



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	Степанов А. О.
Группа	ИУ7-63Б
Оценка (баллы)	
Преподаватель	Градов В.М.

Москва, 2020 г.

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

ИСХОДНЫЕ ДАННЫЕ

Данные для тестирования

$$K_0 = 0.4$$

$$K_n = 0.1$$

$$\alpha_0 = 0.05$$

$$\alpha_n = 0.001$$

$$l = 10$$

$$T_0 = 300$$

$$R = 0.5$$

$$F_0 = 50$$

Уравнение для функции $T(x)$

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

Краевые условия

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

Функция $k(x)$ представлена на формуле 2

$$k(x) = \frac{a}{x - b} \quad (2)$$

, 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 3

$$\alpha(x) = \frac{c}{x - d} \quad (3)$$

, 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

$$A_n y_{n+1} - B_n y_n + C_n y_{n-1} = -D_n, 1 \leq n \leq N-1 \quad (4)$$

$$K_0 y_0 + M_0 y_1 = P_0 \quad (5)$$

$$K_n y_n + M_n u_{n-1} = P_n \quad (6)$$

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

Метод трапеций

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

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 краевых условий при $x = 0$

$$y_0 \cdot \left(x_{\frac{1}{2}} + \frac{h^2}{8} p_{\frac{1}{2}} + \frac{h^2}{4} p_0 \right) - y_1 \cdot \left(x_{\frac{1}{2}} - \frac{h^2}{8} p_{\frac{1}{2}} \right) = \left(hF_0 + \frac{h^2}{4} (f_{\frac{1}{2}} + f_0) \right) \quad (7)$$

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 краевых условий при $x = 1$. Проинтегрируем 1 на отрезке $[x_n - \frac{1}{2}; x_n]$

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

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

$$\frac{x_{n-\frac{1}{2}}y_{n-1}}{h} - \frac{x_{n-\frac{1}{2}}y_n}{h} - \alpha_n y_n + \alpha_n t_0 - \frac{p_{n-\frac{1}{2}}y_{n-1}}{8}h - \frac{p_{n-\frac{1}{2}}y_n}{8}h - \frac{p_n y_n}{4}h + \frac{f_{n-\frac{1}{2}} + f_n}{4}h = 0$$

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

С помощью формул 5 и 7 получаем коэффициенты K_0 , M_0 и P_0 , а с помощью 6 и 8 получаем K_n , M_n и P_n .

Метод прогонки

Для решения системы из 4, 5 и 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 9 и 10).

$$\varepsilon_1 = -\frac{M_0}{K_0} \quad (9)$$

$$\eta_1 = \frac{P_0}{K_0} \quad (10)$$

Затем вычисляются остальные элементы массива прогоночных коэффициентов (формула 11).

$$y_n = \underbrace{\frac{C_n}{B_n - A_n \varepsilon_n}}_{\varepsilon_{n+1}} y_{n+1} + \underbrace{\frac{D_n + A_n \eta_n}{B_n - A_n \varepsilon_n}}_{\eta_{n+1}} \quad (11)$$

Обратный ход

По формуле 12 находим начальное значение y_n .

$$y_n = \frac{P_n - M_n \eta_n}{K_n + M_n \varepsilon_n} \quad (12)$$

Остальные значения находятся по формуле 13.

$$y_n = \varepsilon_{n+1} y_{n+1} + \eta_{n+1} \quad (13)$$

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

ЛИСТИНГИ

Листинг 1: Расчет начальных коэффициентов

```

1  Mathematics::Mathematics(
2      const double k0,
3      const double kn,
4      const double a0,
5      const double an,
6      const double F0
7      const bool ax3
8  )
9      : _multiAlpha(ax3 ? 3.0 : 1.0 ),
10      _k0(k0), _kn(kn), _alpha0(a0), _alphaN(an), _F0(F0),
11      _a((_k0 * _kn * _1) / (_k0 - _kn)),
12      _b((_kn * _1) / (_kn - _k0)),
13      _c((_alpha0 * _alphaN * _1) / (_alpha0 - _alphaN)),
14      _d((_alphaN * _1) / (_alphaN - _alpha0))
15  {
16      double x1_2 = (2 * k(0) * k(_h)) / (k(0) + k(_h));
17      double p0 = p(0);
18      double p1_2 = (p0 + p(_h)) / 2.0;
19      double f0 = f(0);
20      double f1_2 = (f0 + f(_h)) / 2.0;
21

