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

ФАКУЛЬТЕТ «Информатика и системы управления»
КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3

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

Студент Куприй А. А.
Группа ИУ7-63Б
Преподаватель Градов В. М.

Москва, 2020 г.

ВВЕДЕНИЕ

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

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Исходные данные

$$K_0 = 0.4$$
$$K_n = 0.1$$
$$\alpha_0 = 0.05$$
$$\alpha_n = 0.001$$
$$l = 10$$
$$T_0 = 300$$
$$R = 0.5$$
$$F_0 = 50$$

Уравнение для функции $T(x)$

$$\frac{d}{dx}\left(k(x)\frac{dT}{dx}\right) - \frac{2}{R}\alpha(x)T + \frac{2T_0}{R}\alpha(x) = 0 \tag{0.1}$$

Краевые условия

$$\begin{cases} x = 0, -k(0)\frac{dT}{dx} = F_0, \\ x = l, -k(l)\frac{dT}{dx} = \alpha_N\big(T(l) - T_0\big) \end{cases}$$

Функция $k(x)$ представлена на формуле 0.2

$$k(x) = \frac{a}{x - b} \tag{0.2}$$

, где

$$a = -K_0 b = \frac{K_0 K_n l}{K_0 - K_n}$$

$$b = \frac{K_N l}{K_N - K_0}$$

Функция $\alpha(x)$ представлена на формуле 0.3

$$\alpha(x) = \frac{c}{x-d} \tag{0.3}$$

, где

$$c = -\alpha_0 d = \frac{\alpha_0 \alpha_N l}{\alpha_0 - \alpha_N}$$

$$d = \frac{\alpha_n l}{\alpha_n - \alpha_0}$$

Разностная схема

$$A_n y_{n+1} - B_n y_n + C_n y_{n-1} = -D_n, 1 \leq n \leq N-1 \tag{0.4}$$

$$K_0 y_0 + M_0 y_1 = P_0 \tag{0.5}$$

$$K_n y_n + M_n u_{n-1} = P_n \tag{0.6}$$

$$A_n = \frac{x_{n+\frac{1}{2}}}{h}, \, C_n = \frac{x_{n-\frac{1}{2}}}{h}, \; B_n = A_n + C_n + p_n h, \, D_n = f_n h$$

Метод трапеций

$$x_{n\pm\frac{1}{2}} = \frac{2k_n k_{n\pm1}}{k_n + k_{n\pm1}}$$

Краевые условия

$$F = -k(x)\frac{dT}{dx}$$

$$p(x) = \frac{2}{R}\alpha(x)$$

$$f(x) = \frac{2T_0}{R}\alpha(x)$$

$$p_n = p(x_n), f_n = f(x_n)$$

Разностные аналоги кравевых условий при $x = 0$

$$y_0 \cdot \left(x_{\frac{1}{2}} + \frac{h^2}{8}p_{\frac{1}{2}} + \frac{h^2}{4}p_0\right) - y_1 \cdot \left(x_{\frac{1}{2}} - \frac{h^2}{8}p_{\frac{1}{2}}\right) = \left(hF_0 + \frac{h^2}{4}\big(f_{\frac{1}{2}} + f_0\big)\right) \quad (0.7)$$

Для $p_{\frac{1}{2}}$ и $f_{\frac{1}{2}}$ можно принять простую аппроксимацию

$$p_{\frac{1}{2}} = \frac{p_0 + p_1}{2}$$

$$f_{\frac{1}{2}} = \frac{f_0 + f_1}{2}$$

Разностные аналоги кравевых условий при $x = l$. Проинтегрируем 0.1 на отрезке $\left[x_n - \frac{1}{2}; x_n\right]$

$$-\int_{X_{n-\frac{1}{2}}}^{X_n} \frac{dF}{dx}dx - \int_{X_{n-\frac{1}{2}}}^{X_n} p(x)T dx + \int_{X_{n-\frac{1}{2}}}^{X_n} f(x)dx = 0$$

