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 № 3

Тема: Программно-алгоритмическая реализация моделей на основе ОДУ второго порядка с краевыми условиями II и III рода

Студент Лучина Е.Д

Группа ИУ7-61Б

Преподаватель Градов В.М.

Москва.
2020 г.

Содержание

Цель работы.

Получение навыков разработки алгоритмов решения краевой задачи при реализации моделей, построенных на ОДУ второго порядка.

Физическое содержание задачи

Сформулирована математическая модель, которая описывает температурное поле $T(x)$ вдоль цилиндрического стержня радиуса R и длиной l, причем $R \ll l$ и температуру можно принять постоянной по радиусу цилиндра. Ось x направлена вдоль оси цилиндра и начало координат совпадает с левым торцом стержня. Слева при $x = 0$ цилиндр нагружается тепловым потоком F_0. Стержень обдувается воздухом, температура которого равна T_0. В результате происходит съем тепла с цилиндрической поверхности и поверхности правого торца при $x = l$. Функции $k(x)$, $\alpha(x)$ являются, соответственно, коэффициентами теплопроводности материала стержня и теплоотдачи при обдуве.

Исходные данные.

Уравнение для функции $T(x)$

$$\frac{d}{dx}\left(k(x)\frac{dT}{dx}\right) - \frac{2}{R}\alpha(x)T + \frac{2T_0}{R}\alpha(x) = 0 \qquad (1)$$

Отметим $F(x) = -k(x)\frac{dT}{dx}$, $p(x) = \frac{2\alpha(x)}{R}$, $f(x) = \frac{2T_0}{R}\alpha(x)$ (2)

$$p_i = p(x_i);\ f_i = f(x_i)$$

Краевые условия имеют вид

$$\begin{cases} x = 0, & -k(0)\dfrac{dT}{dx} = F_0, \\ x = l, & -k(l)\dfrac{dT}{dx} = \alpha_N\,(T(l) - T_0) \end{cases} \qquad (3)$$

Функции $k(x)$, $\alpha(x)$ заданы своими константами

$$k(x) = \frac{a}{x-b};\ \alpha(x) = \frac{c}{x-d} \qquad (4)$$

$$k_i = k(x_i);\ \alpha_i = \alpha(x_i)$$

Константы α, b следует найти из условий $k(0) = k_0$, $k(l) = k_N$

а константы c, d из условий $\alpha(0) = \alpha_0$, $\alpha(l) = \alpha_N$.

Величины $k_0,\ k_N,\ \alpha_0,\ \alpha_N$ задает пользователь, их надо вынести в интерфейс.

$$a = -k_0 b;\ b = \frac{k_N \times l}{k_N - k_0};\ c = -\alpha_0 d;\ d = \frac{\alpha_N \times l}{\alpha_N - \alpha_0}; \qquad (5)$$

Разностные схемы и аналоги при конкретных x

В Лекции №7 получено:

Разностная схема

$$A_n T_{n+1} - B_n T_n + C_n T_{n-1} + D_n = 0,\ 1 \le n \le N-1 \qquad (6)$$

где

$$A_n = \frac{\chi_{n+1/2}}{h},\ C_n = \frac{\chi_{n-1/2}}{h},\ B_n = A_n + C_n + p_n h,\ D_n = f_n h \qquad (7)$$

$$\chi_{n+1/2} = \frac{2k_n k_{n+1}}{k_n + k_{n+1}},\ \chi_{n-1/2} = \frac{2k_n k_{n-1}}{k_n + k_{n-1}},\ h - \text{ шаг сетки}$$

Разностный аналог краевого условия при $x = 0$

$$T_0 = \frac{(\chi_{\frac12} - \frac{h^2}{8}p_{\frac12})}{(\chi_{\frac12} + \frac{h^2}{8}p_{\frac12} + \frac{h^2}{4}p_0)} T_1 + \frac{hF_0 + \frac{h^2}{4}(f_{\frac12} + f_0)}{(\chi_{\frac12} + \frac{h^2}{8}p_{\frac12} + \frac{h^2}{4}p_0)}$$

$$(\chi_{\frac12} + \frac{h^2}{8}p_{\frac12} + \frac{h^2}{4}p_0)T_0 + (\frac{h^2}{8}p_{\frac12} - \chi_{\frac12})T_1 = hF_0 + \frac{h^2}{4}(f_{\frac12} + f_0) \qquad (8)$$

Можно принять простую аппроксимацию: $p_{\frac12} = \frac{p_0 + p_1}{2},\ f_{\frac12} = \frac{f_0 + f_1}{2}$.

