**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «Информатика»**

**Тема:** [**Основные управляющие конструкции языка Python**](https://e.moevm.info/mod/quiz/view.php?id=2018)

|  |  |  |
| --- | --- | --- |
| Студент гр. 3342 |  | Лапшов К.Н. |
| Преподаватель |  | Иванов Д.В. |

Санкт-Петербург

2023

## Цель работы

Целью работы является освоение работы с функциями в языке python и с библиотекой numpy.

## Задание

Вариант 2.

Задача 1.

Оформите задачу как отдельную функцию: def check\_rectangle(robot, point1, point2, point3, point4) На вход функции подаются: координаты дакибота robot и координаты точек, описывающих перекресток: point1, point2, point3, point4. Точка -- это кортеж из двух целых чисел (x, y).Функция должна возвращать True, если дакибот на перекрестке, и False, если дакибот вне перекрестка.

Задача 2.

Оформите решение в виде отдельной функции check\_collision(). На вход функции подается матрица ndarray Nx3 (N -- количество ботов, может быть разным в разных тестах) коэффициентов уравнений траекторий coefficients. Функция возвращает список пар -- номера столкнувшихся ботов (если никто из ботов не столкнулся, возвращается пустой список).

Задача 3.

Оформите задачу как отдельную функцию check\_path, на вход которой передается последовательность (список) двумерных точек (пар) points\_list. Функция должна возвращать число -- длину пройденного дакиботом пути (выполните округление до 2 знака с помощью round(value, 2)).

## Выполнение работы

Данная программа написана на языке Python с использованием библиотеки numpy. Она состоит из 3-функций, которые вызываются сразу на сайте https://e.moevm.info.

Первая функция check\_crossroad. Функция возвращает True, если дакибот на перекрестке, и False, если дакибот вне перекрестка. Перекресток определяется 4 данными на входе точками. Для выполнения это функции необходимо сравнить координаты робота и координаты точек перекрестка, находя его границы.

Вторая функция check\_collision. Функция возвращает список пар номеров столкнувшихся ботов в виде кортежей. Если никто из ботов не столкнулся, возвращается пустой список. Для реализации этой функции были использованы два цикла, где переменные-итераторы являются индексами строк матрицы с коэффициентами линейных уравнений. Внутри этих циклов создаются массивы, в которые записываются коэффициенты соответствующих строк матрицы. Затем создается матрица, содержащая эти два массива. С помощью функции linalg.matrix\_rank из модуля numpy вычисляется ранг матрицы, который определяет, есть ли пересечения у двух линейных функций. Если пересечения есть, значит роботы столкнулись, и соответствующие индексы строк с коэффициентами записываются в массив result. По окончании всех итераций функция возвращает массив result.

Третья функция check\_path. Функция принимает список точек points\_list и вычисляет длину пути, проходящего через эти точки. Для этого используется функция модуля numpy - linalg.norm которая высчитывает длину вектора. Результат вычислений округляется до двух знаков после запятой и возвращается в виде числа с плавающей точкой.

Переменные, используемые в программе:

- x\_line\_min, x\_line\_max, – ограничение перекрестка по ширине.

- y\_line\_min, y\_line\_max, – ограничение перекрестка по высоте.

- index – переменная, используемая в первом цикле функции check\_collision.

-index\_interior – переменная, используемая во внутреннем цикле функции check\_collision.

- result – список из картежей с номерами столкнувшихся дакиботов.

- distance – сумма длин путей дакибота.

Функции, используемые в этой программе:

-numpy.array возвращает массив типа numpy.ndarray.

-numpy.linalg.matrix\_rank возвращает ранг матрицы.

-numpy.linalg.norm возвращает длину вектора.

-round возвращает округленное число до выбранного значения.

Данная программа демонстрирует использование функций библиотеки numpy и работу функций на языке Python для выполнения различных математический операций.

Разработанный программный код см. в приложении А.

## Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | (15, 16), (5, 10) ,(20, 10), (20, 20), (5, 20) | True |  |
|  | [[-1 5 7]  [1 -5 7]  [9 3 32]] | [(0, 1), (0, 2), (1, 0), (1, 2), (2, 0), (2, 1)] |  |
|  | [(2.0, 2.0), (7.0, 8.0)] | 7.81 |  |

## Выводы

Исследованы принципы работы с функциями в языке Python и использование библиотеки NumPy.

Разработаны функции, которые вычисляют решения конкретных математических задач.

# Приложение А Исходный код программы

Название файла: main.py

import numpy as np

def check\_crossroad(robot, point1, point2, point3, point4):

x\_line\_min, y\_line\_min = point1

x\_line\_max, y\_line\_max = point3

if x\_line\_min <= robot[0] <= x\_line\_max and y\_line\_min <= robot[1] <= y\_line\_max:

return True

else:

return False

def check\_collision(coefficients):

result = []

for index in range(len(coefficients)):

numpay\_arr = coefficients[index][0:2]

for index\_interior in range(len(coefficients)):

numpay\_arr\_insert = coefficients[index\_interior][0:2]

matrix = np.array([numpay\_arr, numpay\_arr\_insert])

rank\_of\_matrix = np.linalg.matrix\_rank(matrix)

if rank\_of\_matrix == len(numpay\_arr):

result.append((index, index\_interior))

return result

def check\_path(points\_list):

distance = 0

for index in range(len(points\_list) - 1):

if index + 1 == len(points\_list):

break

else:

point\_a = np.array(points\_list[index])

point\_b = np.array(points\_list[index + 1])

distance += np.linalg.norm(point\_a - point\_b)

return round(distance, 2)