Второй и третий интегралы вычислим методом трапеций

$$F_{n-\frac{1}{2}} - F_n - \frac{p_{n-\frac{1}{2}}y_{n-\frac{1}{2}} + p_n y_n}{4}h + \frac{f_{n-\frac{1}{2}} + f_n}{4}h = 0$$

Подставим в полученное уравнение

$$F_{n-\frac{1}{2}} = x_{n-\frac{1}{2}}\frac{y_{n-1} - y_n}{h}$$

$$F_n = \alpha_(y_n - T_0)$$

$$y_{n-\frac{1}{2}} = \frac{y_n + Y_{n-1}}{2}$$

Получим

$$\frac{x_{n-\frac{1}{2}} y_{n-1}}{h} - \frac{x_{n-\frac{1}{2}} y_n}{h} - \alpha_n y_n + \alpha_n T_0 - \frac{p_{n-\frac{1}{2}} y_{n-1}}{8} h - \frac{p_{n-\frac{1}{2}} y_n}{8} h - \frac{p_n y_n}{4} h + \frac{f_{n-\frac{1}{2}} + f_n}{4} h = 0$$

$$y_n \cdot \left(-\frac{x_{n-\frac{1}{2}}}{h} - \alpha_n - \frac{p_n}{4} h - \frac{p_{n-\frac{1}{2}}}{8} h \right) + y_{n-1} \cdot \left(\frac{x_{n-\frac{1}{2}}}{h} - \frac{p_{n-\frac{1}{2}}}{8} h \right) = -\left(\alpha_n T_0 + \frac{f_{n-\frac{1}{2}} + f_n}{4} h \right) \tag{0.8}$$

С помощью формул 0.5 и 0.7 получаем коэффициенты K_0, M_0 и P_0, а с помощью 0.6 и 0.8 получаем K_n, M_n и P_n.

Метод прогонки

Для решения системы из 0.4, 0.5 и 0.6 используется метод прогонки, который состоит из двух этапов: прямой ход и обратный ход.

Прямой ход

Для коэффициентов ε и η нужны начальные значения (формулы 0.9 и 0.10).

$$\varepsilon_1 = -\frac{M_0}{K_0} \tag{0.9}$$

$$\eta_1 = \frac{P_0}{K_0} \tag{0.10}$$

Затем вычисляются остальные элементы массива прогоночных коэффициентов (формула 0.11).

$$y_n = \underbrace{\frac{C_n}{B_n - A_n\varepsilon_n}}_{\varepsilon_{n+1}} y_{n+1} + \underbrace{\frac{D_n + A_n\eta_n}{B_n - A_n\varepsilon_n}}_{\eta_{n+1}} \tag{0.11}$$

Обратный ход

По формуле 0.12 находим начальное значение y_n.

$$y_n = \frac{P_n - M_n\eta_n}{K_n + M_n\varepsilon_n} \tag{0.12}$$

Остальные значения находятся по формуле 0.13.

$$y_n = \varepsilon_{n+1}y_{n+1} + \eta_{n+1} \tag{0.13}$$

Полученный массив y будет искомый массив $T(x)$.

ПРАКТИЧЕСКАЯ ЧАСТЬ

Ниже представленны листинги программы:

Листинг 1 — Исходные данные

```
K0 = 0.4
KN = 0.1
alpha0 = 0.05
alphaN = 0.01
l = 10
T0 = 300
R = 0.5
F0 = 50
h = 1e-3
```

Листинг 2 — Параметры для краевых условий

```
b = (KN * l) / (KN - K0)
a = - K0 * b
d = (alphaN * l) / (alphaN - alpha0)
c = - alpha0 * d
```

Листинг 3 — Коэффициенты теплопроводности и теплоотдачи

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

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

Листинг 4 — Выполнение замены

```
def p(x):
    return 2 * alpha(x) / R

def f(x):
    return 2 * alpha(x) * T0 / R
```

Листинг 5 — Простая аппроксимация

```
def x_plus_half(x):
    return (k(x) + k(x + h)) / 2

def x_minus_half(x):
    return (k(x) + k(x - h)) / 2
```

