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: Получние 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 $T(x)$

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

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 2

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

, 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

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

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

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

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

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 $x = 0$

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

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 1 на отрезке $\left[x_n - \frac{1}{2}; x_n\right]$

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

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

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

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

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).

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

Обратный ход

По формуле 12 находим начальное значение y_n.

$$y_n = \frac{P_n - M_n\eta_n}{K_n + M_n\varepsilon_n} \tag{12}$$

Остальные значения находятся по формуле 13.

$$y_n = \varepsilon_{n+1}y_{n+1} + \eta_{n+1} \tag{13}$$

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

ЛИСТИНГИ

Листинг 1: Расчет начальных коэффициентов

```
Mathematics::Mathematics(
    const double k0,
    const double kn,
    const double a0,
    const double an,
    const double F0
    const bool ax3
)
    : _multiAlpha(ax3 ? 3.0 : 1.0 ),
    _k0(k0), _kn(kn), _alpha0(a0), _alphaN(an), _F0(F0),
    _a((_k0 * _kn * _l) / (_k0 - _kn)),
    _b((_kn * _l) / (_kn - _k0)),
    _c((_alpha0 * _alphaN * _l) / (_alpha0 - _alphaN)),
    _d((_alphaN * _l) / (_alphaN - _alpha0))
{
    double x1_2 = (2 * k(0) * k(_h)) / (k(0) + k(_h));
    double p0 = p(0);
    double p1_2 = (p0 + p(_h)) / 2.0;
    double f0 = f(0);
    double f1_2 = (f0 + f(_h)) / 2.0;

```

```
    _K0 = x1_2 + (_h * _h / 8.0) * p1_2 + (_h * _h / 4.0) * p0;
    _M0 = -x1_2 + (_h * _h / 8.0) * p1_2;
    _P0 = _h * _F0 + (_h * _h / 4.0) * (f1_2 + f0);

    double xN1_2 = (2 * k(_l) * k(_l - _h)) / (k(_l) + k(_l - _h));
    double alphaN = alpha(_l);
    double pN = p(_l);
    double pN1_2 = (pN + p(_l - _h)) / 2.0;
    double fN = f(_l);
    double fN1_2 = (fN + f(_l - _h)) / 2.0;

    _KN = -(xN1_2 / _h) - alphaN - (pN / 4.0) * _h - (pN1_2 / 8.0) * _h;
    _MN = (xN1_2 / _h) - (pN1_2 / 8.0 * _h);
    _PN = -alphaN * _T0 - ((fN1_2 + fN) / 4.0) * _h;

    findCoeff();
    findResult();
}
```

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

```
double Mathematics::k(const double x)
{
    return _a / (x - _b);
}

double Mathematics::alpha(const double x)
{
    return _multiAlpha * (_c / (x - _d));
}

double Mathematics::p(const double x)
{
    return (2 * alpha(x)) / _R;
}

double Mathematics::f(const double x)
{
    return (2 * _T0 * alpha(x)) / _R;
}
```

Листинг 3: Прямой ход

```
void Mathematics::findCoeff()
```

```
{
    _eps.append(-_M0 / _K0);
    _eta.append(_P0 / _K0);

    for (double x = _h; x <= _l; x += _h) {
        double epsLast = _eps.last();
        double etaLast = _eta.last();
        _eps.append(C(x) / (B(x) - A(x) * epsLast));
        _eta.append((D(x) + A(x) * etaLast) / (B(x) - A(x) * epsLast));
    }
}

double Mathematics::A(const double x)
{
    double kn = k(x);
    double kn1 = k(x - _h);
    return ((2 * kn * kn1) / (kn + kn1)) / _h;
}

double Mathematics::B(const double x)
{
    return A(x) + C(x) + p(x) * _h;
}

double Mathematics::C(const double x)
{
    double kn = k(x);
    double kn1 = k(x + _h);
    return ((2 * kn * kn1) / (kn + kn1)) / _h;
}

double Mathematics::D(const double x)
{
    return f(x) * _h;
}
```

Листинг 4: Обратный ход

```
void Mathematics::findResult()
{
    X.append(_l);
    Y.append((_PN - _MN * _eta.last()) / (_KN + _MN * _eps.last()));