Разностный аналог краевого условия при $x = l$

Интегро-интерполяционным методом самостоятельно (аналогично решению в лекции) будет получен разностный аналог краевого условия при $x = l$.

Для этого надо проинтегрировать на отрезке $[x_{N-1/2},\ x_N]$ выписанное выше уравнение (1) с учетом

$$F_N = \alpha_N(T_N - T_0),\ F_{N-1/2} = \chi_{N-1/2}\frac{T_{N-1} - T_N}{h} \qquad (9)$$

$$-\int_{x_{N-1/2}}^{x_N} \frac{dF}{dx}dx - \int_{x_{N-1/2}}^{x_N} p(x)T\,dx + \int_{x_{N-1/2}}^{x_N} f(x)dx = 0 \qquad (10)$$

Выполняя интегрирование в первом слагаемом и применяя метод трапеций для остальных интегралов, получим

$$-(F_N - F_{N-1/2}) - (p_N T_N + p_{N-1/2} T_{N-1/2})\frac{h}{4} + (f_N + f_{N-1/2})\frac{h}{4} = 0$$

Подставим F_N, $F_{N-1/2}$, и примем простую аппроксимацию для $p_{N-1/2}$, $f_{N-1/2}$, $T_{N-1/2}$ как среднее значений функции при x_{N-1} и x_N.

$$\chi_{N-1/2}\frac{T_{N-1}-T_N}{h} - \alpha_N(T_N - T_0) - (p_N T_N + \frac{p_N+p_{N-1}}{2}\frac{T_N+T_{N-1}}{2})\frac{h}{4} + (f_N + \frac{f_N+f_{N-1}}{2})\frac{h}{4} = 0$$

приведем подобные слагаемые

$$(-\chi_{N-1/2} - \alpha_N h - \frac{p_N h^2}{4} - \frac{p_N+p_{N-1}}{16}h^2)T_N + (\chi_{N-1/2} - \frac{p_N+p_{N-1}}{16}h^2)T_{N-1} = -\alpha_N h T_0 - (f_N + \frac{f_N+f_{N-1}}{2})\frac{h^2}{4} \quad (11)$$

Метод прогонки

С учетом полученных данных для решения задачи (то есть системы (6)) может быть применен метод прогонки. Прогоночные коэффициенты можно найти по следующим рекуррентным формулам:

$$\xi_{n+1} = \frac{C_n}{B_n - A_n\xi_n}\ ;\ \eta_{n+1} = \frac{D_n + A_n\eta_n}{B_n - A_n\xi_n} \qquad (12)$$

Формулы для краевых значений

$$K_0 T_0 + M_0 T_1 = P_0,\ K_N T_N + M_N T_{N-1} = P_N \qquad (13)$$

Сравним с формулами (8) и (11) получим следующее (14):

$$K_0 = \chi_{1/2} + \frac{h^2}{8}p_{1/2} + \frac{h^2}{4}p_0$$
$$M_0 = \frac{h^2}{8}p_{1/2} - \chi_{1/2}$$
$$P_0 = hF_0 + \frac{h^2}{4}(f_{1/2} - f_0)$$
$$K_N = -\chi_{N-1/2} - \alpha_N h - \frac{p_N h^2}{4} - \frac{p_N+p_{N-1}}{16}h^2$$
$$M_N = \chi_{N-1/2} - \frac{p_N+p_{N-1}}{16}h^2$$
$$P_N = -\alpha_N h T_0 - (f_N + \frac{f_N+f_{N-1}}{2})\frac{h^2}{4}$$

