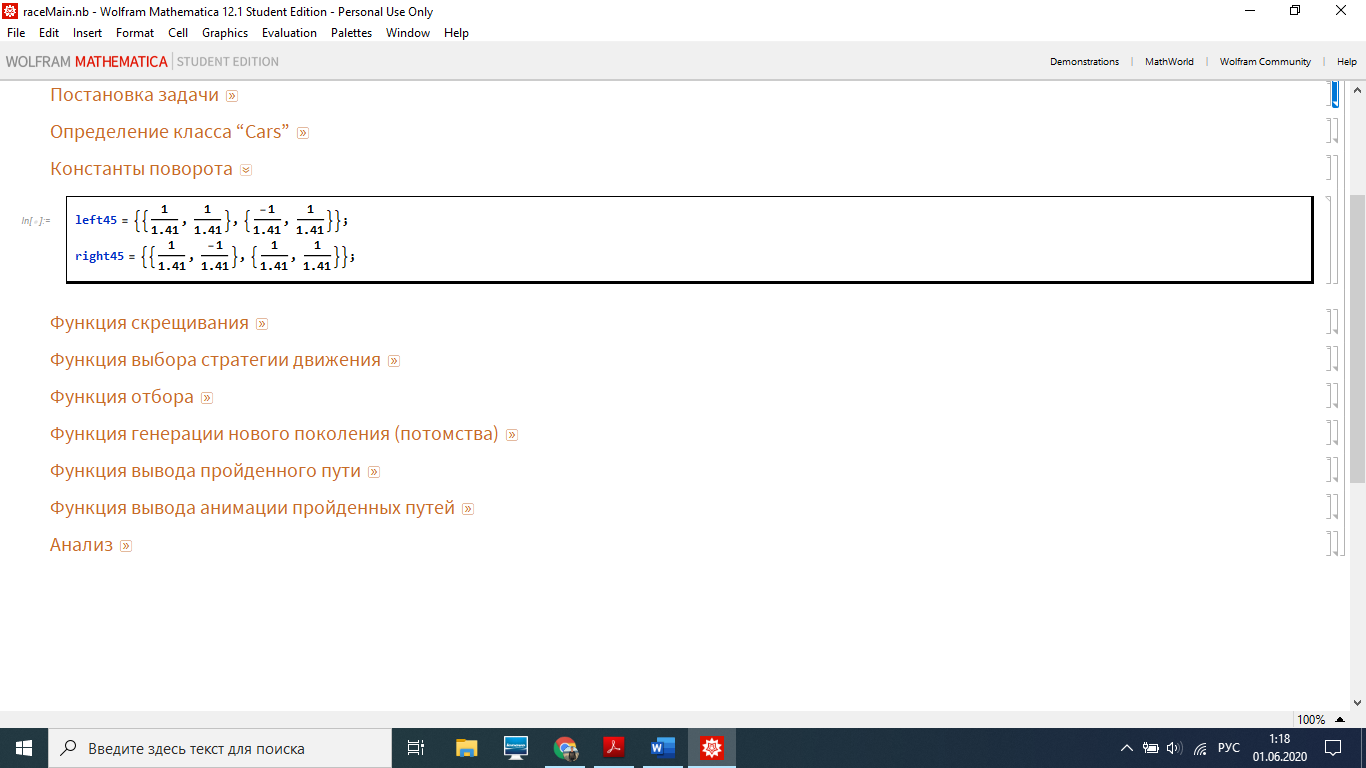


Рисунок 3 – Матрицы поворота

## **1.2. Генетический алгоритм**

Для работы генетического алгоритма, обучающего машинки проходить гоночную трассу, необходимо определить понятия индивида, популяции и , функцию выбора стратегии движения, функцию отбора индивидов, функции приспособленности (или ) и генерации потомства путем использования функции скрещивания.

Индивидом будут называться объекты типа и .

Популяцией будет называться совокупность индивидов.

Шагом генетического алгоритма является последовательность действий: отбор индивидов из текущего поколения, процесс скрещивания отобранных индивидов, на выходе которого мы получаем сгенерированное потомство.

будет называться популяция, полученная на шаге алгоритма.

В ходе данной работы машинки будут проходить гоночную трассу , которая задается двумя окружностями радиусов 5 и 10 с общим центром в точке . Как было показано на рисунке 2, машинки стартуют из точки с координатами (-8,0). Центром гоночной трассы положим точку с координатами (0,0).

Определим функцию выбора стратегии движения :

Для начала введем функцию , которая будет находить индекс в геноме на текущем этапе. На вход подается индивид в качестве обязательного аргумента и значение максимальной скорости в качестве необязательного аргумента. Базовое значение максимальной скорости равно 5. На выходе получаем индекс в геноме выбранного индивида.

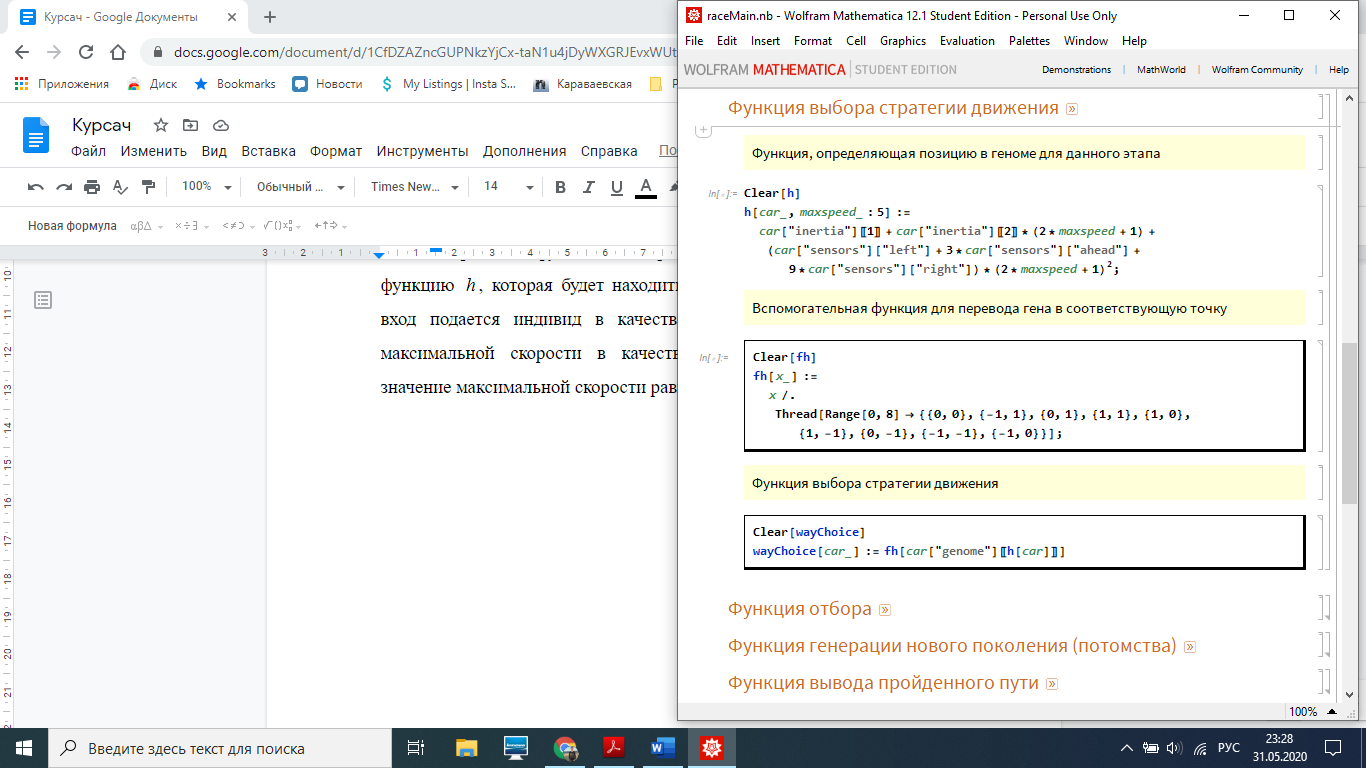


Рисунок 4 – функция

Теперь введем вспомогательную функцию , которая получает на вход ген и на выходе выдает соответствующий ему вектор шага.



Рисунок 5 – функция

Используя функции и определим функцию выбора стратегии движения . На вход подается индивид. На выходе получаем вектор шага.

