	Министерство науки и высшего образования Российской Федерации **Федеральное государственное бюджетное образовательное учреждение высшего образования** **«Московский государственный технический университет имени Н.Э. Баумана** **(национальный исследовательский университет)»** **(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 2

Тема: Построение и программная реализация алгоритма многомерной интерполяции табличных функций.

Студент: Козлова И. В.

Группа: ИУ7-42Б

Оценка (баллы): _______________

Преподаватель: Градов В.М.

Москва
2020 г

Цель работы: Получение навыков построения алгоритма интерполяции таблично заданных функций двух переменных.

Алгоритм решения

Разделенные разности для полинома Ньютона:

$$y(x_i, x_j) = \frac{y_i - y_j}{x_i - x_j}$$

Для двух точек:

$$y(x_i, x_j, x_k) = \frac{y(x_i, x_j) - y(x_j, x_k)}{x_i - x_k}$$

Для трех точек:

Отсюда выводим для любого кол-ва точек:

$$y(x_i, x_j, \ldots, x_n) = \frac{y(x_i, x_j, \ldots, x_{n-1}) - y(x_j, \ldots, x_n)}{x_i - x_n}$$

Далее можно написать формулу для самого **полинома Ньютона:**

$$P_n(x) = y_0 + (x - x_0)y(x_0, x_1) + (x - x_0)(x - x_1)y(x_0, x_1, x_2) + \ldots$$
$$+ (x - x_0)(x - x_1)\ldots(x - x_{n-1})y(x_0, x_1, \ldots, x_n)$$

$$P_n(x) = y_0 + \sum_{k=0}^{n}(x - x_0)\ldots(x - x_{n-1})y(x_0, x_1, \ldots, x_k)$$

Алгоритм для многомерной интерполяции:

Необходимо найти z(x, y)

1. Интерполяция по строкам - Необходимо найти z(x, yk), где k от 0 до ny. Для каждого k результат необходимо сохранить в массив из k элементов.
2. Интерполяция по столбцам - Далее находим z(x, Lk) - результат нашей программы.

Исходные данные

1. Таблица функции с количеством узлов 5х5.

y x	0	1	2	3	4
0	0	1	4	9	16

1	1	2	5	10	17
2	2	5	8	13	20
3	4	10	13	18	25
4	16	17	20	25	32

2. Степень аппроксимирующих полиномов - nx и ny .

3. Значение аргументов x, y, для которого выполняется интерполяция.

Код программы

```
import math
import openpyxl as xls

def F(x, y):
   '''
       Функция, используемая в программе.
   '''
   return x ** 2 + y ** 2

def parse_table(name):
   '''
       Загрузка таблицы в программу.
   '''
   try:
       pos = 1
       points = xls.load_workbook(name).active
       table = []
       while points.cell(row = pos, column = 1).value is not None:
           buf = []
           for i in range(1, 7):
               buf.append(float(points.cell(row = pos, column = i).value))
           table.append(buf)
           pos += 1

       # Разбиение на нужные массивы (X, Y, Z)
       x = []
       y = []
       z = []
       for i in range(1, len(table)):
           buf = []
           x.append(table[i][0])
           y.append(table[0][i])
           for j in range(1, len(table)):
               buf.append(table[i][j])
           z.append(buf)
       print(x, y, z)
```

```
        return z, x, y

    except TypeError:
        print("Проверьте данные на вводе!!!")
        return None, None, None
    except ValueError:
        print("Проверьте данные на вводе!!!")
        return None, None, None

def find_x0_xn(data, power, arg):
    '''
        Нахождение начального и конечного индекса в
таблице (x/y0 и x/yn).
    '''
    index_x = 0

    while arg > data[index_x]:
        index_x += 1
        if arg < data[index_x]:
            index_x -= 1
            break

    index_x0 = index_x - math.ceil(power / 2) + 1
    index_xn = index_x + math.ceil(power / 2) + ((power - 1) % 2)

    if index_xn > len(data) - 1:
        index_x0 -= index_xn - len(data) + 1
        index_xn = len(data) - power
    elif index_x0 < 0:
        index_xn += -index_x0
        index_x0 = 0

    return index_x0, index_xn

def div_diff(z, node):
    '''
        Расчет разделенных разниц для полинома
Ньютона
    '''
    for i in range(node):
        pol = []
        for j in range(node - i):
            buf = (z[i + 1][j] - z[i + 1][j + 1]) / (z[0][j] - z[0][j + i + 1])
            pol.append(buf)
        z.append(pol)

    return z

def polinom_n(z, node, arg):
    '''
        Расчет значение функции от заданного
аргумента.
        Полином Ньютона.
```

```
    '''
    pol = div_diff(z, node)
    y = 0
    buf = 1
    for i in range(node + 1):
        y += buf * pol[i + 1][0]
        buf *= (arg - pol[0][i])

    return y

def multid_interp(z, x, y, power_x, power_y, arg_x, arg_y):
    '''
        А л г о р и т м  м н о г о м е р н о й  и н т е р п о л я ц и и.
    '''
    index_x0, index_xn = find_x0_xn(x, power_x + 1, arg_x)
    index_y0, index_yn = find_x0_xn(y, power_y + 1, arg_y)

    x = x[index_x0:index_xn]
    y = y[index_y0:index_yn]
    z = z[index_y0:index_yn]

    for i in range(power_y + 1):
        z[i] = z[i][index_x0:index_xn]

    x1 = [polinom_n([x, z[i]], power_x, arg_x) for i in range(power_y + 1)]
    y1 = polinom_n([y, x1], power_y, arg_y)

    return y1

def input_xy():
    '''
        В в о д  а р г у м е н т а. (в  с л у ч а е  о ш и б к и  д а е т с я  е щ е
п о п ы т к а)
    '''
    print("Enter X: ")
    flag = 0
    x = 0
    while flag == 0:
        x = input(float)
        try:
            val = float(x)
            flag = 1
        except ValueError:
            print("Some error! Try again")
    print("Enter Y: ")
    flag = 0
    y = 0
    while flag == 0:
        y = input(float)
        try:
            val = float(y)
            flag = 1
        except ValueError:
```

```
            print("Some error! Try again")
    return float(x), float(y)

def main():
    z, x, y = parse_table("points.xlsx")
    arg_x, arg_y = input_xy()
    arr_n = [1, 2, 3]
    printf_matrix(z)

    print("\n|  nx   |  ny   |    x    |    y    | Found Y | F(x, y) | Error   |")
    for n in arr_n:
        found_y = multid_interp(z, x, y, n, n, arg_x, arg_y)

        print("|   %d   |   %d   |  %.2f |  %.2f | %.5f | %.5f | %.4f |" \
            % (n, n, arg_x, arg_y, found_y, F(arg_x, arg_y),
            abs(found_y - F(arg_x, arg_y))))

if __name__ == "__main__":
    main()
```

