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 (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

ОТЧЕТ

по лабораторной работе № 5

Название: Построение и программная реализация алгоритмов численного интегрирования

Дисциплина: Вычислительные алгоритмы

Студент	ИУ7И - 46Б (Группа)	(Подпись, дата)	Андрич К. (И.О. Фамилия)
Преподаватель		(Подпись, дата)	В.М. Градов (И.О. Фамилия)

Москва, 2021

Цель работы

Получение навыков построения алгоритма вычисления двукратного интеграла с использованием квадратурных формул Гаусса и Симпсона

Задание

Построить алгоритм и программу для вычисления двукратного интеграла при фиксированном значении параметра.

$$\varepsilon(\tau) = \frac{4}{\pi} \int_0^{\pi/2} d\varphi \int_0^{\pi/2} [1 - \exp(-\tau \frac{l}{R})] \cos\theta \ \sin\theta \ d\theta ,$$

где $\frac{l}{R} = \frac{2\cos\theta}{1 - \sin^2\theta\cos^2\varphi}$,

θ, φ - углы сферических координат.

Применить метод последовательного интегрирования. По одному направлению использовать формулу Гаусса, а по другому - формулу Симпсона.

Код программы

В программе есть 2 файлов: main.py и functions.py

main.py

```
from functions import *
import math

def main():
    t, rc = get_t()
    if rc == 1:
        return
    else:
        rc = 1
        while not rc == 2:
            n1, m1 = get_method()
            n2, m2 = get_method()
            t_solve = lambda t: get_res(t_func(t), 0, math.pi / 2, 0, math.pi / 2,
m1, m2, n1, n2)
            result = t_solve(t)
            print("Result while τ = %.2f: %.6f" %(t, result))
            label = make_label(m1, m2, n1, n2)
            prepare_graph(t_solve, label)
            rc = menu()
```

```
        show_graph()

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

functions.py

```
from numpy.linalg.linalg import _multi_dot_matrix_chain_order
from numpy.polynomial.legendre import leggauss
from numpy import arange
import matplotlib.pyplot as plt
import math

def menu():
    print("Do you want to continue?")
    print("1 - yes")
    print("2 - no (graph will be shown)")
    try:
        rc = float(input("Choice: "))
    except:
        print("Input error")
    return rc

def get_t():
    t, rc = 0, 0
    try:
        t = float(input("Input τ: "))
    except:
        print("Input error")
        rc = 1
    return t, rc

def get_method():
    print("Choose integral method: ")
    print(" 1 - Gauss")
    print(" 2 - Simpson")
    n = 0
    try:
        n = int(input("Choice: "))
    except:
        print("Input error")
    m = 0
    if n == 1:
        min_value = 0
    else:
        min_value = 2
    while (m <= min_value):
        try:
            m = int(input("Number of nodes: "))
        except:
            print("Input error")
        if min_value == 2 and m <= 2:
```

```
            print("For Simpson's method number of nodes must be greater than 2")
    return n, m

def Gauss(function, a, b, n):
    args, coeffs = leggauss(n)
    result = 0
    for i in range(n):
        result += (b - a) / 2 * coeffs[i] * function((b + a) / 2 + (b - a) * args[i]
/ 2)
    return result

def Simpson(function, a, b, n):
    h = (b - a) / (n - 1)
    x = a
    result = 0
    for i in range((n - 1) // 2):
        result += function(x) + 4 * function(x + h) + function(x + 2 * h)
        x += 2 * h
    result *= (h / 3)
    return result

def t_func(t):
    coeff = lambda x, y: 2 * math.cos(x) / (1 - (math.sin(x) ** 2) * (math.cos(y) **
2))
    func = lambda x, y: (4 / math.pi) * (1 - math.exp(-t * coeff(x, y))) *
math.cos(x) * math.sin(x)
    return func

def get_res(function, lim1, lim2, lim3, lim4, m1, m2, n1, n2):
    if n1 == 1:
        method1 = Gauss
    else:
        method1 = Simpson
    if n2 == 1:
        method2 = Gauss
    else:
        method2 = Simpson
    internal = lambda x: method1(lambda y: function(x, y), lim3, lim4, m2)
    result =  method2(internal, lim1, lim2, m1)
    return result