```

```

22  _K0 = x1_2 + (_h * _h / 8.0) * p1_2 + (_h * _h / 4.0) * p0;
23  _M0 = -x1_2 + (_h * _h / 8.0) * p1_2;
24  _P0 = _h * _F0 + (_h * _h / 4.0) * (f1_2 + f0);
25
26  double xN1_2 = (2 * k(_l) * k(_l - _h)) / (k(_l) + k(_l - _h));
27  double alphaN = alpha(_l);
28  double pN = p(_l);
29  double pN1_2 = (pN + p(_l - _h)) / 2.0;
30  double fN = f(_l);
31  double fN1_2 = (fN + f(_l - _h)) / 2.0;
32
33  _KN = -(xN1_2 / _h) - alphaN - (pN / 4.0) * _h - (pN1_2 / 8.0) * _h;
34  _MN = (xN1_2 / _h) - (pN1_2 / 8.0 * _h);
35  _PN = -alphaN * _T0 - ((fN1_2 + fN) / 4.0) * _h;
36
37  findCoeff();
38  findResult();
39 }

```

Листинг 2: Функции $k(x)$ $\alpha(x)$ $p(x)$ и $f(x)$

```

1  double Mathematics::k(const double x)
2  {
3      return _a / (x - _b);
4  }
5
6  double Mathematics::alpha(const double x)
7  {
8      return _multiAlpha * (_c / (x - _d));
9  }
10
11 double Mathematics::p(const double x)
12 {
13     return (2 * alpha(x)) / _R;
14 }
15
16 double Mathematics::f(const double x)
17 {
18     return (2 * _T0 * alpha(x)) / _R;
19 }

```

Листинг 3: Прямой ход

```

1  void Mathematics::findCoeff()

```

```

2 {
3     _eps.append(-_M0 / _K0);
4     _eta.append(_P0 / _K0);
5
6     for (double x = _h; x <= _l; x += _h) {
7         double epsLast = _eps.last();
8         double etaLast = _eta.last();
9         _eps.append(C(x) / (B(x) - A(x) * epsLast));
10        _eta.append((D(x) + A(x) * etaLast) / (B(x) - A(x) * epsLast));
11    }
12 }
13
14 double Mathematics::A(const double x)
15 {
16     double kn = k(x);
17     double kn1 = k(x - _h);
18     return ((2 * kn * kn1) / (kn + kn1)) / _h;
19 }
20
21 double Mathematics::B(const double x)
22 {
23     return A(x) + C(x) + p(x) * _h;
24 }
25
26 double Mathematics::C(const double x)
27 {
28     double kn = k(x);
29     double kn1 = k(x + _h);
30     return ((2 * kn * kn1) / (kn + kn1)) / _h;
31 }
32
33 double Mathematics::D(const double x)
34 {
35     return f(x) * _h;
36 }

```

Листинг 4: Обратный ход

```

1 void Mathematics::findResult()
2 {
3     X.append(_l);
4     Y.append((_PN - _MN * _eta.last()) / (_KN + _MN * _eps.last()));
5

```



```

6     for (int i = _eta.length() - 2; i >= 0; --i) {
7         X.append(i * _h);
8         Y.append(_eps[i + 1] * Y.last() + _eta[i + 1]);
9     }
10 }