Начальные прогоночные коэффициенты

$$\xi_1 = -\frac{M_0}{K_0},\ \eta_1 = \frac{P_0}{K_0} \qquad (15)$$

Алгоритм прогонки

- Найти K_0, M_0, P_0, K_N, M_N, P_N - формулы (14)
- Найти A_n, B_n, C_n, D_n - формулы (7)
- Найти прогоночные коэффициенты (прямой ход) - формулы (12), (16)
- Найти значения T (обратный ход) по формулам

$$T_N = \frac{P_N - M_N \eta_N}{K_N + M_N \xi_N} \quad (16)$$

$$T_n = \xi_{n+1} T_{n+1} + \eta_{n+1} \quad (17)$$

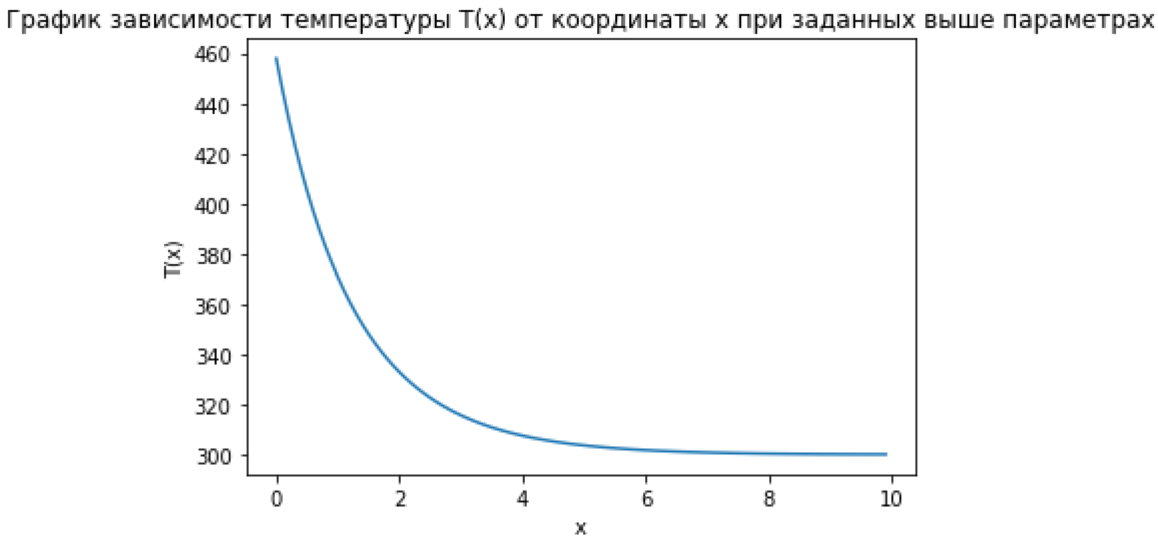
Значения параметров для отладки

все размерности согласованы

k_0 = 0.4 Вт/см К k_N = 0.1 Вт/см К	α_0 = 0.05 Вт/см2 К α_N = 0.01 Вт/см^2 К,	l = 10 см R = 0.5 см	$T_0 = 300K$ F_0 = 50 Вт/см2