def make_label(m1, m2, n1, n2):
    label = "m1 = " + str(m1) + ", m2 = " + str(m2) + ";  "
    if n1 == 1:
        label += "Gauss "
    else:
        label += "Simpson "
    if n2 == 1:
        label += "- Gauss"
    else:
        label += "- Simpson"
    return label
```

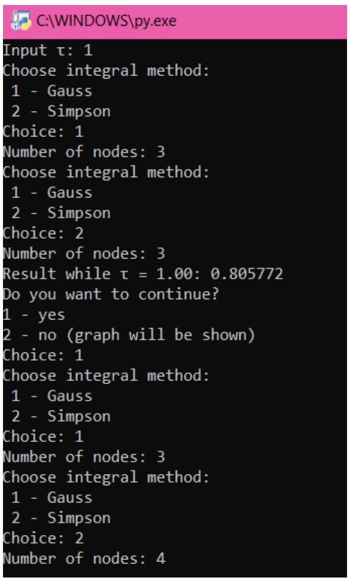
```python
def prepare_graph(function, label):
    X, Y = [], []
    for t in arange(0.05, 10 + 0.05, 0.05):
        X.append(t)
        Y.append(function(t))
    plt.plot(X, Y, label = label)

def show_graph():
    plt.legend()
    plt.ylabel("Result")
    plt.xlabel("τ value")
    plt.show()
```

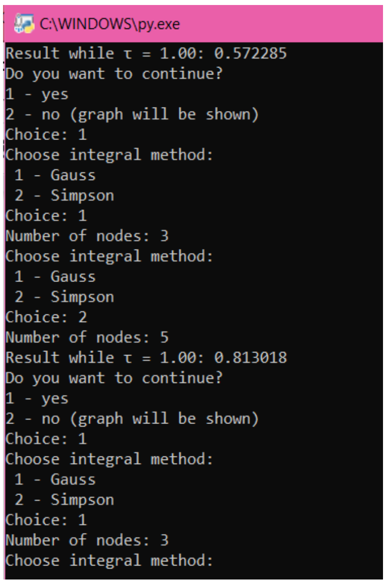
Результаты работы

1. Описать алгоритм вычисления n корней полинома Лежандра n-ой степени P (x) n при реализации формулы Гаусса.
   Из-за того что на отрезке [-1, 1] у полинома Лежандра n-ой степени n различней корней и из каждого корня x следует -x можем искать их только на (0, 1]. Для этого пользуем метод половинного деления. Интервал разбиваем на несколько меньших и проверяем каждый. Если у отрезка разные знаки на концах, это значит что корень находится на этом отрезке, а если один из них равен 0 это значит что корень найден.

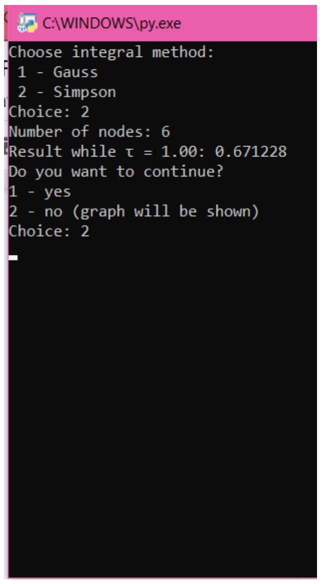
2. Исследовать влияние количества выбираемых узлов сетки по каждому направлению на точность расчетов.



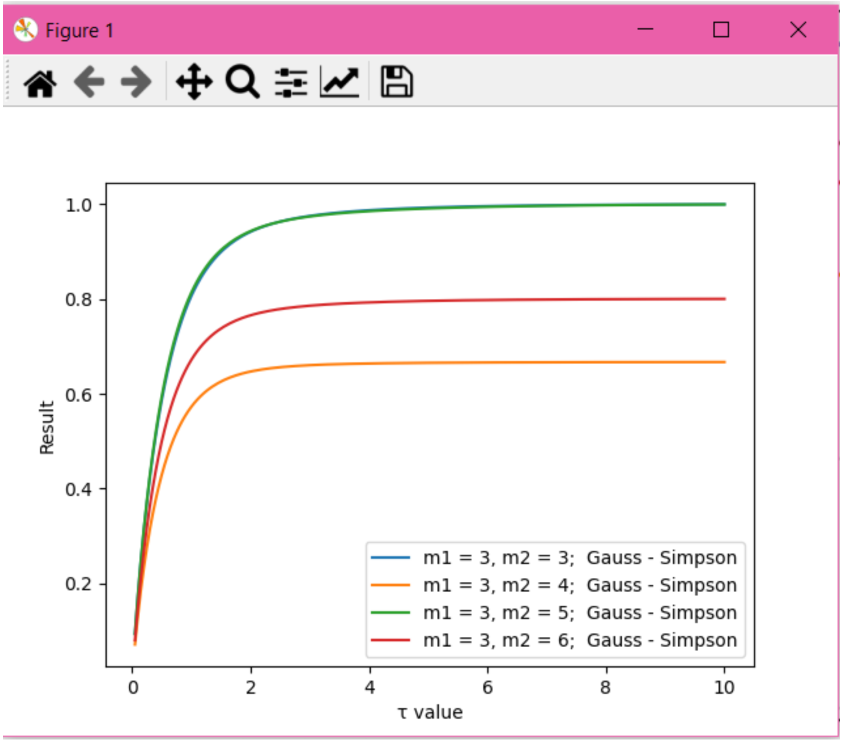
```
C:\WINDOWS\py.exe
Input τ: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Result while τ = 1.00: 0.805772
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 4
```