Листинг 6 — Разностная схема

```
def A(x):
    return x_plus_half(x) / h

def C(x):
    return x_minus_half(x) / h

def B(x):
    return A(x) + C(x) + p(x) * h

def D(x):
    return f(x) * h
```

Листинг 7 — Краевые условия при x

```
K0 = x_plus_half(0) + h * h * (p(0) + p(h)) / 16 + h * h * p(0) / 4
M0 = -(x_plus_half(0) - h * h * (p(0) + p(h)) / 16)
P0 = h * F0 + h * h / 4 * ((f(0) + f(h)) / 2 + f(0))
```

Листинг 8 — Краевые условия при x

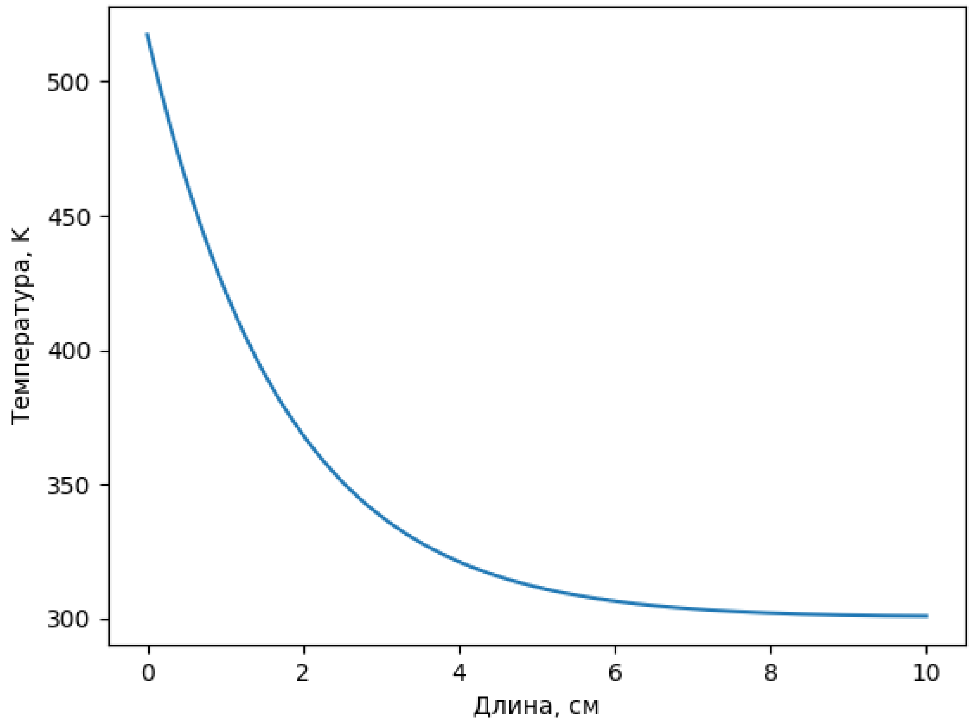
```
KN = -x_minus_half(l) / h - alphaN - p(l) * h / 4 - (p(l) + p(l - h)) * h /
    16
MN = x_minus_half(l) / h - (p(l) + p(l - h)) * h / 16
PN = -(alphaN * T0 + ((f(l) + f(l - h)) / 2 + f(l)) * h / 4)
```