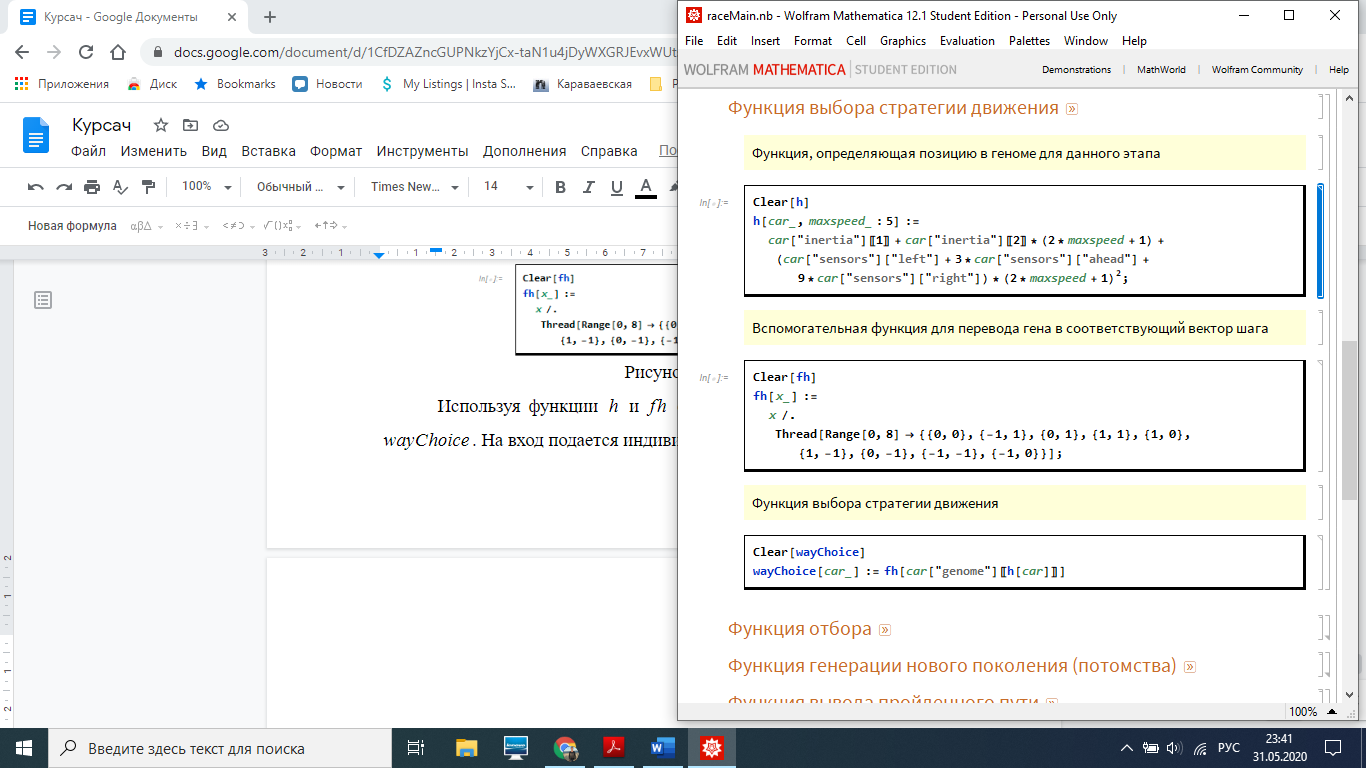


Рисунок 6 – функция

Определим функцию отбора :

Функции отбора будет задана рекурсивно (см. рисунок 11). На вход подается список из 100 индивидов (текущее поколение).

Идея функции отбора заключается в том, чтобы прогонять машинки по гоночной трассе, отбрасывая разбившиеся до тех пор, пока в живых не останется 10 машинок. Очевидно, что оставшиеся индивиды будут являться лучшими в данном поколении, поэтому именно они в дальнейшем будут выступать в роли родителей в процессе скрещивания и создания нового поколения машинок (потомства).

Таким образом вход в тело рекурсии осуществляется, пока число индивидов в списке, указанном в аргументе, больше десяти. В ином случае – выводим аргумент функции.

Тело рекурсии будет состоять из четырех частей, которые выполняются последовательно друг за другом.

Первая часть настраивает сенсоры каждого индивида из списка, поданного в качестве аргумента функции (см. рисунок 7).

Во второй части каждая машинка делает свой ход, то есть к текущей позиции каждой машинки прибавляется ее вектор инерции и вектор, полученный из функции для данной машинки. Затем для каждого индивида рассчитывается новое значение и новое значение вектора инерции. Реализация второй части показана на рисунке 8.

В третьей части происходит отбор индивидов по следующим критериям:

1. Индивид остался в пределах гоночной трассы

2. Значение у индивида стало больше, чем на предыдущем этапе отбора

3. Модуль значения вектора инерции по абсциссе не превосходит 5

4. Модуль значения вектора инерции по ординате не превосходит 5

Реализация третьей части показана на рисунке 9.

Четвертая часть состоит из проверки нескольких условий, в зависимости от которых отобранные на текущем этапе индивиды либо повторно попадают в тело рекурсии, либо попадают в выход из рекурсии. Стоит отметить, что если у индивида значение больше или равно 5.6, то он попал в зону финиша и, следовательно, успешно прошел гоночную трассу. Поэтому первым условие для проверки в четвертой части является поиск таких индивидов. Если они нашлись, то на экране появляется сообщение «Победа» и из текущего списка индивидов выбирают первые десять обученных машинок (или меньше десяти в зависимости от их количества).

Если же обученных машинок нет, а длина списка индивидов при этом оказывается меньше 10, то мы дополняем до 10 текущий список отобранных индивидов с конца лучшими из тех, кто не смог пройти отбор и тем самым попадаем в выход из рекурсии. Дополнение до десяти индивидов таким образом объективно в силу наличия мутации в процессе скрещивания, о котором подробнее будет написано ниже.

Если первые два условия не прошли, то это означает, что перед нами список из десяти или более индивидов. Он становится аргументом функции , то есть возвращается в тело рекурсии.