```
C:\WINDOWS\py.exe
Result while τ = 1.00: 0.572285
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 5
Result while τ = 1.00: 0.813018
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Choose integral method:
```



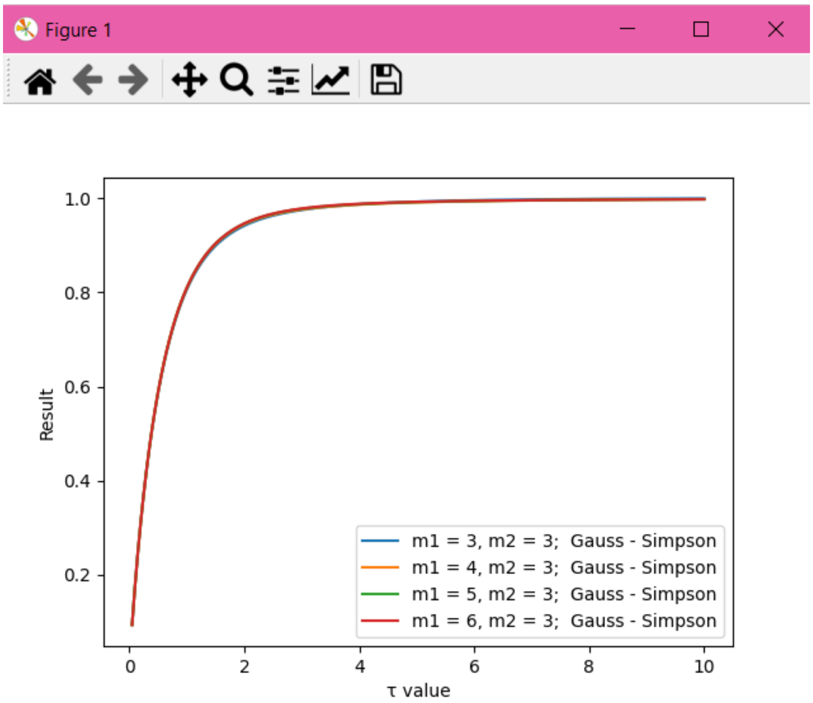
```
C:\WINDOWS\py.exe
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 6
Result while τ = 1.00: 0.671228
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 2
```



Здесь видим что при нечетных m результаты сильно отличаются от четных, получаем расхождение.

```
C:\WINDOWS\py.exe
Input τ: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Result while τ = 1.00: 0.805772
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 4
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Result while τ = 1.00: 0.809147
Do you want to continue?
1 - yes
2 - no (graph will be shown)
```

```
C:\WINDOWS\py.exe
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 5
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Result while τ = 1.00: 0.809460
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 6
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Result while τ = 1.00: 0.809439
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 2
```



Здесь видно что у методы Гаусса высокая сходимость независимо от четности.

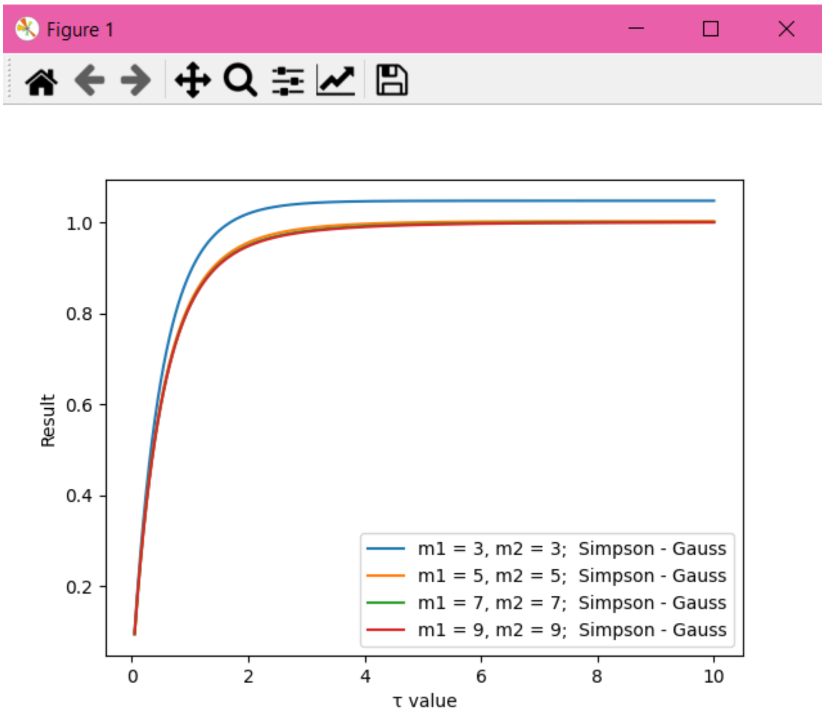
3. Построить график зависимости ε(τ) в диапазоне изменения τ =0.05-10. Указать при каком количестве узлов получены результаты.

C:\WINDOWS\py.exe