Листинг 9 — Метод прогонки

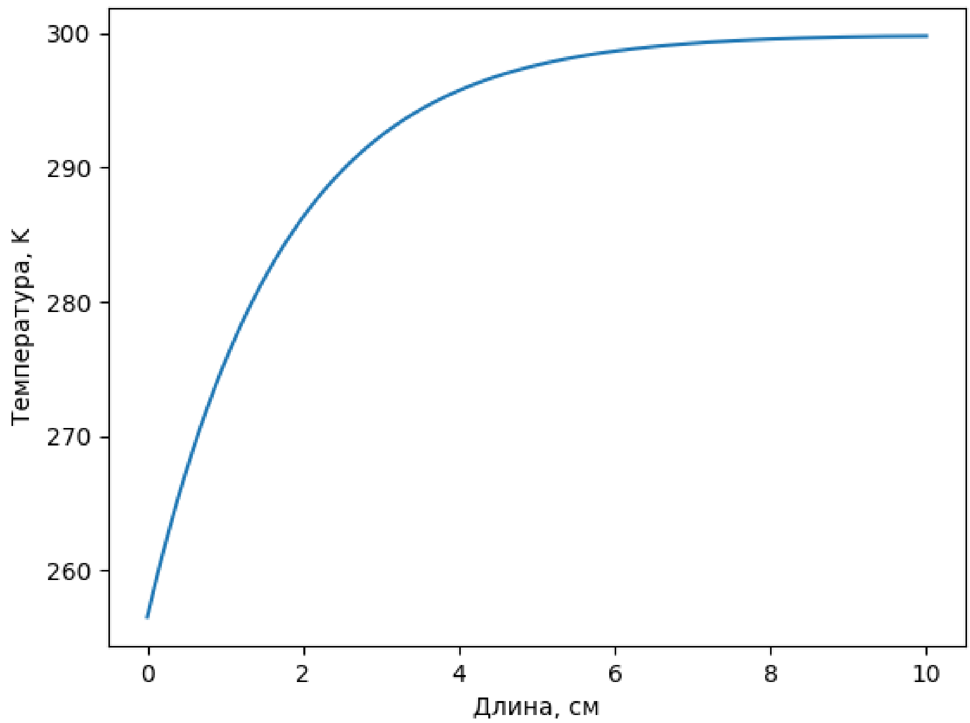
```
def run():
    eps = [0, -M0 / K0]
    eta = [0, P0 / K0]

    x = h
    n = 1
    while (x + h < l):
        eps.append(C(x) / (B(x) - A(x) * eps[n]))
        eta.append((D(x) + A(x) * eta[n]) / (B(x) - A(x) * eps[n]))
        n += 1
        x += h

    t = [0] * (n + 1)
    t[n] = (PN - MN * eta[n]) / (KN + MN * eps[n])

    for i in range(n - 1, -1, -1):
        t[i] = eps[i + 1] * t[i + 1] + eta[i + 1]

    return t
```

На изображениях ниже представлены скриншот работы программы:

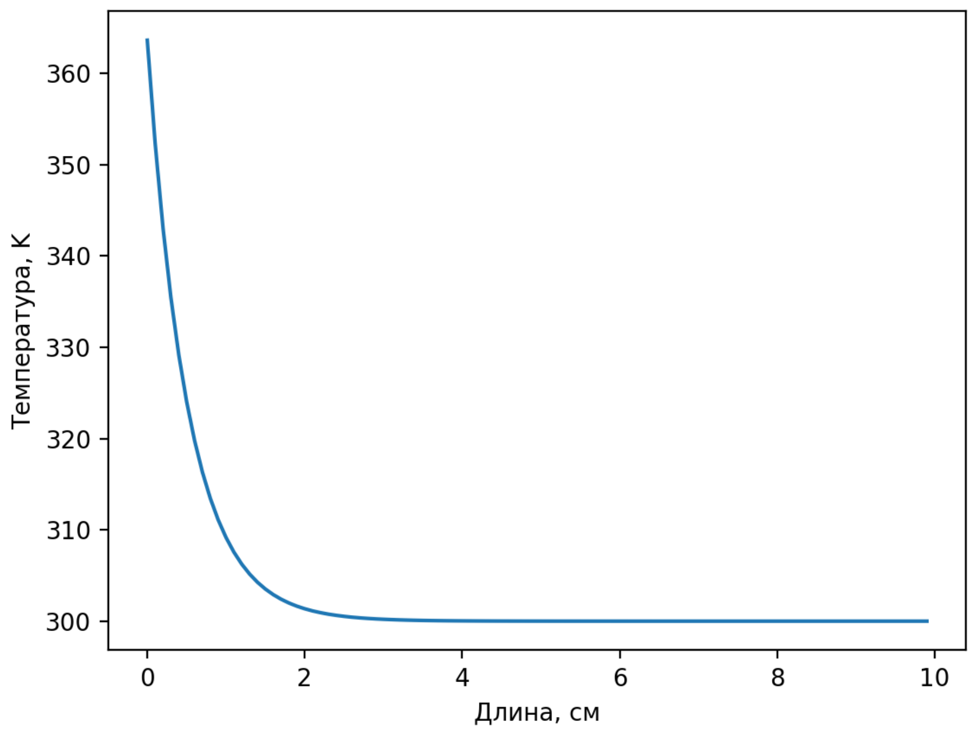
а) График зависимости температуры $T(x)$ от координаты x при заданных выше параметрах:



б) График зависимости $T(x)$ при F_0 = -10.

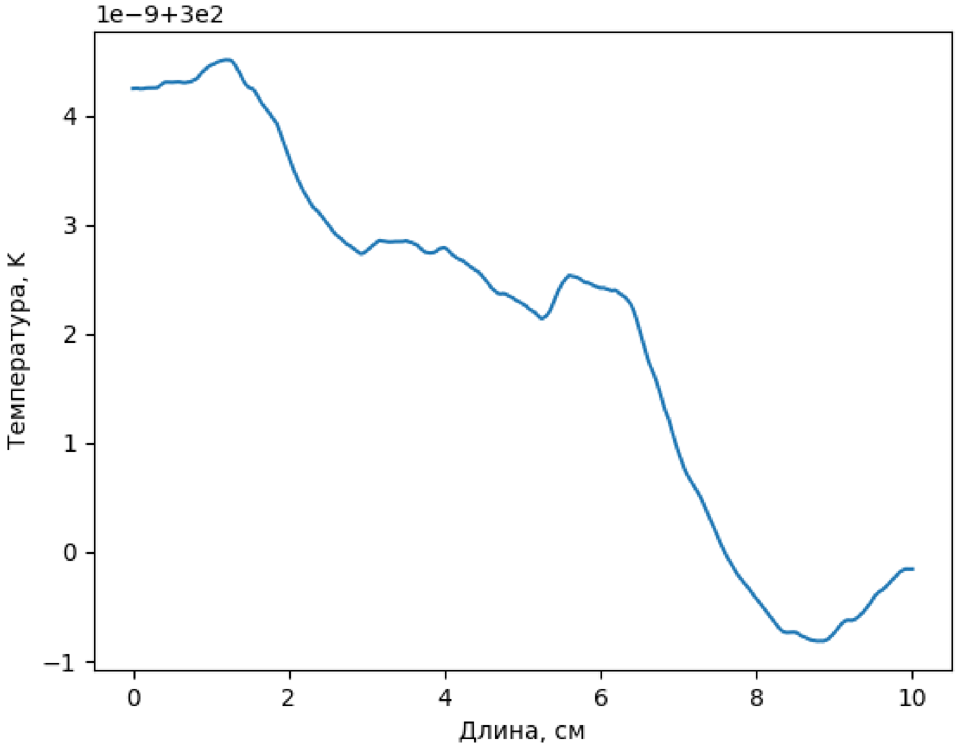


в) График зависимости при увеличенных значениях (например, в 3 раза).



Справка. При увеличении теплосъема и неизменном потоке уровень температур должен снижаться, а градиент увеличиваться.

г) График зависимости $T(x)$ при $F_0 = 0$. Так как используемая библиотека приближает график по осям таким образом, чтобы границами графика являлись минимальные и максимальные значения, на этом графике прямой не видно.