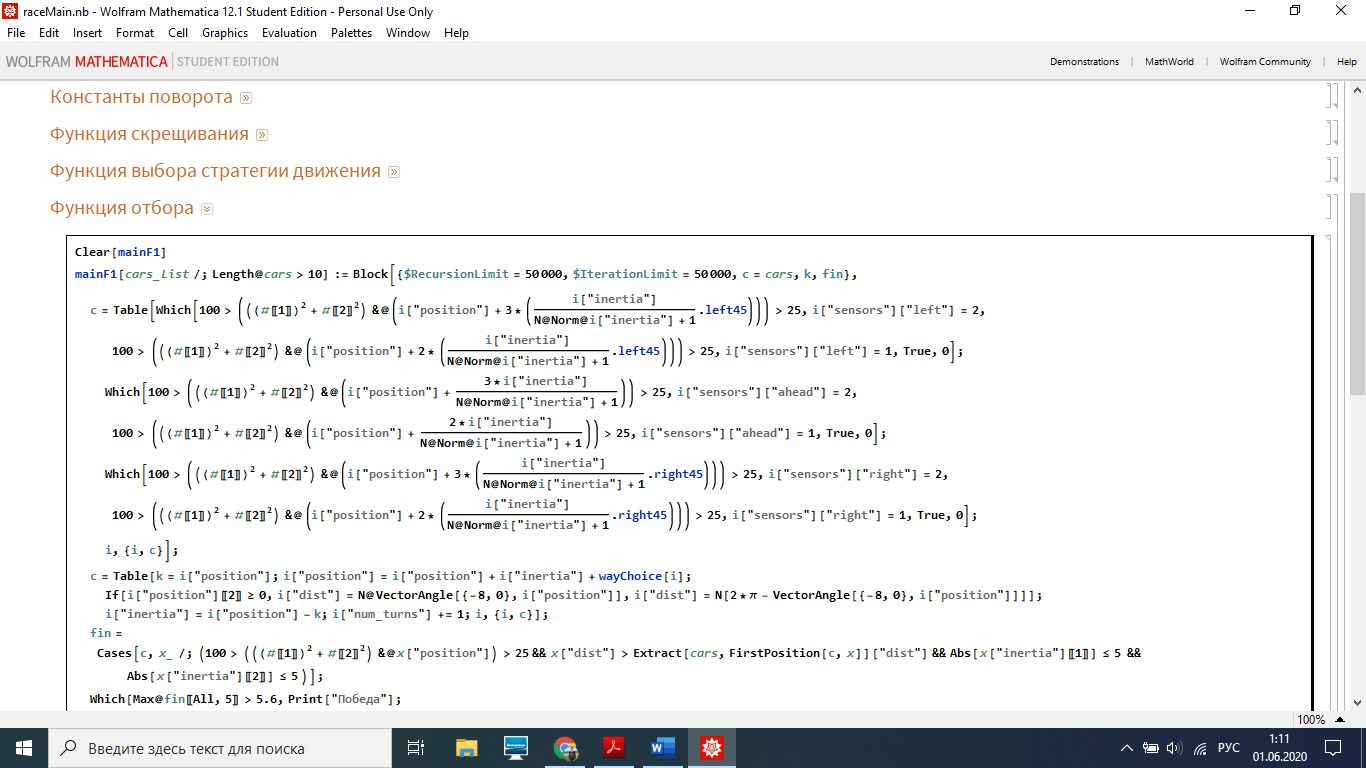


Рисунок 7 – Первая часть функции

Переменная – список индивидов данного этапа

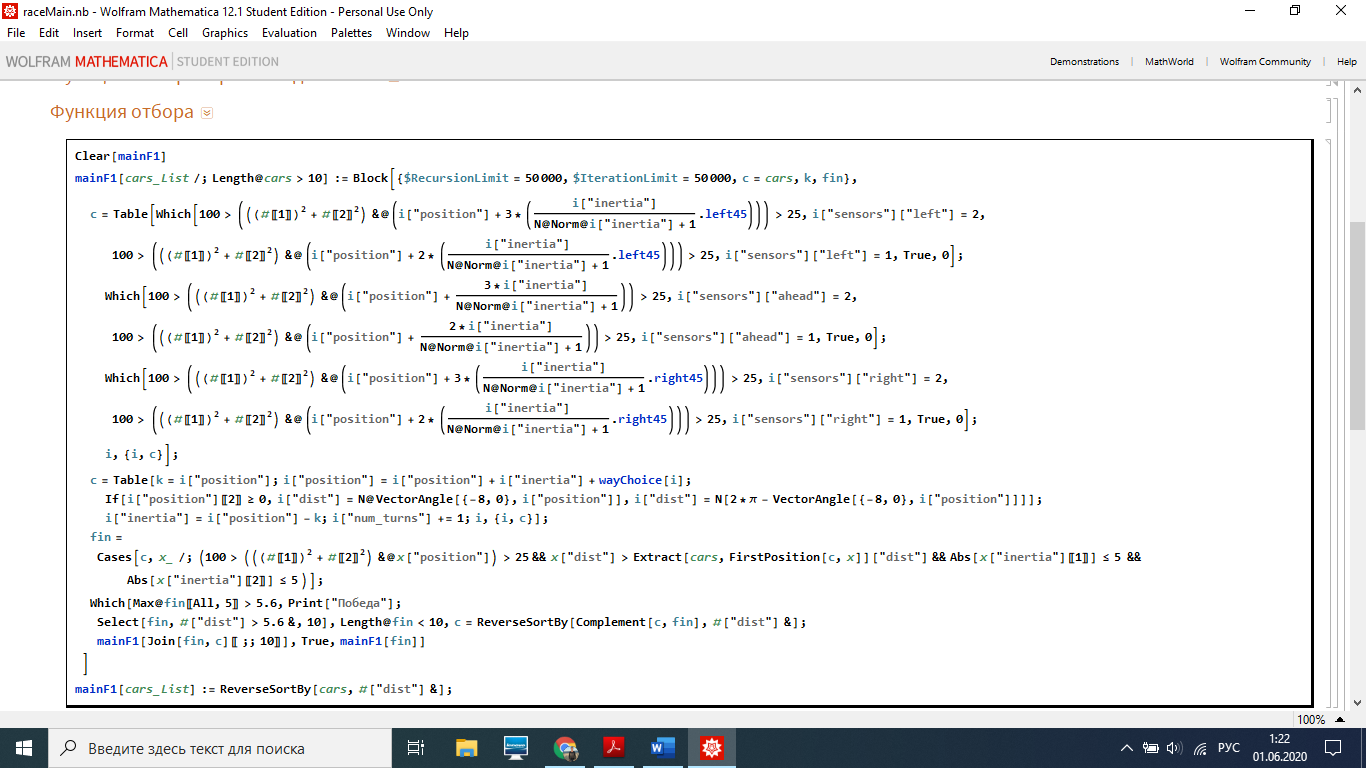


Рисунок 8 – Вторая часть функции

Переменная – список индивидов, полученный в первой части

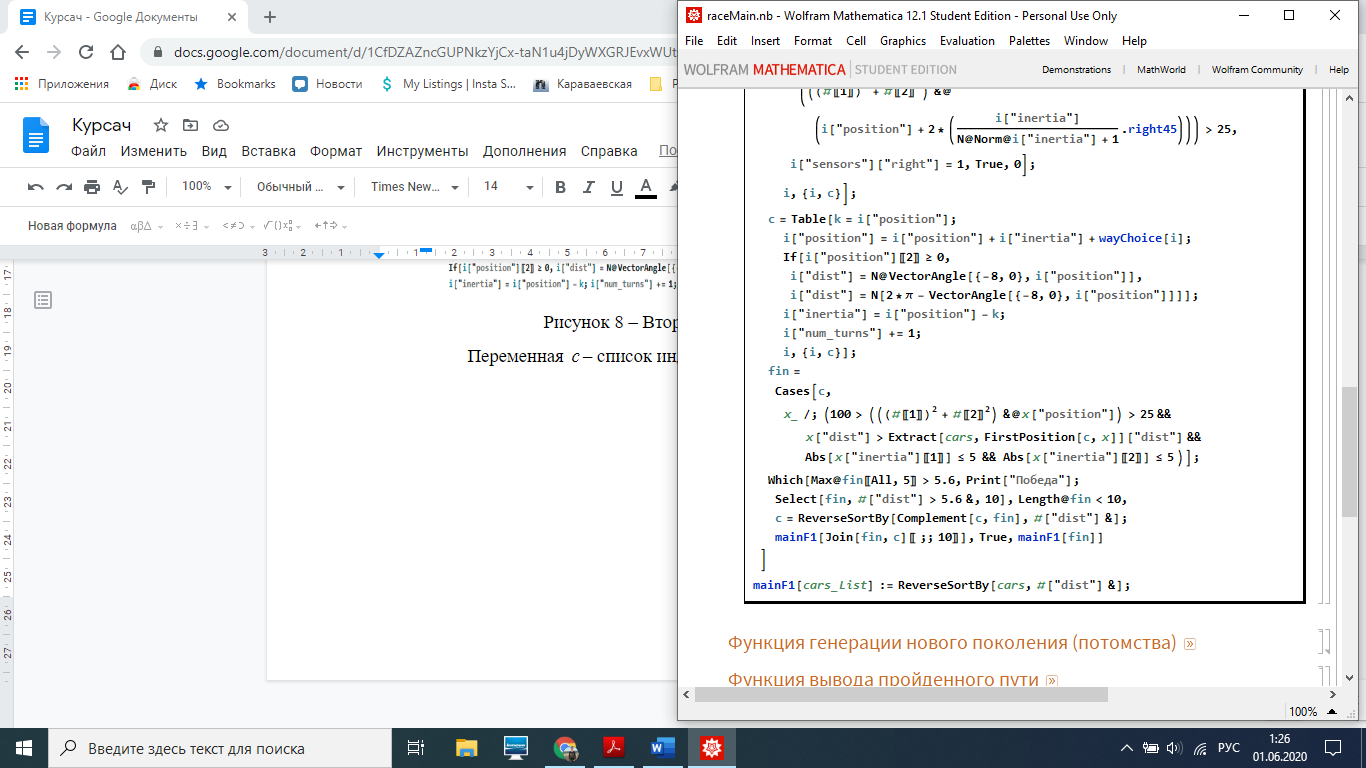


Рисунок 9 – Третья часть функции

Переменная – список индивидов, полученный во второй части

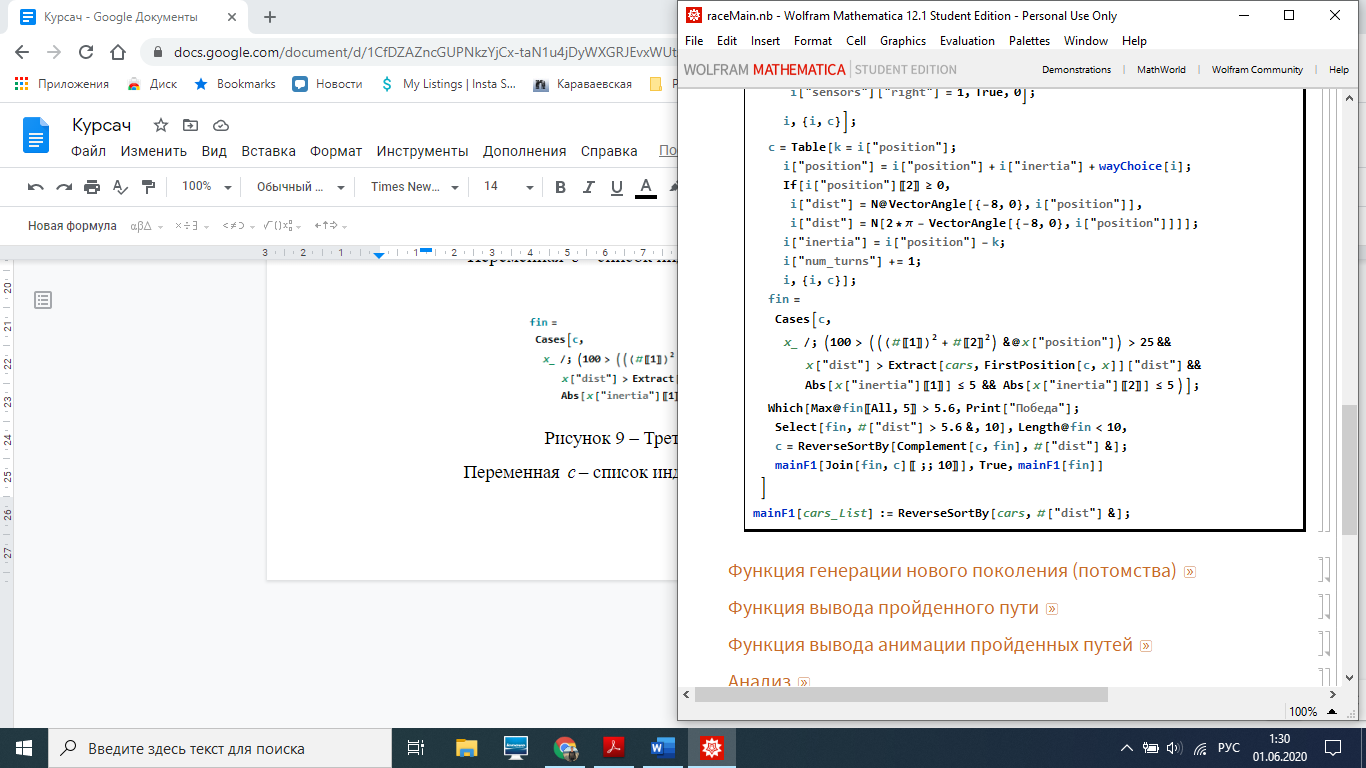


Рисунок 10 – Четвертая часть функции

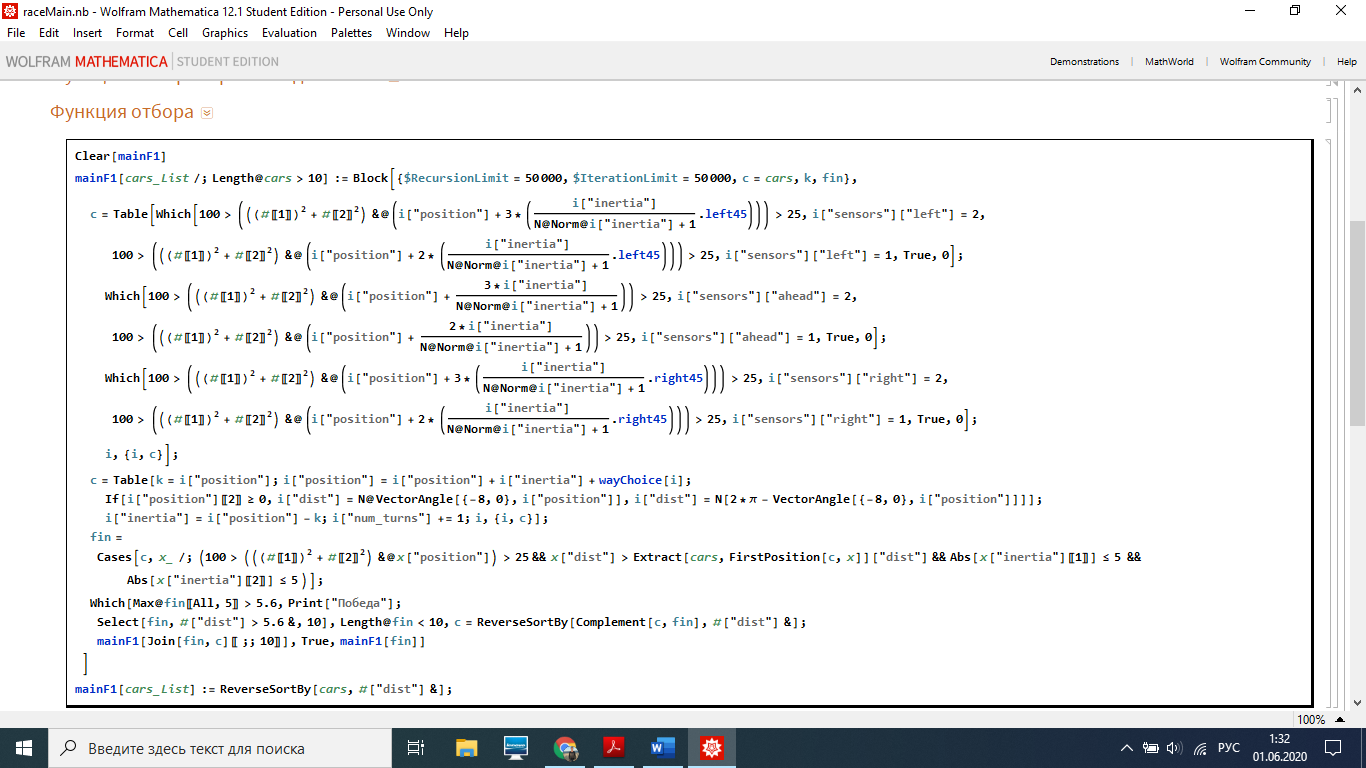


Рисунок 11 – Функция

Функция приспособленности (или ) будет иметь две перегрузки:

1. На вход подается индивид. На выходе получаем значение его

2. На вход подается последовательность из двух индивидов. На выходе получаем сумму их значений

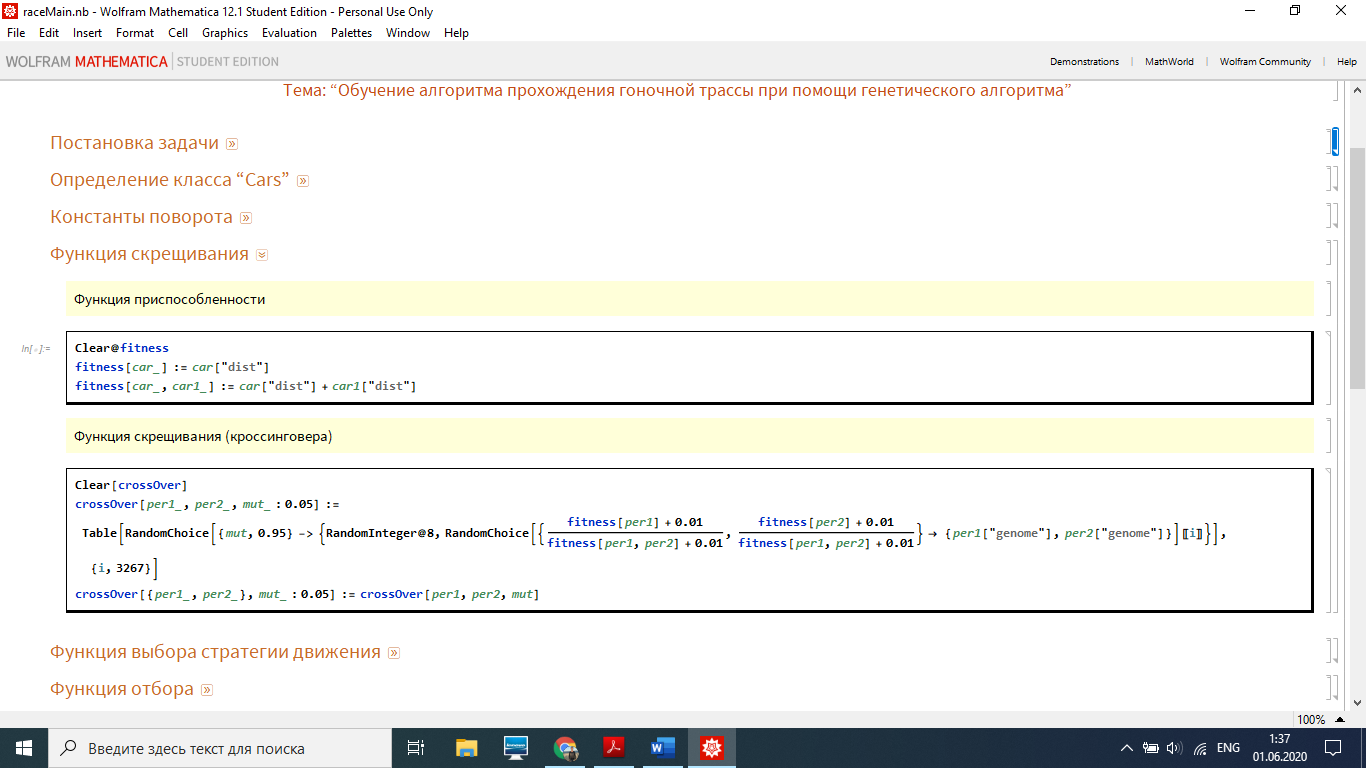


Рисунок 12 – Функция

Функция необходима для того, чтобы определить вероятности выбора генов из геномов двух родителей по следующему правилу:

, где и – индивиды, а и – соответствующие им вероятности. Иными словами, чем больше значение функции у индивида, тем больше он приспособлен к данной гоночной трассе.

Функция скрещивания также будет иметь две перегрузки:

1. На вход подается последовательность из двух обязательных аргументов (два индивида) и одного необязательного аргумента , отвечающего за вероятность мутации генов. Базовое значение равно 0.05.

На выходе – новый геном для будущего потомка

2. На вход подается последовательность из одного обязательного аргумента (список с двумя индивидами) и одного необязательного аргумента , отвечающего за вероятность мутации генов. Базовое значение равно 0.05. На выходе – функция скрещивания от элементов списка и (см. рисунок 13)