```

РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

График при вводе исходных данных

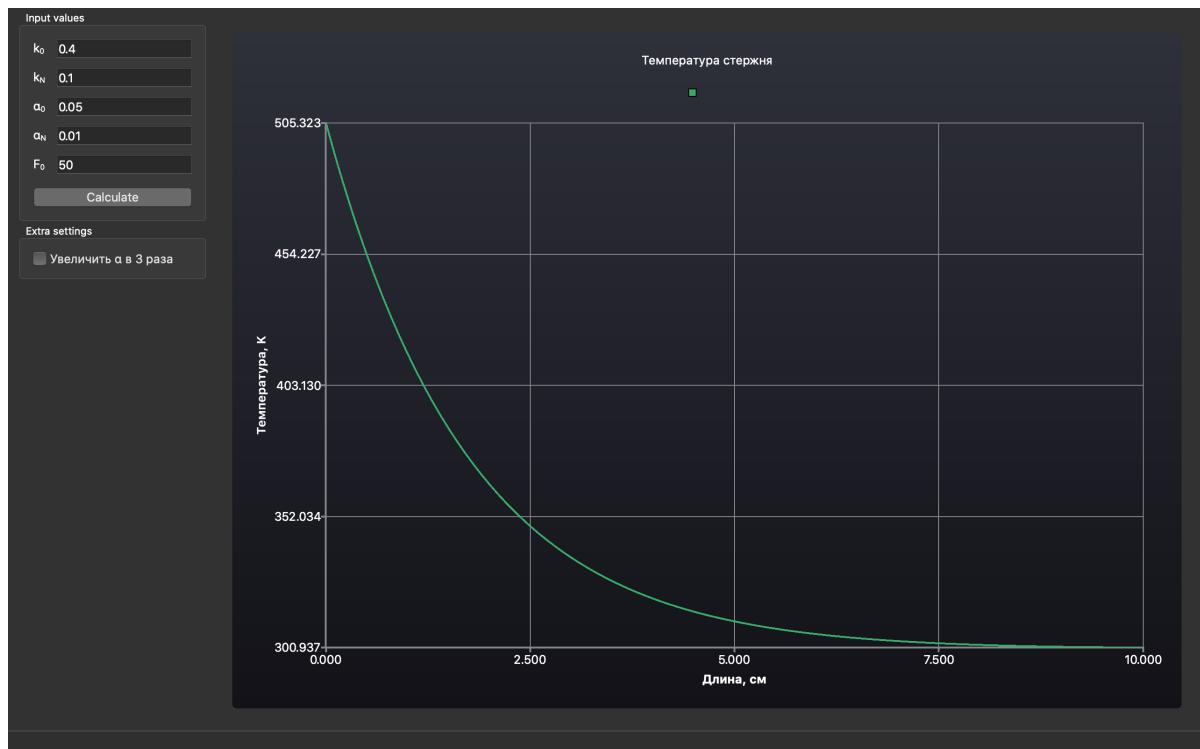


Рис. 1: Исходные данные

График при $F_0 = -10$

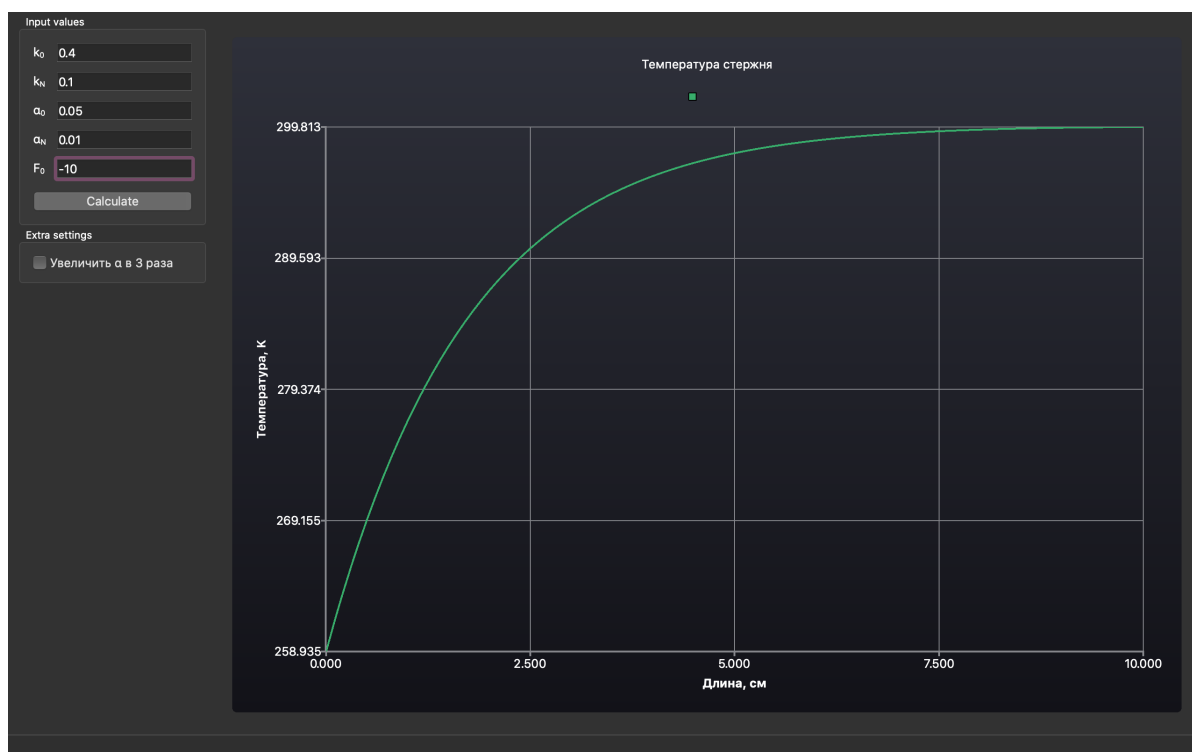


