**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”

# ОТЧЁТ

## *К ЛАБОРАТОРНОЙ РАБОТЕ №1*

## *НА ТЕМУ:*

## *“Разработка ПО”*

Студент ИУ7-68Б(В) _____ Д.П. Косаревский
(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель _____ В.И. Солодовников
(Подпись, дата) (И.О.Фамилия)

*2021 г.*

**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)**

УТВЕРЖДАЮ
Заведующий кафедрой ИУ7
(Индекс)
_____ И.В.Рудаков
(И.О.Фамилия)
« _____ » _____ 2021 г.

# З А Д А Н И Е
## на выполнение лабораторной работы

по дисциплине “Основы программной инженерии”
Студент группы ИУ7-68Б(В)
Косаревский Дмитрий Петрович
(Фамилия, имя, отчество)
Тема лабораторной работы “Разработка ПО”

***Задание:***
Написать программу для приближенного вычисления определенного интеграла методами трапеций с заданным шагом, трапеций с заданной точностью, а также точки пересечения выбранной пользователем функции с осью абсцисс на заданном интервале методом дихотомии. Выбранную для проверки функцию передавать как отдельный параметр подпрограмм вычисления значений функции, интеграла, корня.
Для проверки использовать следующие функции:
1) $\sin(x)$;
2) $\cos^2(x) \cdot \ln^2(x + 5)$.

Дата выдачи задания « ___ » _____ 2021 г.

**Преподаватель** _____ В.И. Солодовников
(Подпись, дата) (И.О.Фамилия)

**Студент** _____ Д.П. Косаревский
(Подпись, дата) (И.О.Фамилия)

# Код основных функций программы

Основные функции программы:

Функция для приближенного вычисления определенного интеграла методами трапеций с заданным шагом:

```python
def trapezoidal_rule(func, low_limit: float, up_limit: float, intervals: float) -> float:
    """
        Правило трапеций для численной аппроксимации интегральной заданной функции
    :param func: математическая функция
    :param low_limit: нижний предел интегрирования
    :param up_limit: верхний предел интегрирования
    :param intervals: число отрезков, на которые разбивается
    :return: результат вычислений
    """
    sum_xi = 0.0
    h = (up_limit - low_limit) / intervals  # finding midpoint, (b-a)/n
    sum_ = func(low_limit) + func(up_limit)  # find the f(a) and f(b)
    for i in range(1, int(intervals)):
        sum_xi += func(low_limit + i * h)
    fx = (h / 2) * (sum_ + 2 * sum_xi)

    return round(fx, 5)
```

Функция для приближенного вычисления определенного интеграла методами трапеций с заданной точностью:

```python
def precision_trapezoidal_rule(func, low_lim: float, up_lim: float, max_err: float = .1, intervals: int = 1) -> float:
    """
        Правило трапеций с заданной точностью
    :param func: математическая функция
    :param low_lim: нижний предел интегрирования
    :param up_lim: верхний предел интегрирования
    :param max_err: заданная точность
    :param intervals: число отрезков, на которые разбивается
    :return: результат вычислений
    """
    dx = (up_lim - low_lim) / intervals
    total = 0

    # выполняем интеграцию
    x = low_lim
    for interval in range(intervals):
        # добавляем область трапеции для этого среза
        total += slice_area(func, x, x + dx, max_err)

        # переходим к следующему срезу
        x += dx

    return round(total, 5)
```

Вспомогательная функция для приближенного вычисления определенного интеграла методами трапеций с заданной точностью, вызываемая внутри функции precision_trapezoidal_rule:

```
def slice_area(function, x1, x2, max_error):
    # вычисляем функцию в конечных и средних точках
    y1 = function(x1)
    y2 = function(x2)
    xm = (x1 + x2) / 2
    ym = function(xm)

    # рассчитываем площади срезов и самого большого участка
    large = (x2 - x1) * (y1 + y2) / 2
    first = (xm - x1) * (y1 + ym) / 2
    second = (x2 - xm) * (y2 + ym) / 2
    both = first + second

    # рассчитываем ошибку
    error = (both - large) / large

    # сравниваем ошибку с допустимым значением ошибки (точности)
    if abs(error) < max_error:
        return both

    # если ошибка больше допустимого значения - делим ее на две части (два среза)
    return slice_area(function, x1, xm, max_error) + slice_area(function, xm, x2, max_error)
```