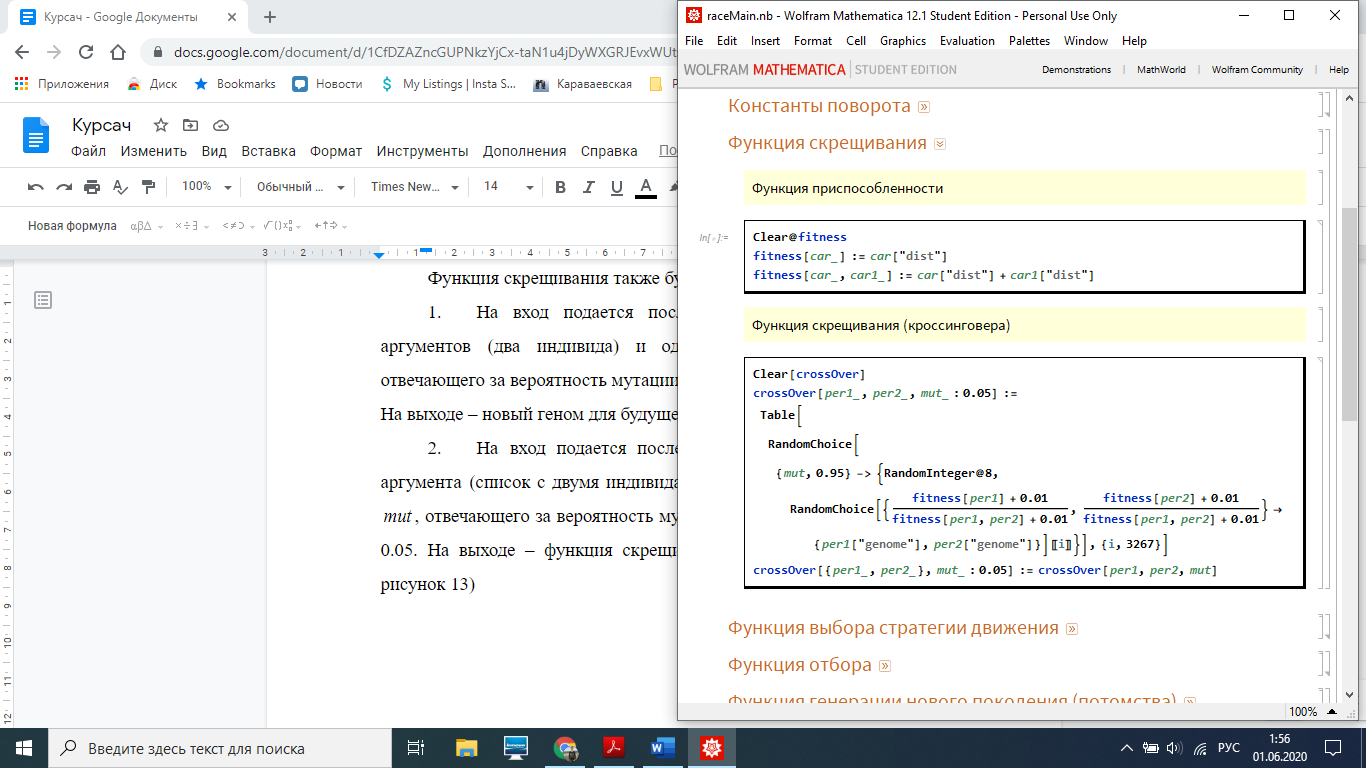


Рисунок 13 – Функция скрещивания

Теперь имеется вся необходимая база для определения функции генерации нового поколения (потомков) . Принцип ее работы заключается в создании 100 объектов типа и присоединения к ним геномов, полученных благодаря отобранным ранее индивидам и функции скрещивания.

Стоит заметить, что в процессе работы алгоритма есть вероятность так называемого «застоя», когда после генерации нового поколения получается список, в котором более половины индивидов идентичны. Во избежание данной ситуации функция имеет вторую перегрузку, в которой происходит «встряска» перед тем, как создавать новое поколение. Суть встряски заключается в удалении дубликатов в списке индивидов и дополнения его произвольными индивидами, у которых значение положено равным двум. Реализация показана на рисунке 14.

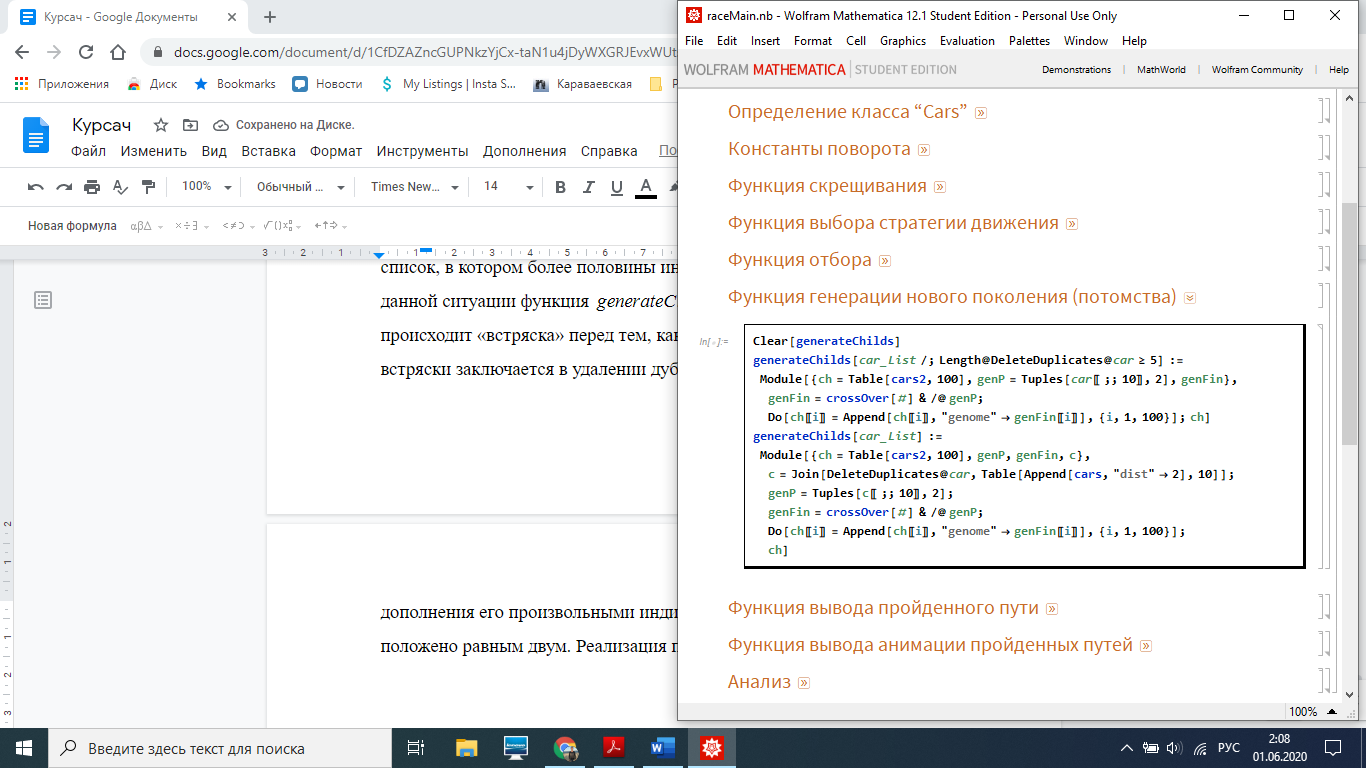


Рисунок 14 – Функция

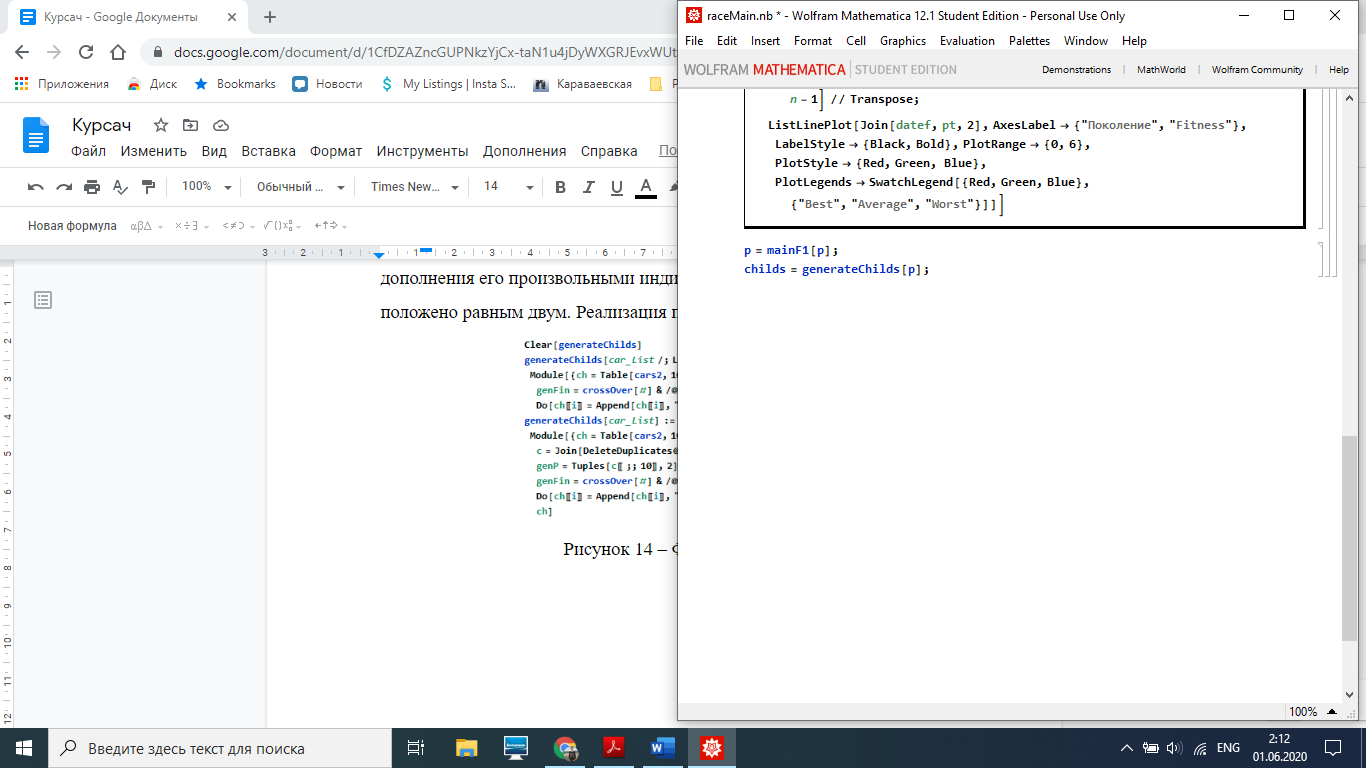


Рисунок 15 – Пример шага генетического алгоритма.