Рис. 2: $F_0 = -10$

График при α увеличенное в 3 раза

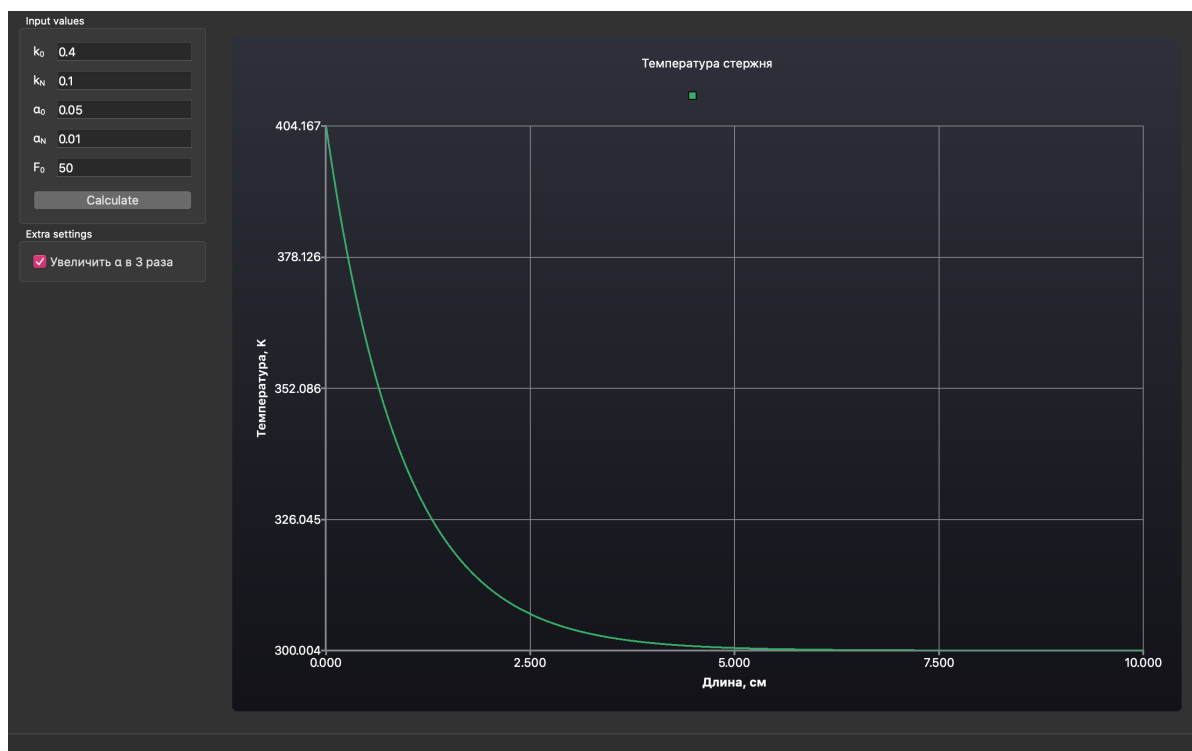


Рис. 3: α увеличенное в 3 раза

График при $F_0 = 0$

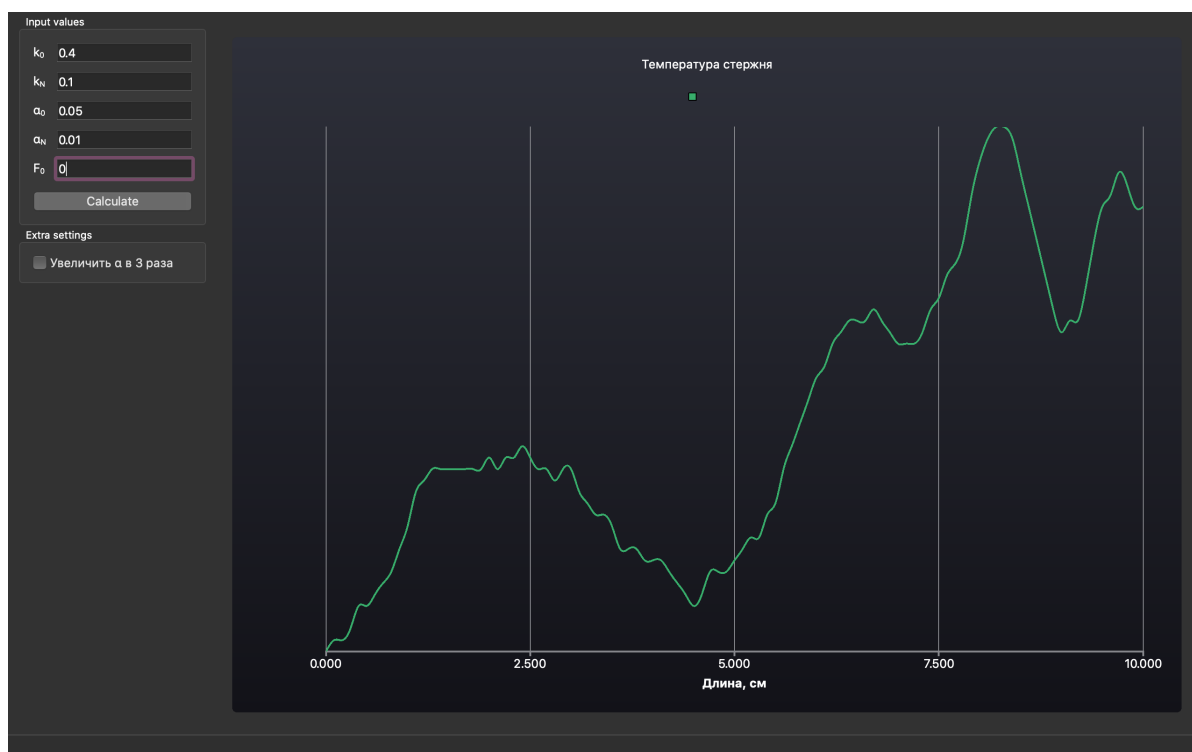


Рис. 4: $F_0 = 0$

Библиотека **QtCharts** приближает график по осям так, чтобы границы являлись максимальными и минимальными значениями графика, поэтому на этом графике не видно прямой. Выведем значения полученные программой и построим по ним график.