Для получения желаемого графика выведем значения, полученные программой, и посторим график по ним:

```
✘ san_sanchez@LEX  ~/workspace/model/lab_03/source  python3 main.py
0.0 300
0.2 300
0.4 300
0.6000000000000001 300
0.8 300
1.0 300
1.2000000000000002 300
1.4000000000000001 300
1.6 300
1.8 300
2.0 300
2.2 300
2.4000000000000004 300
2.6 300
2.8000000000000003 300
3.0 300
3.2 300
3.4000000000000004 300
3.6 300
3.8000000000000003 300
4.0 300
4.2 300
4.4 300
4.6000000000000005 300
4.800000000000001 300
5.0 300
5.2 300
5.4 300
5.6000000000000005 300
5.800000000000001 300
6.0 300
6.2 300
6.4 300
6.6000000000000005 300
6.800000000000001 300
7.0 300
7.2 300
7.4 300
7.6000000000000005 300
7.800000000000001 300
8.0 300
8.200000000000001 300
```

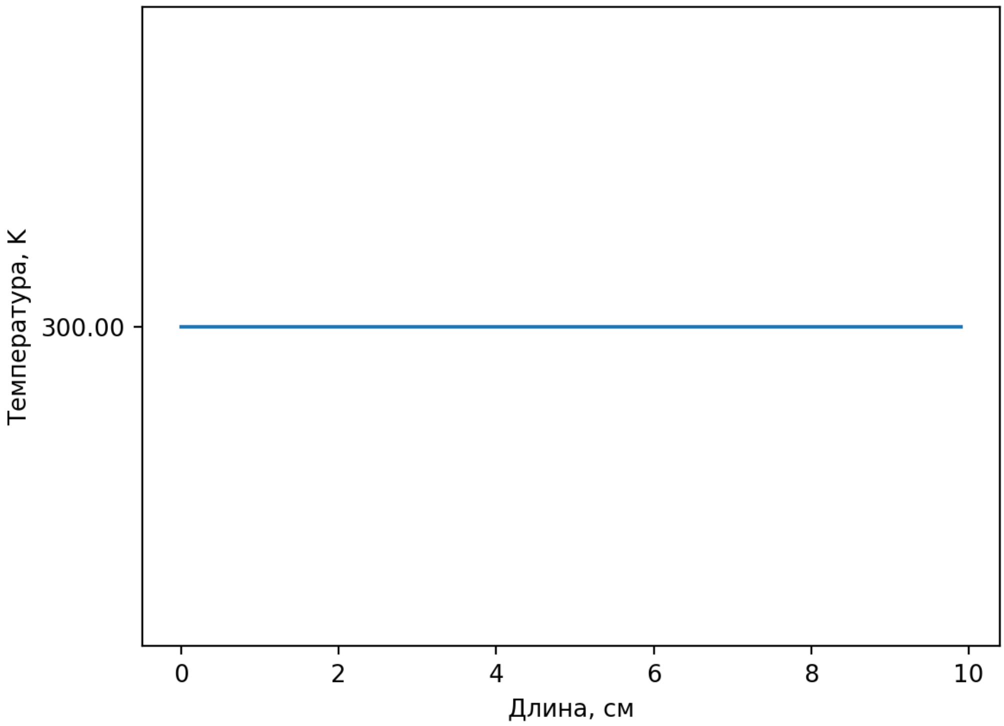


Рисунок 0.1 — Вывод программы

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).

ОТВЕТЫ НА ВОПРОСЫ.

Какие способы тестирования программы можно предложить?

Можно предложить три разных способа тестирования:

— Ввести F_0 меньше нуля. Это означает, что съем тепла идет слева, поэтому температура будет увеличиваться от 0 до l.

— Ввести F_0 равным нулю. Это значит, что тепловое нагружение отсутствует, поэтому температура стержня будет везде равна температуре окружающей среды.

— Увеличить значения коэффициента теплоотдачи в несколько раз, Это значит, что стержень будет отдавать больше тепла и скорость снижения температуры будет увеличена.

Получите простейший аналог нелинейного краевого условия при

$$x = l, -k(l)\frac{dT}{dx} = \alpha_N\big(T(l) - T_0\big) + \varphi(T)$$

, где $\varphi(T)$ – заданная функция. Производную аппроксимируйте односторонней разностью.