Переменная – некоторое поколение

# **2. ВИЗУАЛИЗАЦИЯ РАБОТЫ АЛГОРИТМА**

## **2.1. Прорисовка путей отобранных в данном поколении индивидов**

Для того, чтобы прорисовать пути, необходимо определить вспомогательную функцию , которая получает на вход индивида и возвращает список координат всех его положений, которые он принимал в процессе выполнения функции отбора. Заметим, что на рисунке 2 в определении классов и фигурирует параметр , а также в функции отбора этот параметр у каждой машинки, которая еще не выбыла, увеличивается на единицу при попадании таковой в тело рекурсии. Благодаря данному параметру, на выходе из рекурсии функции отбора получается список индивидов, в которых содержится информацию о том, сколько раз они перемещались в процессе отбора.

Определим функцию :

Она будет задана рекурсивно. Для входа в рекурсию в качестве аргумента функции будет подаваться индивид, затем в самой функции будет создана его копия, которая будет преобразована. Очевидно, что к моменту передачи индивида в аргумент функции, его последнее положение на гоночной трассе отлично от начального, также как и текущий вектор инерции вместе со значениями сенсоров. Однако необходимо будет изобразить путь машинки из начальной позиции с базовыми значениями указанных величин. Поэтому в созданной копии возвращаются те самые значения указанным параметрам индивида, после чего вызывается функция от трех аргументов: измененная копия индивида, список списков, который на данный момент содержит только координаты начального положения, и значения параметра данного индивида (см. рисунок 16). Тем самым происходит переход в тело рекурсии, в котором данный индивид будет совершать шаги по гоночной трассе в соответствии с функциями, которые имели место в первой и второй части функции отбора за одним исключением. Так как известно точное число шагов, проделанных индивидом в процессе отбора, не нужно менять имеющееся значение параметра , а также точно известно, когда нужно выйти из тела рекурсии. После каждого запуска тела рекурсии, указанное в третьем аргументе значение будет уменьшаться на единицу и указанный во втором аргументе список будет дополняться с конца новым подсписком с позицией индивида в данный момент. Тем самым, когда третий аргумент функции станет равным нулю, второй аргумент будет содержать список всех посещенных машинкой точек, который и выводится.

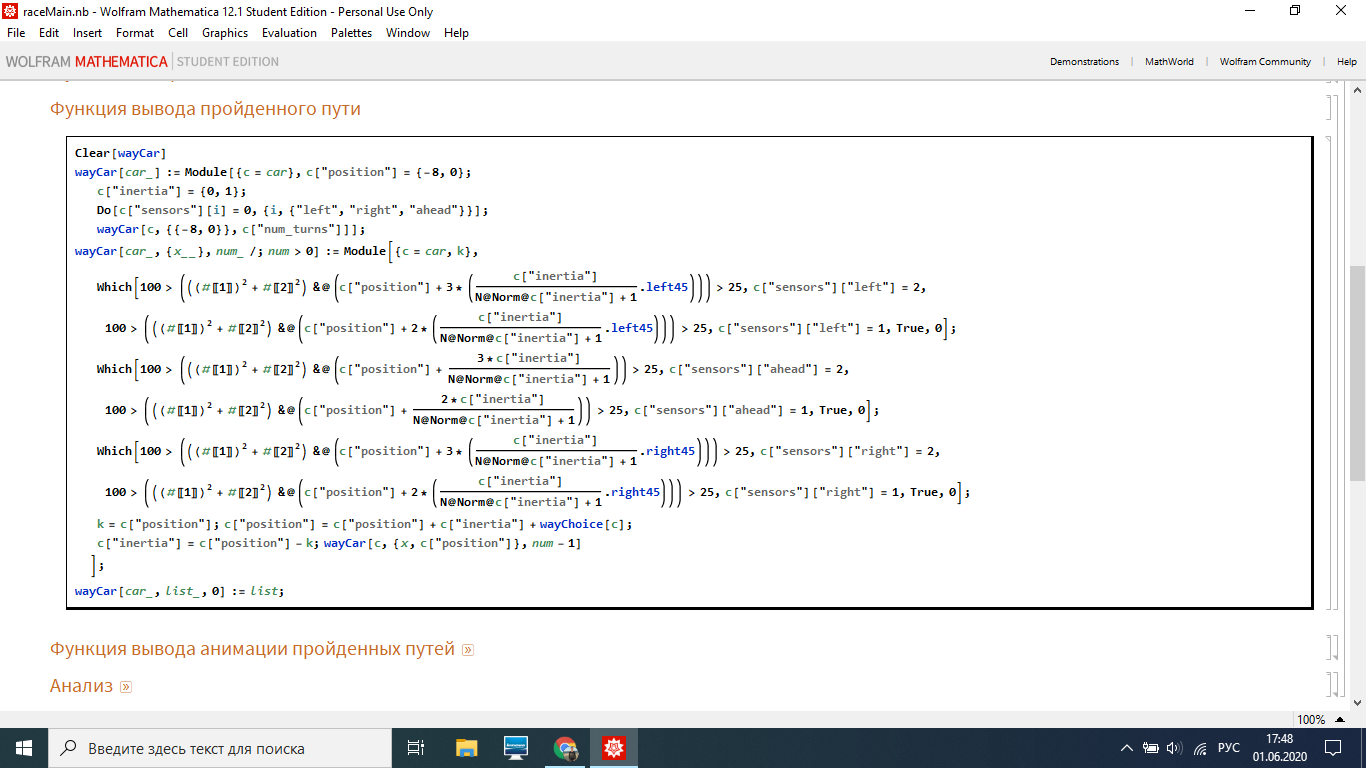


Рисунок 16 – Функцию

Теперь имеется все необходимое для определения функции прорисовки путей отобранных в данном поколении индивидов, как показано на рисунке 17. Преимуществом функции отбора является то, что по завершении процесса отбора выводится ровно 10 индивидов, лучших индивидов по критерию приспособленности. Единственная ситуация, когда может вернуться меньшее число индивидов – это наличие обученной машинки среди отобранных индивидов, о чем сразу же скажет сообщение «Победа» по завершении процесса отбора.



Рисунок 17 – Функция

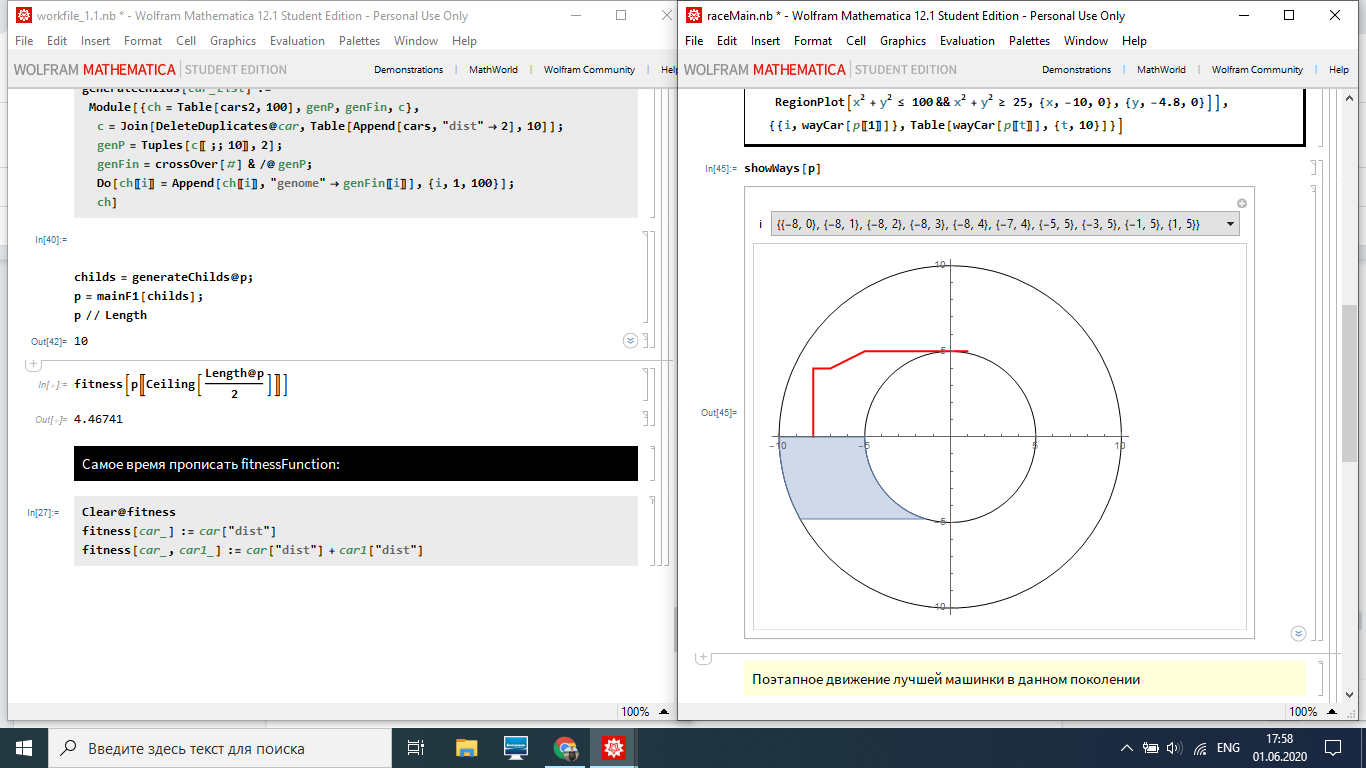


Рисунок 18 – Пример работы функции

Переменная – некоторое поколение

## **2.2. Поэтапное движение лучшего индивида данного поколения**

Докажем лемму о том, что если в списке отобранных индивидов присутствует хотя бы одна обученная машинка, то на первой позиции данного списка заведомо будет стоять обученная машинка. Известно, что функция отбора по завершении работы всегда выводит список индивидов в порядке убывания по критерию приспособленности (см. рисунок 11), то есть на первом месте стоит самая приспособленная машинка. Пусть среди отобранных индивидов есть обученная машинка, но на первой позиции в списке стоит машинка, не являющаяся обученной, то есть индивид, неспособный дойти до зоны финиша. Тогда имеем противоречие, так как функция приспособленности для такой машинки вернет значение строго меньшее значения функции приспособленности для обученной машинки, но индивиды в списке расположены в порядке убывания по критерию приспособленности.