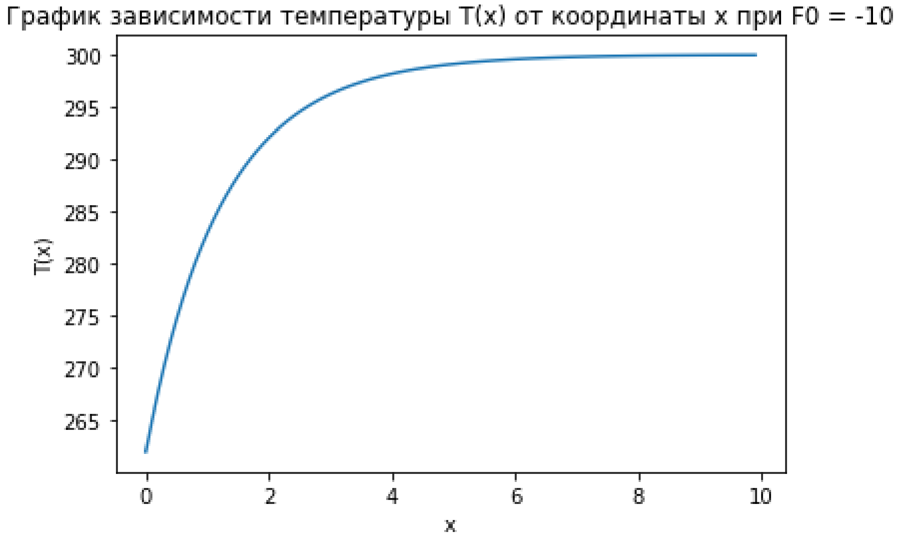
Результаты работы

1. Представить разностный аналог краевого условия при $x = l$ и его краткий вывод интегро-интерполяционным методом - пункт “Разностный аналог краевого условия при $x = l$”
2. График зависимости температуры $T(x)$ от координаты x при заданных выше параметрах.



3. График зависимости $T(x)$ при $F_0 = -10$ Вт/см2.

Справка. При отрицательном тепловом потоке слева идет съем тепла, поэтому производная $T'(x)$ должна быть положительной.



Действительно функция возрастающая, производная положительна

4. График зависимости $T(x)$ при увеличенных значениях $\alpha(x)$ (например, в 3 раза). Сравнить с п.2.

Справка. При увеличении теплосъема и неизменном потоке F_0 уровень температур $T(x)$ должен снижаться, а градиент увеличиваться.

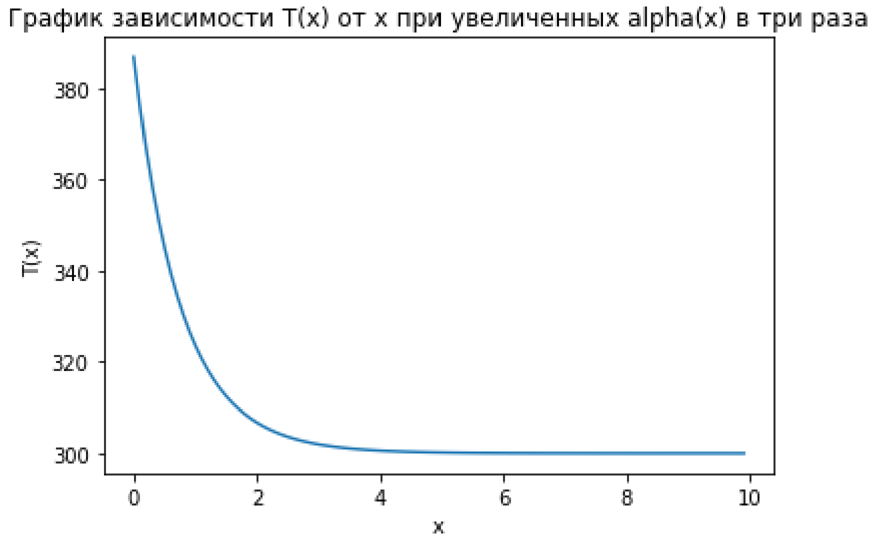
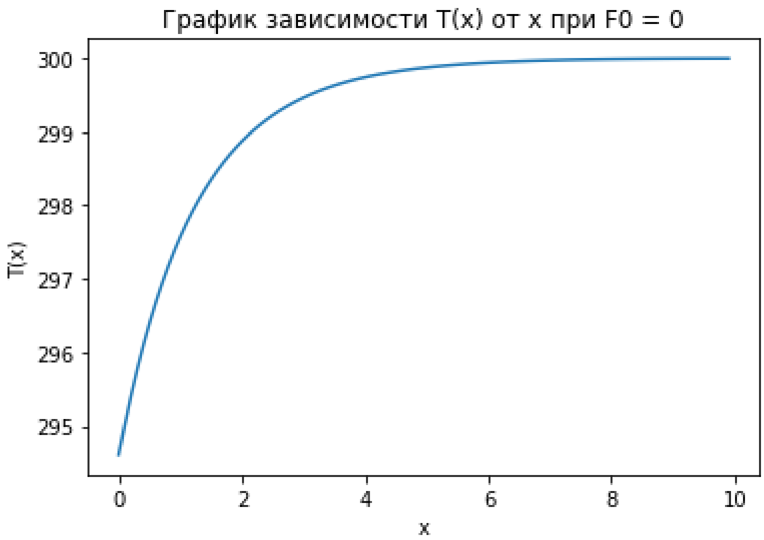


График более приближен к началу координат. Максимальное значение температуры достигает 390, и уже при x=2 температура становится приблизительно 300K. При исходных значениях температура изменяется от 460 до нуля и спад распределен на все 4 сантиметра.