Заменяем производную разностью

$$-k(l)\frac{T(x+h) - T(x)}{h} = \alpha_N\big(T(l) - T_0\big) + \varphi(T)$$

При $x = l$ полуаем

$$-k(l)\frac{T(l) - T(l-1)}{h} = \alpha_N\big(T(l) - T_0\big) + \varphi\big(T(l)\big)$$

Заменим $k(l) = k_l, T(l) = T_l$

$$k_l T_{l-1} - k_l T_l = \alpha_N T_l h - \alpha_N T_0 h + \varphi(T_l)h$$

$$\big(k_l + \alpha_N h\big)T_l - k_l T_{l-1} = \big(\alpha_N T_0 - \varphi(T_l)\big)h$$

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

$$\begin{cases} x = 0, -k(0)\frac{dT}{dx} = F_0 \\ x = l, -k(l)\frac{dT}{dx} = \alpha_N\big(T(l) - T_0\big) + \varphi(T) \end{cases}$$

Поскольку краевое условие при $x = 0$ линейное, то будем использовать правую прогонку.

$$y_n = \varepsilon_{n+1}y_{n+1} + \eta_{n+1}$$

Используем аппроксимацию первого порядка точности для краевого условия $x = 0$

$$-k_0\frac{T_1 - T_0}{h} = F_0$$

$$T_0 - T_1 = \frac{F_0 h}{k_0}$$

$$\begin{cases} K_0 = 1 \\ M_0 = -1 \\ P_0 = \frac{F_0 h}{k_0} \end{cases}$$

Разностная аппроксимация для краевого условия $x = l$ будет иметь вид

$$-k_l\frac{T_l - T_{l-1}}{h} = \alpha_N\big(T_l - T_0\big) + \varphi(T_l)$$

$$k_l T_{l-1} + k_l T_l = \alpha_N T_l h - \alpha_N T_0 h + \varphi(T_l)h$$

$$k_l T_{l-1} + (k_l - \alpha_N h)T_l = \varphi(T_l)h - \alpha_N T_0 h$$

$$\begin{cases} K_l = k_l - \alpha_N h \\ M_l = k_l \\ P_l = \varphi(T_l)h - \alpha_N T_0 h \end{cases}$$

Начальные прогоночные коэффициенты будут равны

$$\begin{cases} \varphi_1 = -\frac{M_0}{K_0} \\ \eta_1 = \frac{P_0}{K_0} \end{cases}$$

Значение T в точке l будет равно

$$T_l = \frac{P_l - M_l \eta_l}{K_l + M_l \varepsilon_l}$$

Опишите алгоритм определения единственного значения сеточной функции y_p в одной заданной точке p. Использовать встречную прогонку, то есть комбинацию правой и левой прогонок. Краевые условия линейны.

Метод встречных прогонок подразумевает, что на промежутке $0 \leq n \leq p+1$ будет использоваться правая прогонка, а на промежутке $p \leq n \leq N$ – левая.

Тогда коэффициенты для правой прогонки будут

$$\varepsilon_{n+1} = \frac{C_n}{B_n - A_n \varepsilon_n}$$

$$\eta_{n+1} = \frac{D_n + A_n \eta_n}{B_n - A_n \varepsilon_n}$$

$$0 \leq n \leq p+1$$

Для левой прогонки:

$$\xi_{n-1} = \frac{C_n}{B_n - A_n \xi_n}$$

$$\pi_{n-1} = \frac{A_n \pi_n + D_n}{B_n - A_n \xi_n}$$

$$p \le n \le N$$

Тогда

$$\begin{cases} y_n = \varepsilon_{n+1} y_{n+1} + \eta_{n+1} \\ y_{n+1} = \xi_n y_n + \pi_n \end{cases}$$

Подставим вместо n p, получим

$$\begin{cases} y_p = \varepsilon_{p+1} y_{p+1} + \eta_{p+1} \\ y_{p+1} = \xi_p y_p + \pi_p \end{cases}$$

$$y_p = \varepsilon_{p+1} \xi_p y_p + \varepsilon_{p+1} \pi_p + \eta_{p+1}$$

$$y_p = \frac{\varepsilon_{p+1} \pi_p + \eta_{p+1}}{1 - \varepsilon_{p+1} \xi_p}$$