```
Input τ: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 3
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 3
Result while τ = 1.00: 0.888695
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 5
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 5
Result while τ = 1.00: 0.822008
Do you want to continue?
1 - yes
2 - no (graph will be shown)
```

C:\WINDOWS\py.exe

```
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 7
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 7
Result while τ = 1.00: 0.816547
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 1
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 2
Number of nodes: 9
Choose integral method:
 1 - Gauss
 2 - Simpson
Choice: 1
Number of nodes: 9
Result while τ = 1.00: 0.815300
Do you want to continue?
1 - yes
2 - no (graph will be shown)
Choice: 2
```



Оптимальное значение при 5х5

Вопросы при защите лабораторной работы

1. *В каких ситуациях теоретический порядок квадратурных формул численного интегрирования не достигается.*
   Теоретический порядок квадратурных формул численного интегрирования не достигается если у подынтегральной функции нет соответствующих производных.

2. *Построить формулу Гаусса численного интегрирования при одном узле.*

$$\sum_{i=1}^{1} A_i = 2 \Rightarrow A_1 = 2$$

$$P_1(x) = x \Rightarrow x_1 = 0(1) \quad *$$

$$x = \frac{b+a}{2} + \frac{b-a}{2} t$$

$$\int_a^b f(x)\,dx = \frac{b-a}{2} \int_{-1}^{1} f\left(\frac{b+a}{2} + \frac{b-a}{2} t\right) dt$$

$$\int_{-1}^{1} f(t)\,dt = A_1 f(x_1) \quad **$$

$$* \text{ и } ** \Rightarrow \int_a^b f(x)\,dx = (b-a)\, f\left(\frac{b+a}{2}\right)$$

3. *Построить формулу Гаусса численного интегрирования при двух узлах.*

$$\begin{cases} A_0 = 1 \\ A_1 = 1 \end{cases} \qquad \begin{cases} t_0 = -\dfrac{1}{\sqrt{3}} \\ t_1 = \dfrac{1}{\sqrt{3}} \end{cases}$$

$$\int_a^b f(x)\,dx = \frac{b-a}{2}\left(\left(f\left(\frac{b+a}{2} + \frac{b-a}{2}\left(-\frac{1}{\sqrt{3}}\right)\right)\right) + f\left(\frac{b+a}{2} + \frac{b-a}{2}\cdot\frac{1}{\sqrt{3}}\right)\right)$$

4. *Получить обобщенную кубатурную формулу, аналогичную (6.6) из лекции №6, для вычисления двойного интеграла методом последовательного интегрирования на основе формулы трапеций с тремя узлами по каждому направлению.*

$$\int_c^d \int_a^b f(x,y)\,dx\,dy = \int_a^b F(x)\,dx \quad \text{где}$$

$$F(x) = \int_c^d f(x,y)\,dy$$

$$\int_c^d \int_a^b f(x,y)\,dx\,dy = h_x\left(\frac{1}{2}F(x_0) + F(x_1) + \frac{1}{2}F(x_2)\right)$$

$$h_x = \frac{b-a}{N}$$

$$\int_c^d \int_a^b f(x,y)\,dx\,dy = h_x h_y \frac{1}{2}\left(\frac{1}{2}f(x_0,y_0) + f(x_0,y_1) + \frac{1}{2}f(x_0,y_2)\right) +$$

$$+ \left(\frac{1}{2}f(x_1,y_0) + f(x_1,y_1) + \frac{1}{2}f(x_1,y_2)\right) + \frac{1}{2}\left(\frac{1}{2}f(x_2,y_0) + f(x_2,y_1) + \frac{1}{2}f(x_2,y_2)\right)$$

$$h_y = \frac{d-c}{M}$$