МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский ядерный университет «МИФИ»
(НИЯУ МИФИ)
Институт Интеллектуальных Кибернетических Систем
Кафедра Кибернетики

Лабораторная работа №3:

По курсу «Численные методы»

Вариант 16

Работу выполнил: студент группы Б17-511: Чудновец И.В.

Проверил: Саманчук В.Н.

Москва 2019

Постановка задачи

Интерполировать таблично заданную функцию, используя полином Лагранжа 5-ого порядка:

X	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Y	5	6	8	10	12	13	12	10	8	10	8	11	7	9	11	10	9	12	11	6

Методика решения

Для решения задачи была написана программа на языке Python, в которой реализован алгоритм интерполирования таблично заданной функции, используя полином Лагранжа произвольного порядка.

Теоретическая справка

Метод интерполяции Лагранжа заключается в поиске n+1 многочленов для каждой пары из n+1 заданных узлов, в которых определена функция.

$$L_j(x) = \begin{cases} 1, & x = x_j \\ 0, & x = x_i \end{cases} \quad j = 1, \cdots, n+1; i = 1, \cdots, n+1$$

$$y(x) = \sum_{j=1}^{n+1} L_j(x) * y_j$$

где $y(x)$ – многочлен n-ой степени, интерполирующий табличную функцию,

$$L_j(x) = \frac{(x - x_1) * (x - x_2) * \cdots * (x - x_{j-1}) * (x - x_{j+1}) * \cdots * (x - x_{n+1})}{(x_j - x_1) * (x_j - x_2) * \cdots * (x_j - x_{j-1}) * (x_j - x_{j+1}) * \cdots * (x_j - x_{n+1})}$$

Решение задачи

interpolation.py

```
# -*- coding: utf-8 -*-
from sympy import *
init_printing()


class Interpolate:
    @staticmethod
    def build_Lagrange(x_, x_list, y_list):
        if len(x_list) != len(y_list):
            return
        res = 0
        for j, y_j in enumerate(y_list):
            r = 1
            x_j = x_list[j]
            for i, x_i in enumerate(x_list):
                if i != j:
                    r *= (x_ - x_i) / (x_j - x_i)
```

```
            res += r * y_j
        return res

    @staticmethod
    def get_y_values(x_, x_list, y_list, n, accuracy):
        # n - количество точек для интерполяции
        # для полинома 5-ой степени n = 6
        # accuracy - количество промежуточных точек между
        # узлами (степень гладкости)
        if len(x_list) != len(y_list):
            return
        number_intervals = len(x_list) - 1
        # Интерполяция происходит следующим образом:
        # несколько левых(left) интервалов первым полиномом
        # интервалы в центре - по одному на каждый новый полином
        # правые(right) интервалы - последним полиномом
        left = n // 2
        right = n - left
        x_values = []
        y_values = []
        # Параметры для запуска 3-х циклов:
        # (number_intervals, l_from, l_to, disp)
        # disp = число пройденных интервалов
        interpolate_parts = [(left, 0, n, 0),
                             *[(1, i + 1, n + 1 + i, left + i) for i in
range(number_intervals - n)],
                             (right, -n, len(x_list), number_intervals -
right)]
        for param in interpolate_parts:
            disp = param[3]
            l_from = param[1]
            l_to = param[2]
            lagrange = __class__.build_Lagrange(x_, x_list[l_from:l_to],
y_list[l_from:l_to])
            # Итерируемся по интервалам
            for i in range(disp, param[0] + disp):
                x_values.append(x_list[i])
                y_values.append(y_list[i])
                step = (x_list[i + 1] - x_list[i]) / (accuracy + 1)
                for j in range(1, accuracy + 1):
                    x_values.append(x_list[i] + step * j)
                    y_values.append(lagrange.subs(x_, x_list[i] + step * j))
        x_values.append(x_list[-1])
        y_values.append(y_list[-1])
        return x_values, y_values
```

test_interpolation.ipynb

```
from interpolation import Interpolate
import matplotlib.pyplot as plt
import pylab
pylab.rcParams['figure.figsize'] = (15.0, 10.0)
plt.rcParams.update({'font.size': 22})

from sympy import *
init_printing()

var('x')
---
y_20 = [5, 6, 8, 10, 12, 13, 12, 10, 8, 10, 8, 11, 7, 9, 11, 10, 9, 12, 11, 6]
x_20 = list(range(1, len(y_20) + 1))
```