Тем самым доказано, что при наличии хотя бы одной обученной машинки в списке отобранных индивидов, на первой позиции обязательно будет стоять обученная машинка. И было бы интересно изучить пошагово траекторию ее движения от начальной позиции до зоны финиша. В этом поможет функция , определенная как показано на рисунке 19.

На экран выводится окно с ползунком, меняя положение которого, можно пошагово изучать движение обученной машинки (см. рисунок 20)

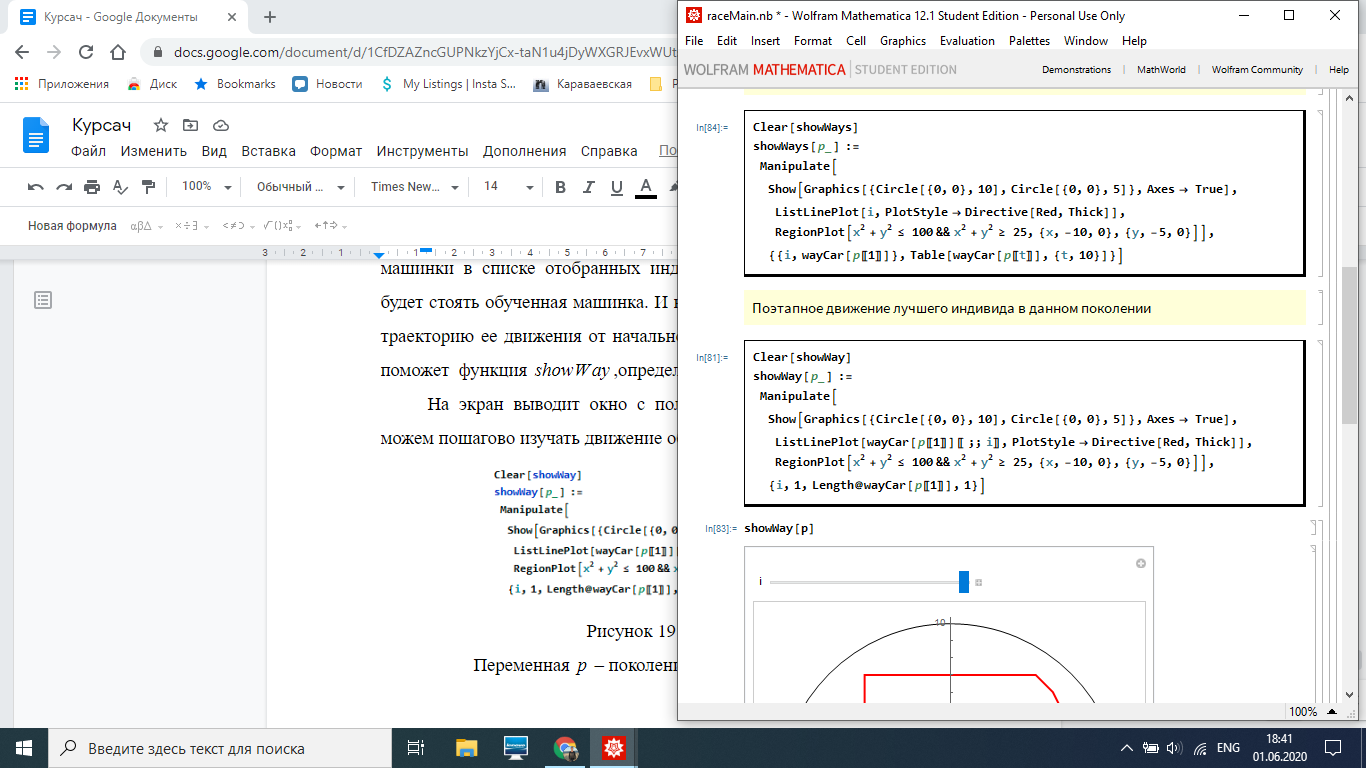


Рисунок 19 – Функция

Переменная – поколение, содержащее обученную машинку

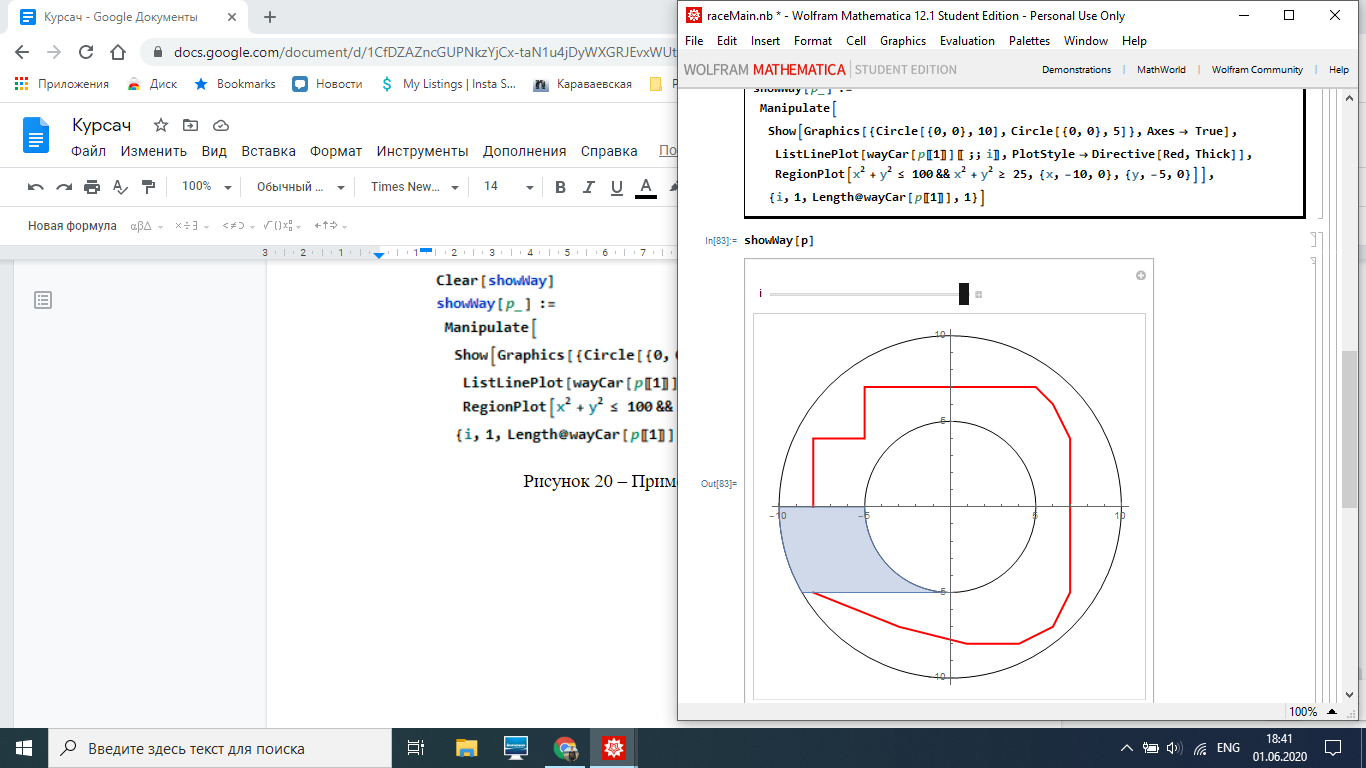


Рисунок 20 – Пример работы функции

# **3. АНАЛИЗ РАБОТЫ АЛГОРИТМА**

## **3.1. Проверка выдвинутой гипотезы**

В самом начале работы к рассмотрению была предложена гипотеза, что генетический алгоритм действительно способен обучить машинки проходить гоночную трассу.

Очевидно, что генетический алгоритм в представленном виде действительно обучает индивидов, однако как убедиться в успешности данного обучения? Заметим, что качество обучения напрямую зависит от показателей функции приспособленности индивидов в поколениях.

Тогда если с ростом натурального числа показатели приспособленности у также растут, это и будет означать, что генетический алгоритм для поставленной задачи работает успешно.

На рисунке 21 представлен график, оси которого соответствуют номеру поколения и значению . Красная линия соответствуют лучшему индивиду каждого из рассмотренных поколений, зеленая – среднему, а синяя – худшему. Данный график получен с помощью функции , показанной на рисунке 22.



