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

Студент гр. 3342 _____ Русанов А.И.

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. Программа состоит из трех функций.

Первая функция check_crossroad возвращает True, если дакибот находится на перекрестке, и False, если дакибот находится вне перекрестка. Перекресток определяется 4 точками, которые передаются в качестве входных данных. Для реализации данной функции необходимо сравнить координаты дакибота с координатами перекрестка.

Вторая функция check_collision. Функция возвращает список пар в виде кортежей - номера столкнувшихся ботов (если никто из ботов не столкнулся, возвращается пустой список). Для ее реализации были использованы два цикла, переменные-итераторы которых являются индексами строк матрицы с коэффициентами линейных уравнений. Внутри циклов создаются массивы, в которые записываются коэффициенты соответствующих строк матрицы. Затем создается матрица, которая содержит в себе эти два массива. С помощью функции из модуля numpy linalg.matrix_rank вычисляется ранг матрицы, с помощью которого определяется факт наличия пересечения у двух линейных функций. Если пересечения имеются – значит робот столкнулся, и в массив collisions записываются соответствующие индексы строк с коэффициентами. После завершения цикла функция возвращает массив collisions.

Третья функция check_path принимает список точек "points_list" и вычисляет длину пути, проходящего через эти точки. Для этого используется формула расстояния между двумя точками на плоскости. С помощью функции round() результат вычислений округляется до двух знаков после запятой и возвращается в виде числа с плавающей точкой.

Переменные, используемые в программе:

- collisions – список из кортежей с номерами столкнувшихся дакиботов.
- path_length – сумма длин путей дакибота.

Функции, используемые в программе:
-numpy.linalg.matrix_rank() возвращает ранг матрицы.
-numpy.array() возвращает массив типа numpy.ndarray.
-round() возвращает округленное число до выбранного значения.

Разработанная программа демонстрирует использование функций библиотеки numpy, а также 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
1.	(5, 8) (0, 3) (12, 3) (12, 16) (0, 16)	True	
2.	[[-1 -4 0] [-7 -5 5] [1 4 2] [-5 2 2]]	[(0, 1), (0, 3), (1, 0), (1, 2), (1, 3), (2, 1), (2, 3), (3, 0), (3, 1), (3, 2)]	
3.	[(2.0, 3.0), (4.0, 5.0)]	2.83	

Выводы

Были изучены правила работы с функциями в языке python и работа с библиотекой numpy.

Разработаны функции, возвращающие решения определенных математических заданий.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
import numpy as np


def check_crossroad(robot, point1, point2, point3, point4):
    return all([point1[i] <= robot[i] <= point3[i] for i in (0, 1)])


def check_collision(coefficients):
    collisions = []
    for i in range(len(coefficients)):
        arr_i = coefficients[i][:2]
        for j in range(len(coefficients)):
            if i != j:

                arr_j = coefficients[j][:2]
                matrix = np.array([arr_i, arr_j])

                if np.linalg.matrix_rank(matrix) == 2:
                    collisions.append((i, j))

    return collisions


def check_path(points_list):
    path_length = 0
    for i in range(len(points_list) - 1):
        x0 = points_list[i][0]
        x1 = points_list[i + 1][0]
        y0 = points_list[i][1]
        y1 = points_list[i + 1][1]
        path_length += ((x1 - x0) ** 2 + (y1 - y0) ** 2) ** 0.5
    path_length = np.round(path_length, 2)
    return path_length
```