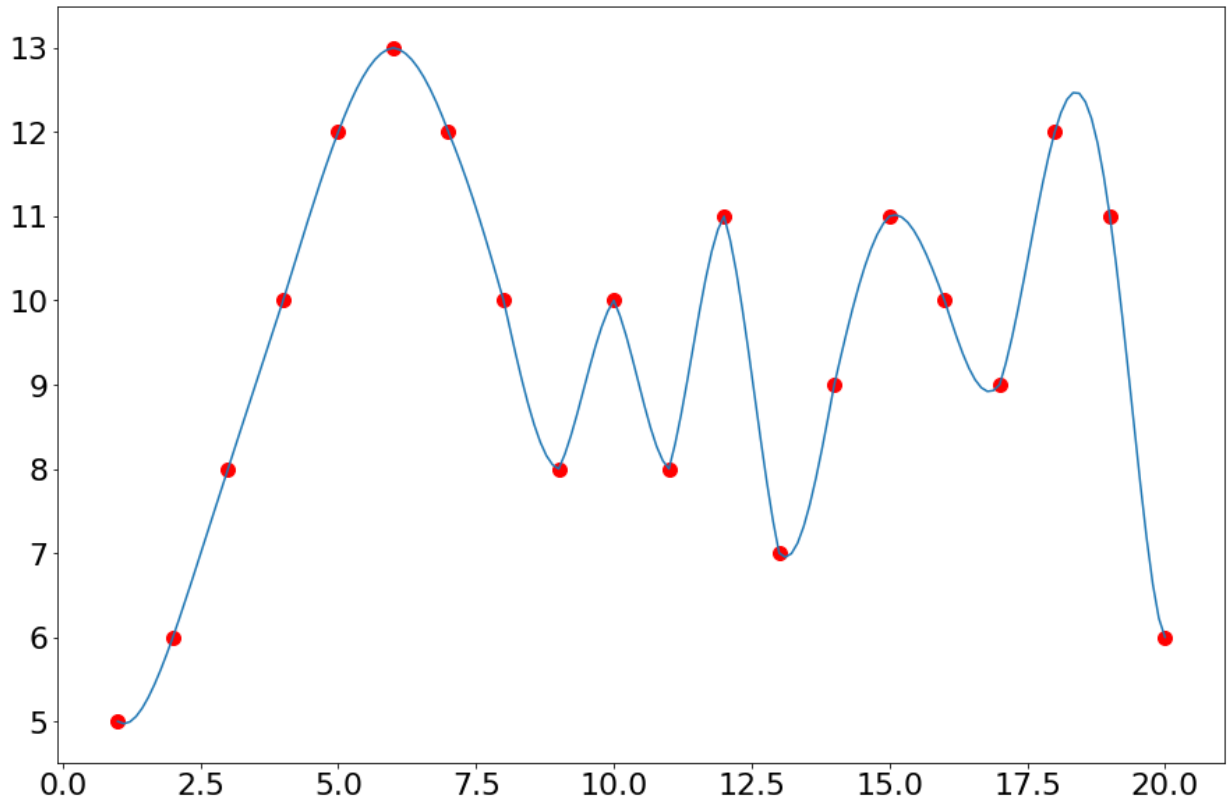
```
display(y_20, x_20)
---
#Интерполяция полиномом 5-ой степени
#8 точек между узлами
x_list, y_list = Interpolate.get_y_values(x, x_20, y_20, 6, 8)
plt.plot(x_list, y_list)
plt.scatter(x_20, y_20, color='red', s=100)
```

Условные обозначения:

--- - означает, что код в Jupyter Notebook разделён в разные ячейки

Результат работы

```
[5, 6, 8, 10, 12, 13, 12, 10, 8, 10, 8, 11, 7, 9, 11, 10, 9, 12, 11, 6]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]
<matplotlib.collections.PathCollection at 0x71e5a58>
```



Заключение

В работе требовалось интерполировать таблично заданную функцию, используя полином Лагранжа 5-ого порядка. Для решения данной задачи была написана программа. Результат представлен в виде скриншота выше.