[illegible]

Рис. 5: Вывод программы

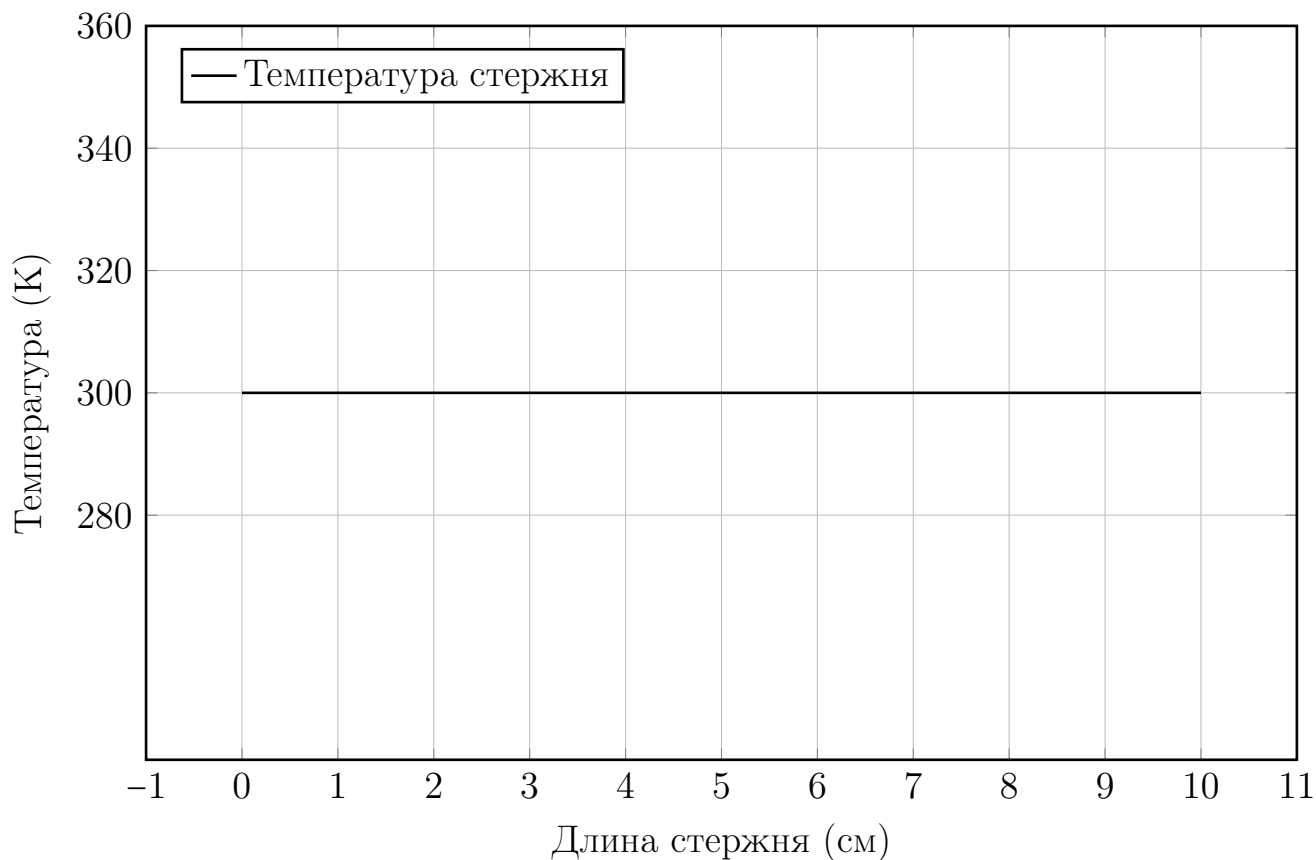


Рис. 6: $F_0 = 0$

ВОПРОСЫ

Какие способы тестирования программы можно предложить?

1. Ввести F_0 (тепловой поток) меньше нуля. Это означает, что съем тепла идет слева, поэтому температура будет увеличиваться от 0 до l .
2. Ввести F_0 (тепловой поток) равным нулю. Это значит, что тепловое нагружение отсутствует, поэтому температура стержня будет везде равна температуре окружающей среды.
3. 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 $x = l$

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

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

Заменяем производную разностью

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

При $x = l$ получаем

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

Заменим $k(l) = k_l, T(l) = T_l$

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

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

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

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

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:

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

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

$$p \leq n \leq N$$

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

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

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

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