Результаты работы

Результат интерполяции z(x,y) при степенях полиномов 1,2,3 для x=1.5, y=1.5 .

```
|  nx   |  ny   |    x    |    y    | Found Y | F(x, y) | Error   |
|   1   |   1   |  1.50 |  1.50 | 5.00000 | 4.50000 | 0.5000 |
|   2   |   2   |  1.50 |  1.50 | 4.50000 | 4.50000 | 0.0000 |
|   3   |   3   |  1.50 |  1.50 | 4.50000 | 4.50000 | 0.0000 |
```

nx	ny	x	y	Found Y	F(x, y)	Error
1	1	1.5	1.5	5.0	4.5	0.5
2	2	1.5	1.5	4.5	4.5	0.0
3	3	1.5	1.5	4.5	4.5	0.0

Контрольные вопросы

1. Пусть производящая функция таблицы суть z(x,y)=x^2 +y^2 . Область определения по x и y 0-5 и 0-5. Шаги по переменным равны 1. Степени nx = ny = 1, x = y = 1.5. Приведите по шагам те значения функции, которые получаются в ходе последовательных интерполяций по строкам и столбцу.

```
Enter X:
<class 'float'>1.5
Enter Y:
<class 'float'>1.5


 y/x   0   1   2   3   4   5
   0   0   1   4   9  16  25
   1   1   2   5  10  17  26
   2   4   5   8  13  20  29
   3   9  10  13  18  25  34
   4  16  17  20  25  32  41
   5  25  26  29  34  41  50

|  nx   |  ny   |    x    |    y    | Found Y | F(x, y) | Error  |
|   1   |   1   |  1.50   |  1.50   | 5.00000 | 4.50000 | 0.5000 |
```

Выполняя первый шаг из алгоритма описанного выше, то есть находим z(x, yk), где k от 0 до 2, то получим значения [3.5, 6.5] - соответственно для первой строки и второй. Далее проводим еще раз операцию интерполяции теперь по столбцам и уже получаем значение 5.0, (при подстановки в функцию z = x^2 + y^2 получим значение 4.5, то есть погрешность 0.5)

2. Какова минимальная степень двумерного полинома, построенного на четырех узлах? На шести узлах?

Минимальная степень двумерного полинома, построенного на 4 узлах, будет третья, всего степени могут быть от 0 до 3. На 6 узлах, будет пятая, всего степени могут быть от 0 до 5.

Так как количество необходимых узлов, для нахождения полинома n степени равно n + 1.

3. Предложите алгоритм двумерной интерполяции при хаотичном расположении узлов, т.е. когда таблицы функции на регулярной сетке нет, и метод последовательной интерполяции не работает. Какие имеются ограничения на расположение узлов при разных степенях полинома?

При хаотичном расположении узлов, ограничиваясь, интерполяционным полиномом первой степени, получаем $z = a + bx + cy$, и все его коэффициенты находят по трем узлам, которые мы выбираем в окрестности точки

интерполяция $z_i = a + bx_i + cy_i,\ 0 \le i \le 2$, здесь i - номер узла.

Ограничения: пример: при интерполяции полиномом 1 степени P(x, y) узлы не могут лежать на одной прямой в плоскости, а при интерполяции второй степени не должны лежать на одной плоскости в пространстве.

4. Пусть на каком-либо языке программирования написана функция, выполняющая интерполяцию по двум переменным. Опишите алгоритм использования этой функции для интерполяции по трем переменным.

1. Для начала необходимо выбрать нужные нам три переменные и их отрезки.
2. Пусть есть функция F(x, y, z). Делаем двумерную интерполяцию для переменных (x, y) nz раз, где nz - это степень переменной z. Полученные значения необходимо записать в какой-нибудь массив (например g).
3. Далее делаем еще раз двумерную интерполяцию для переменых (g, z).

5. Можно ли при последовательной интерполяции по разным направлениям использовать полиномы несовпадающих степеней или даже разные методы одномерной интерполяции, например, полином Ньютона и сплайн?

Можно. Так как результат данной интерполяции не зависит от порядка шагов: можно сначала интерполировать вдоль оси абсцисс, а затем вдоль оси ординат, а так же и наоборот, результат будет одним и тем же. Также не зависит от метода интерполяции.

6. Опишите алгоритм двумерной интерполяции на треугольной конфигурации узлов.

Алгоритм двумерной интерполяции на треугольной конфигурации узлов будет отличатся от выше описанного и использованного алгоритма лишь дополнительной проверкой на нужное количество узлов при одномерной интерполяции по оси ординат или оси абсцисс, или же если интерполяция не одномерная, то использование сразу всех узлов.