```

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

РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

График при вводе исходных данных

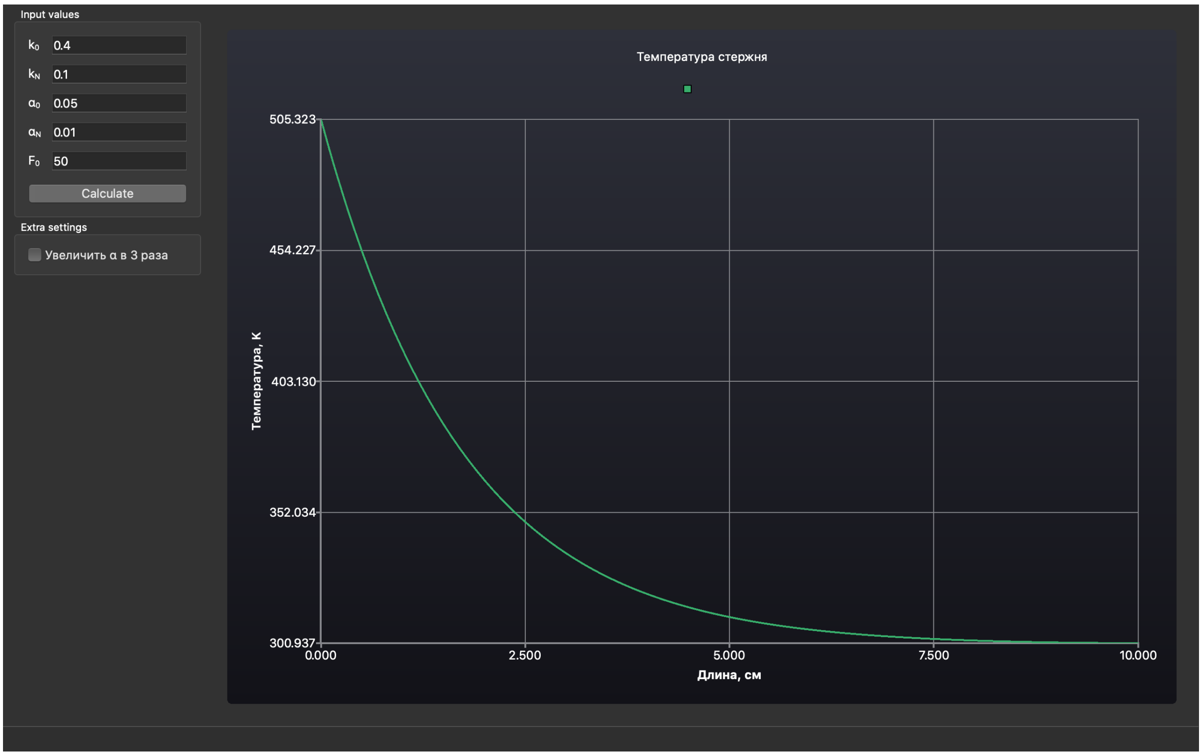

Рис. 1: Исходные данные

График при $F_0 = -10$

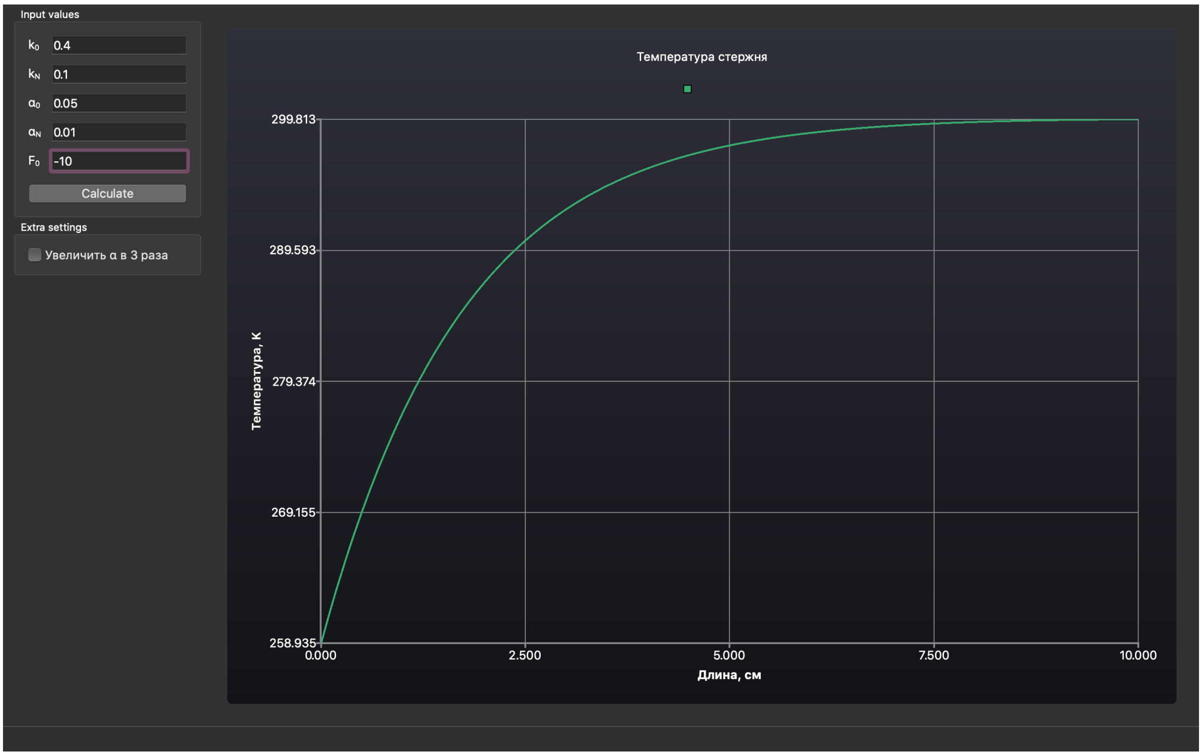

Рис. 2: $F_0 = -10$

График при α увеличенное в 3 раза

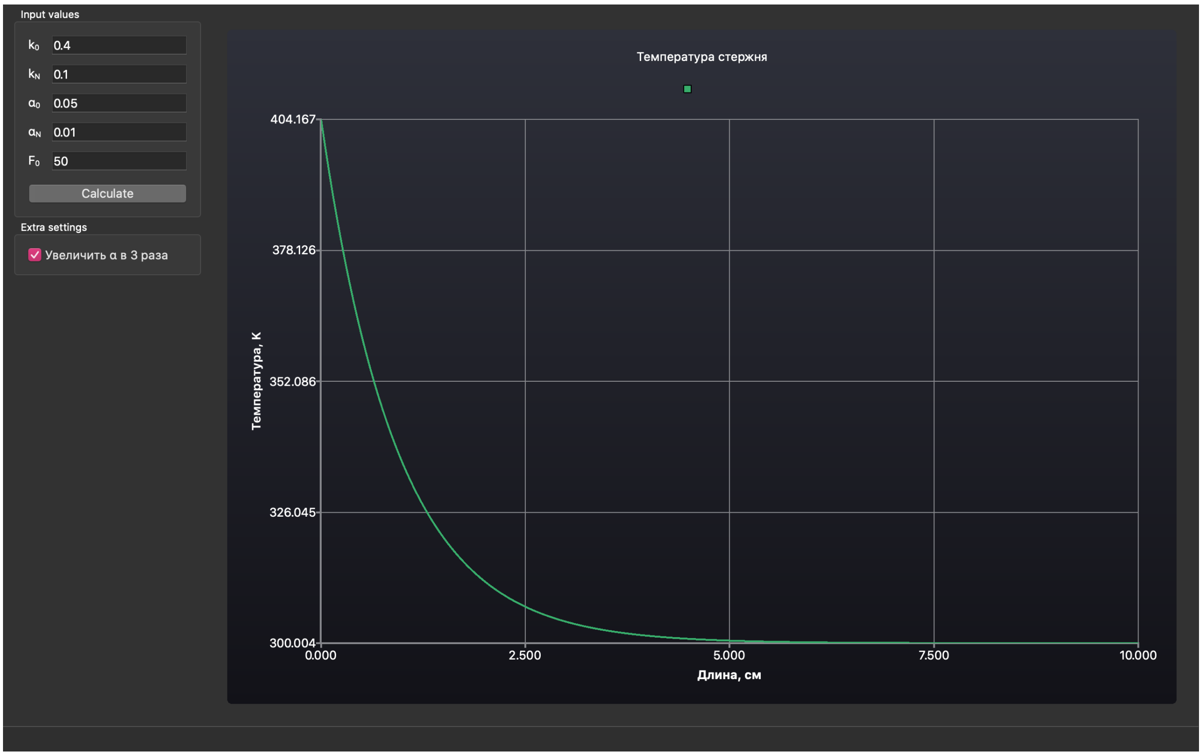

Рис. 3: α увеличенное в 3 раза

График при $F_0 = 0$