Функция для вычисления точки пересечения выбранной пользователем функции с осью абсцисс на заданном интервале методом дихотомии:

```
def dichotomy(f, a, b, tol):
    niter = 0
    inc = []
    y = (a + b) / 2

    if f(a) * f(b) < 0:
        while abs(b - a) > tol:
            x = (a + b) / 2
            inc.append(abs(x - y))
            y = x
            niter += 1
            if f(a) * f(x) <= 0:
                b = x
            else:
                a = x
        zero = (a + b) / 2
        # res = f(zero)
        # err = abs(a - b)
        st.write(f"Функция {f.__doc__} пересекает ось абсцисс в точке [{round(zero, 2)}, 0]")
        # st.write(f"Остальная часть функции в нулевой точке: f (zero) = {res}")
        # st.write(f"Количество итераций: niter = {niter}")
        # st.write(f"Вектор, содержащий остатки на каждой итерации: inc = {inc}")
        # st.write(f"Длина последнего интервала: err = {err}")
        return zero
    elif f(a) * f(b) > 0:
        st.error(f"Невозможно применить метод дихотомии для функции {f.__doc__} на интервале [{a}, {b}]")

    else:
        if f(a) == 0:
            st.write(f"Нуль функции {f.__doc__} на [{a}, {b}] над точкой [{round(a, 2)}, 0]")
            return a
        else:
            st.write(f"Нуль функции {f.__doc__} на [{a}, {b}] b = [{round(b, 2)}, 0]")
            return b
```

Математические функции, используемые для проверки:

```python
def equation_1(x):
    """sin(x)"""
    return np.sin(x)


def equation_2(x):
    """cos^2(x) * ln^2(x+5)"""
    return np.cos(x) ** 2 * np.log(x + 5) ** 2
```

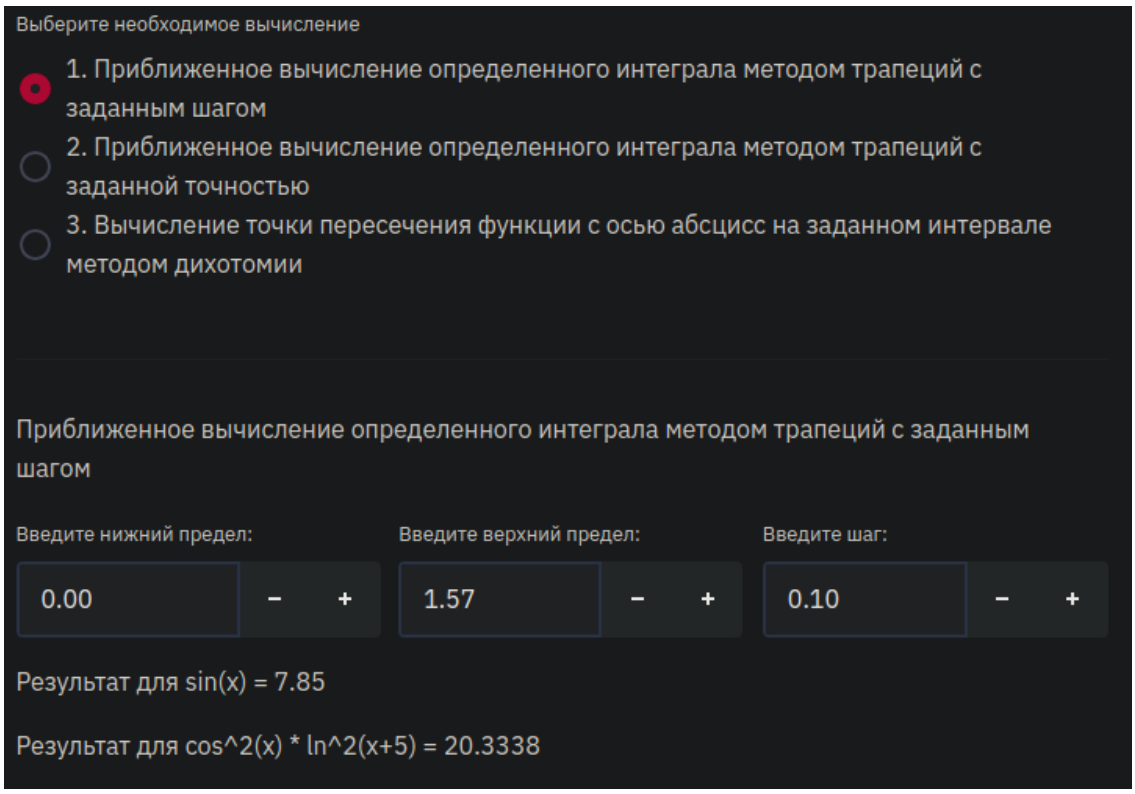
Функция для отрисовки графиков:

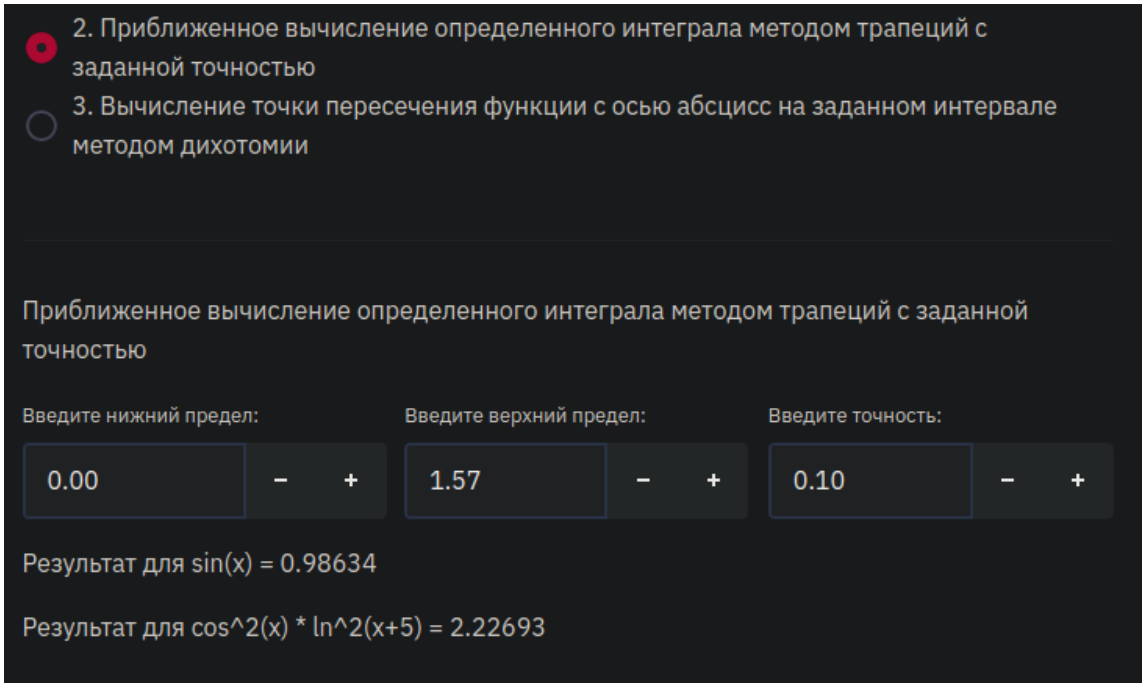
```python
def plot(func, a, b, zero):
    """ отрисовка графика """
    x = np.arange(a, b, 0.001)
    y = [func(i) for i in x]
    fig = plt.figure()
    ax = fig.add_subplot(1, 1, 1)
    ax.plot(x, y)
    if zero:
        ax.scatter(zero, 0)
    ax.axhline(0, color='black')
    ax.set_title(f"Функция {func.__doc__}")
    ax.set_xlabel("$x$")
    ax.set_ylabel("$f(x)$")
    ax.grid(True)

    st.write(fig)
```

# Результаты тестирования

В результате запуска и тестирования программы были получены следующие результаты:



3. Вычисление точки пересечения функции с осью абсцисс на заданном интервале методом дихотомии

Вычисление точки пересечения функции с осью абсцисс на заданном интервале методом дихотомии