Рисунок 21 – График приспособленностей для некоторого поколения

График рисунка 21 соответствует некоторому произвольному начальному поколению и его дальнейшему развитию вплоть до .Заметим, что уже в появилась обученная машинка, причем, судя по всему, состоит из нее единственной. Как это можно понять? На происходит резкий спад значений . Это произошло из-за «встряски», которая была необходима для появления новых индивидов . Иными словами, к обученной машинке добавилось еще 9 случайных индивидов, и на полученному наборе из 10 индивидов было «взращено» , не очень удачное поколение. Однако далее, к 9 поколению, вновь появляются обученные машинки.

Тем самым можно сделать вывод, что представленный алгоритм, достигая поколения, которое содержит обученную машинку, в последующих поколения будет по-прежнему возвращать обученные машинки после конечного числа предшествующих поколений без таковых.

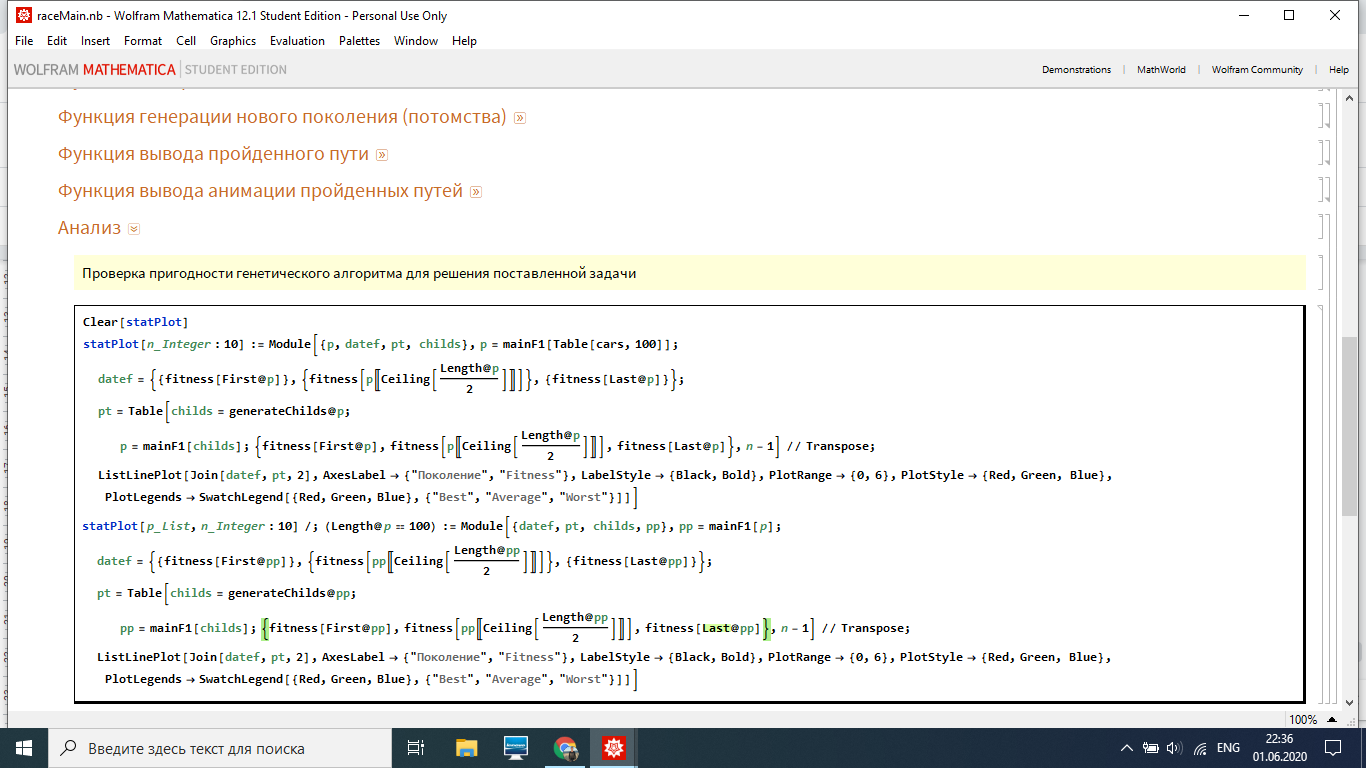


Рисунок 22 – Функция

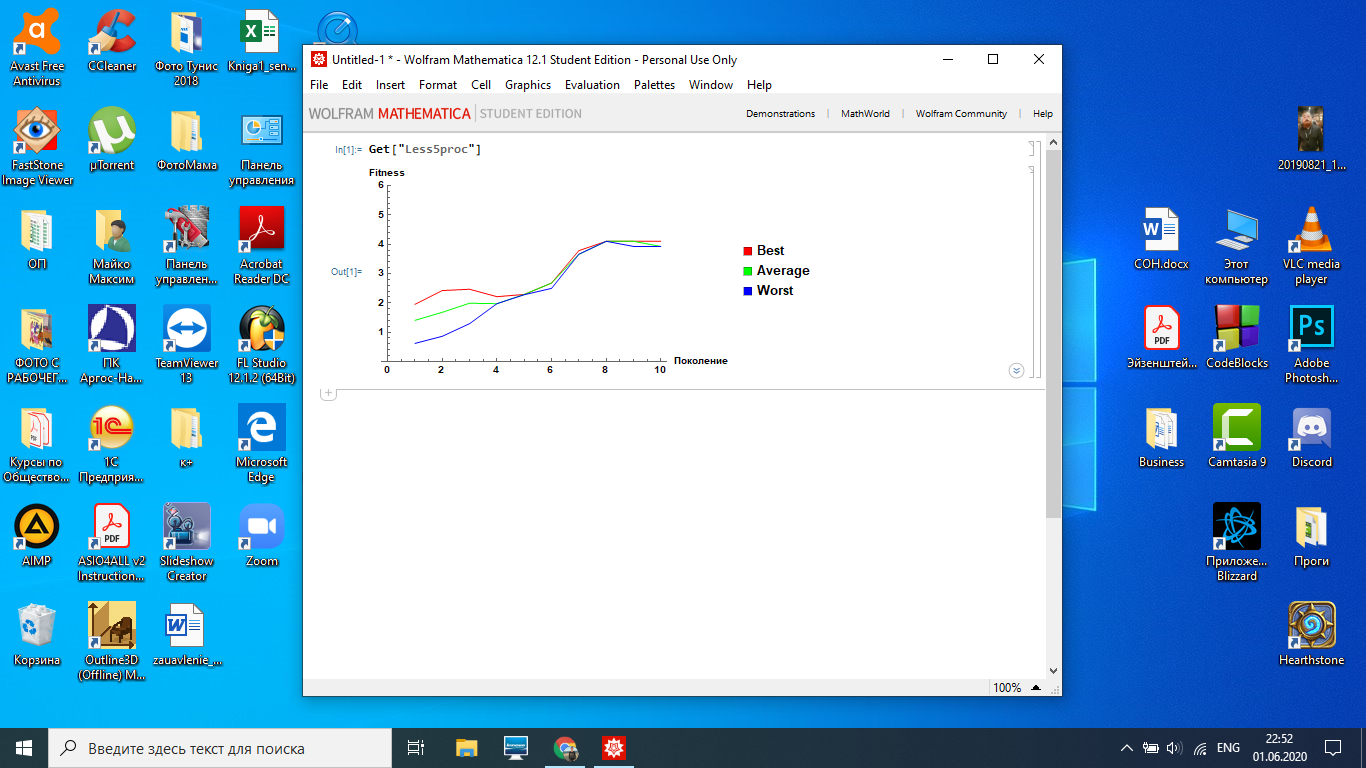


Рисунок 23 – Проблема стагнации при значении мутации меньше 0.05

# **ЗАКЛЮЧЕНИЕ**

В ходе проделанной работы был реализован алгоритм обучения прохождения гоночной трассы, где за основу были взяты идеи генетического алгоритма. Более того, был найден универсальный для поставленной задачи способ измерения пройденного индивидом расстояния, которое используется в ключевой функции приспособленности. Это весомое достижение, так как изначально рассматривались другие варианты, которые также оказались работоспособными, однако накладывали некоторые существенные ограничения на саму гоночную трассу. Первым вариантом было определить для каждой точки трассы расстояния до зоны финиша, то есть составить разреженную матрицу для каждого из предлагаемых треков. При таком варианте появляется естественное ограничение на размеры гоночной трассы, так как матрица требует дополнительной затраты памяти. Вторым вариантом стало проведение средней линии по всей гоночной трассе, измерение ее длины и затем проекция текущих позиций индивидов на нее с целью оценки уже пройденного расстояния. В таком случае становятся невозможными для рассмотрения извилистые гоночные трассы, так как в них появляется двусмысленность проектирования. Третьим вариантом было измерение площади пройденной индивидом части трассы, что накладывало естественную вычислительную нагрузку.

Все вышесказанное подтверждает весомость выбора в пользу измерения угла, так как для любой из рассматриваемых гоночных трасс всегда возможно определить центр, начальную точку и позиции индивидов. Кроме того, машинки двигаются в строго заданном направлении.

В ходе анализа работы представленного алгоритма удалось проверить выдвинутую гипотезу о пригодности генетического алгоритма для обучения индивидов и рассмотреть случай с необходимостью использования «встряски», после чего был сделан вывод, что ситуация затем не поворачивается в худшую сторону.

# **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Robert Olsson. A Genetic Algorithm in the Game Racetrack / Robert Olsson, Andreas Tarandi // KTH, School of Computer Science and Communication, Kandidat Exjobb CSC; K11062. - 2011 г.

2. Компьютерная система Wolfram Mathematica [Электронный ресурс] / <http://www.wolfram/com/mathematica/>