5. График зависимости $T(x)$ при $F_0 = 0$.

Справка. В данных условиях тепловое нагружение отсутствует, причин для нагрева нет, температура стержня должна быть равна температуре окружающей среды T_0

(разумеется с некоторой погрешностью, определяемой приближенным характером вычислений).



Температура равна 300 на протяжении всего стержня с погрешностью в примерно 5K.

Листинг программы

Программа написана на языке Python. Графики построены с помощью модуля matplotlib.

```
# k0 = 0.4
# kN = 0.1
# alpha0 = 0.05
# alphaN = 0.01
k0 = float(input('k0: '))
kN = float(input('kN: '))
alpha0 = float(input('alpha0: '))
alphaN = float(input('alphaN: '))

l = 10
R = 0.5
T0 = 300
F0 = 50
# F0 = -10
# F0 = 0

h = 0.1
N = (int)(l / h)
```

```
# k(x), aplha(x), соответсвенно a, b, c, d
b = (kN * l) / (kN - k0)
a = -k0 * b
d = (alphaN * l) / (alphaN - alpha0)
c = -alpha0 * d

def k(x): return (a / (x - b))
# def k(x): return (5 * a / (x - b))
```

```
def alpha(x): return (c / (x - d))
# def alpha(x): return (3 * c / (x - d))
```

```
# Определим формулы для p, f
def p(x): return 2 * alpha(x) / R
def f(x): return 2 * T0 * alpha(x) / R

# Введем Хи
def Hi(x1, x2):
    k1 = k(x1)
    k2 = k(x2)
    return 2 * k1 * k2 / (k1 + k2)
```

```
# Функция высчитывающая K0, M0, P0. Левую границу.
def left_border():
    Hi12 = Hi(0, 1)
    p0 = p(0)
    p12 = (p0 + p(1)) / 2
    f0 = f(0)
    f12 = (f0 + f(1)) / 2
    h24 = h * h / 4
    h28p12 = h24 / 2 * p12

    K0 = Hi12 + h28p12 + h * h24 * p0
    M0 = h28p12 - Hi12
    P0 = h * F0 + h24 * (f12 - f0)
    return (K0, M0, P0)
```

```
# Функция высчитывающая KN, MN, PN. Праву границу
def right_border():
    HiN12 = Hi(l, l - h)
    pN = p(l)
    pN1 = p(l - h)
    pNpN116h2 = (pN + pN1) * h * h / 16
    fN = f(l)
    fN1 = f(l - h)

    KN = - HiN12 - alphaN * h - pN * h * h / 4 - pNpN116h2
    MN = HiN12 - pNpN116h2
    PN = - alphaN * h * T0 - (fN + (fN + fN1) / 2) * h * h / 4
    return (KN, MN, PN)
```

```
# получим An, Bn, Cn, Dn
def A(x): return Hi(x, x + h) / h
def C(x): return Hi(x, x - h) / h
def B(x): return A(x) + C(x) + p(x) * h
def D(x): return f(x) * h

# заполним матрицу коэффициентами
def coefs():
    A_arr = [0 for x in range(N)]
    B_arr = [0 for x in range(N)]
    C_arr = [0 for x in range(N)]
    D_arr = [0 for x in range(N)]
    for i in range (N):
        x = i * h
        A_arr[i] = A(x)
        B_arr[i] = B(x)
        C_arr[i] = C(x)
```

```
        D_arr[i] = D(x)
    return (A_arr, B_arr, C_arr, D_arr)
```

```
# вычислим коэффициенты прогонки
def ksi_etta(K0, M0, P0, A_arr, B_arr, C_arr, D_arr):
    ksi_arr = [None] * N
    etta_arr = [None] * N
    ksi_arr[0] = -M0 / K0
    etta_arr[0] = P0 / K0
    for i in range(1, N):
        denominator = B_arr[i - 1] - A_arr[i - 1] * ksi_arr[i - 1]
        ksi_arr[i] = C_arr[i - 1] / denominator
        etta_arr[i] = (D_arr[i - 1] + A_arr[i - 1] * etta_arr[i - 1]) /
denominator
    return (ksi_arr, etta_arr)
```