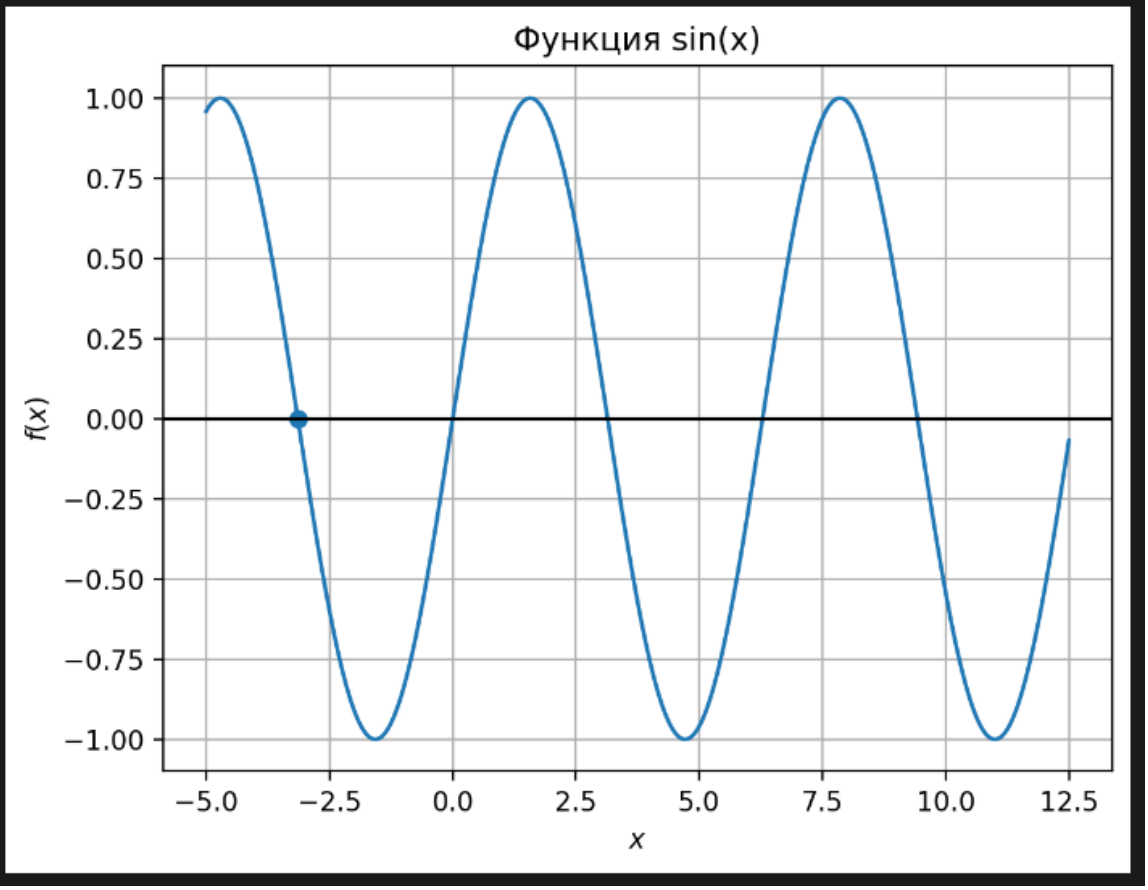
Начало интервала (a): -5.00 − +

Конец интервала (b): 12.50 − +

Эпсилон (e): 0.00 − +

Значение Эпсилон = 1e-05

Функция sin(x) пересекает ось абсцисс в точке [-3.14, 0]



# Результат

В результате работы была достигнута поставленная цель:

1. Создана программа в соответствии с описанным заданием
2. Реализованы 3 варианта вычислений
3. Реализована возможность параметризации вычислений пользователем
4. Реализовано отображение графиков и их изменений в зависимости от заданных параметров

Код программы находится в открытом репозитории по ссылке: https://github.com/dKosarevsky/SEF_lab_001/blob/main/integral_trapezoidal.py

Работающую программу можно увидеть и протестировать по ссылке: https://share.streamlit.io/dkosarevsky/sef_lab_001/main/integral_trapezoidal.py

Программа написана на языке программирования Python 3.8.8 с использованием следующих библиотек:

- streamlit
- numpy
- matplotlib