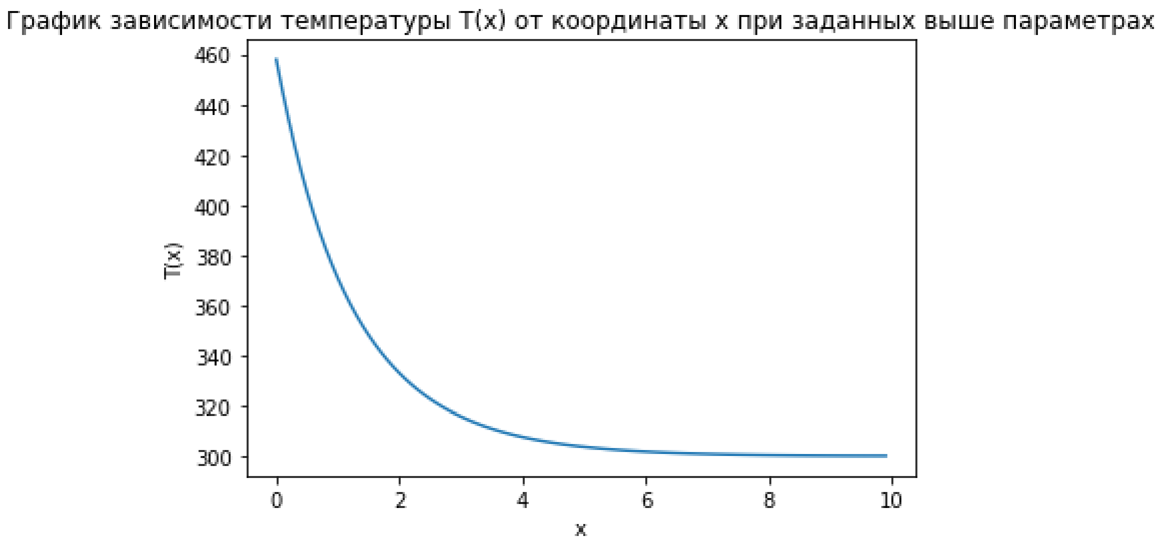
```
# вычислим значения T
def get_T(KN, MN, PN, ksi_arr, etta_arr):
    T_arr = [None] * N
    T_arr[N - 1] = (PN - MN * etta_arr[N - 1]) / (KN + MN * ksi_arr[N - 1])
    for i in range(N - 2, -1, -1):
        T_arr[i] = ksi_arr[i + 1] * T_arr[i + 1] + etta_arr[i + 1]
    return T_arr
```

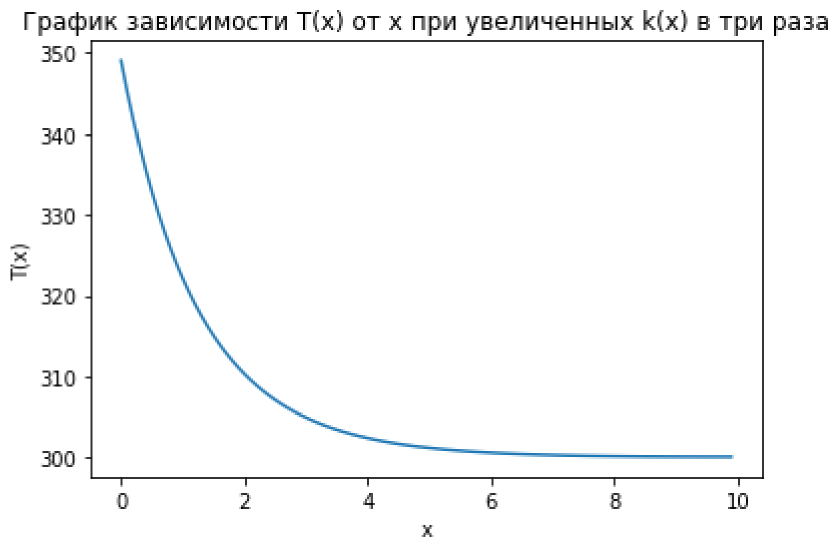
```
# соберем все вместе
from matplotlib import pyplot as plt
def solve():
    K0, M0, P0 = left_border()
    KN, MN, PN = right_border()
    A_arr, B_arr, C_arr, D_arr = coefs()
    ksi_arr, etta_arr = ksi_etta(K0, M0, P0, A_arr, B_arr, C_arr, D_arr)
    result_T = get_T(KN, MN, PN, ksi_arr, etta_arr)
    X = [0 for i in range (N)]
    for i in range (N):
        X[i] = i * h
    plt.plot(X, result_T)
    plt.ylabel('T(x)')
    plt.xlabel('x')
    plt.title('График зависимости T(x) от x')

solve()
```

Вопросы при защите лабораторной работы.

1. Какие способы тестирования программы можно предложить?

Если коэффициент теплопроводности стержня больше, температура будет падать медленнее. График будет более пологий. Увеличим k(x) в 3 раза.





<u>2. Получите простейший разностный аналог нелинейного краевого условия</u>

при $x = l, \quad = -k(l)\frac{dT}{dx} = \alpha_N(T(l) - T_0) + \varphi(T)$, где $\varphi(T)$ - заданная функция.

$$T_n = \xi_{n+1} T_{n+1} + \eta_{n+1} => -k_N(T_N - \xi_N T_N + \eta_N) = \alpha_N(T_N - T_0)h + \varphi(T_N)h$$

приведем подобные слагаемые и получим уравнение относительно T_N:

$$\varphi(T_N)h + (\alpha_N h + k_N - \xi_N k_N)T_N + k_N \eta_N - \alpha_N T_0 h \ = \ 0 \qquad (18)$$

<u>3. Опишите алгоритм применения метода прогонки,</u>

если при $x = 0$ краевое условие линейное (как в настоящей работе), а при $x = l$, как в п.2.

Алгоритм применения метода прогонки такой же, как и описано выше. Единственное отличие - вычисление T_N, так как краевое условие при x=l теперь нелинейно. T_N найдем решая уравнение (18) методом дихотомии.

4. Опишите алгоритм определения y_p

единственного значения сеточной функции в **одной** заданной точке p. Использовать встречную прогонку, т.е. комбинацию правой и левой прогонок (лекция №8). Краевые условия линейные.

Комбинация левой и правой прогонки позволяет распараллелить вычисления значений искомой функции. Левой прогонкой $1 \le y_i \le$, правой прогонкой $\le n \le$, $= \frac{n}{2}$. Для правой прогонки вычислим коэффициенты как было приведено выше.

Для левой прогонки вычислим коэффициенты следующим образом:

$$y_{n+1} = \xi_n y_n + \eta_n$$

Подставим это уравнение в систему линейных уравнений с трехдиагональной матрицей:

$$A_n y_{n-1} - B_n y_n + C_n y_{n+1} = -F_n$$

Получим

$$A_n y_{n-1} - B_n y_n + C_n(\xi_n y_n + \eta_n) = -F_n$$

Приведем подобные слагаемые

$$y_n = -\frac{A_n}{B_n + C_n \xi_n} y_{n-1} + \frac{F_n - C_n \eta_n}{B_n + C_n \xi_n}$$

$$\xi_{n+1} = -\frac{A_n}{B_n + C_n \xi_n} = \alpha_{n+1} \ ; \ \eta_{n+1} = \frac{F_n - C_n \eta_n}{B_n + C_n \xi_n} = \beta_{n+1}$$

Из основных формул левой и правой прогонки

$$y_n = \alpha_{n-1} y_{n-1} + \beta_{n-1} \quad ; \quad y_{n-1} = \xi_n y_n + \eta_n$$

Составим систему

$$\begin{cases} y_{p-1} = \xi_p y_p + \eta_p \\ y_p = \alpha_{p-1} y_{p-1} + \beta_{p-1} \end{cases}$$

Решая систему найдем

$$y_p = \frac{\alpha_{p-1}\eta_p + \beta_{p-1}}{1 - \xi_p \alpha_{p-1}}$$