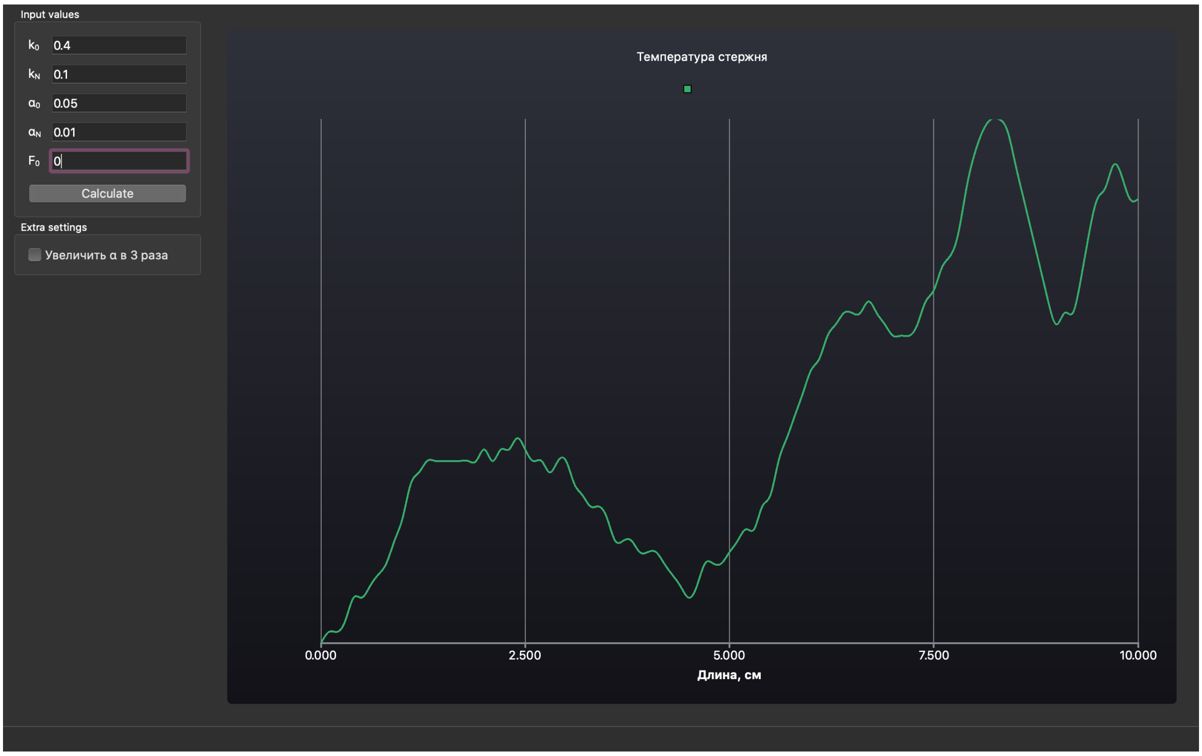

Рис. 4: $F_0 = 0$

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.

```
$ ./lab_03
QVector(10, 9.9, 9.8, 9.7, 9.6, 9.5, 9.4, 9.3, 9.2, 9.1, 9, 8.9, 8.8, 8.7, 8.6, 8.5, 8.4, 8.3, 8.2, 8.1, 8, 7.9, 7.8, 7.7, 7.6, 7.5, 7.4, 7.3, 7.2, 7.1, 7, 6
.9, 6.8, 6.7, 6.6, 6.5, 6.4, 6.3, 6.2, 6.1, 6, 5.9, 5.8, 5.7, 5.6, 5.5, 5.4, 5.3, 5.2, 5.1, 5, 4.9, 4.8, 4.7, 4.6, 4.5, 4.4, 4.3, 4.2, 4.1, 4, 3.9, 3.8, 3.7,
 3.6, 3.5, 3.4, 3.3, 3.2, 3.1, 3, 2.9, 2.8, 2.7, 2.6, 2.5, 2.4, 2.3, 2.2, 2.1, 2, 1.9, 1.8, 1.7, 1.6, 1.5, 1.4, 1.3, 1.2, 1.1, 1, 0.9, 0.8, 0.7, 0.6, 0.5, 0.
4, 0.3, 0.2, 0.1, 0)
QVector(300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300,
 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 3
00, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300, 300
, 300, 300, 300, 300, 300, 300, 300, 300)
```

Рис. 5: Вывод программы

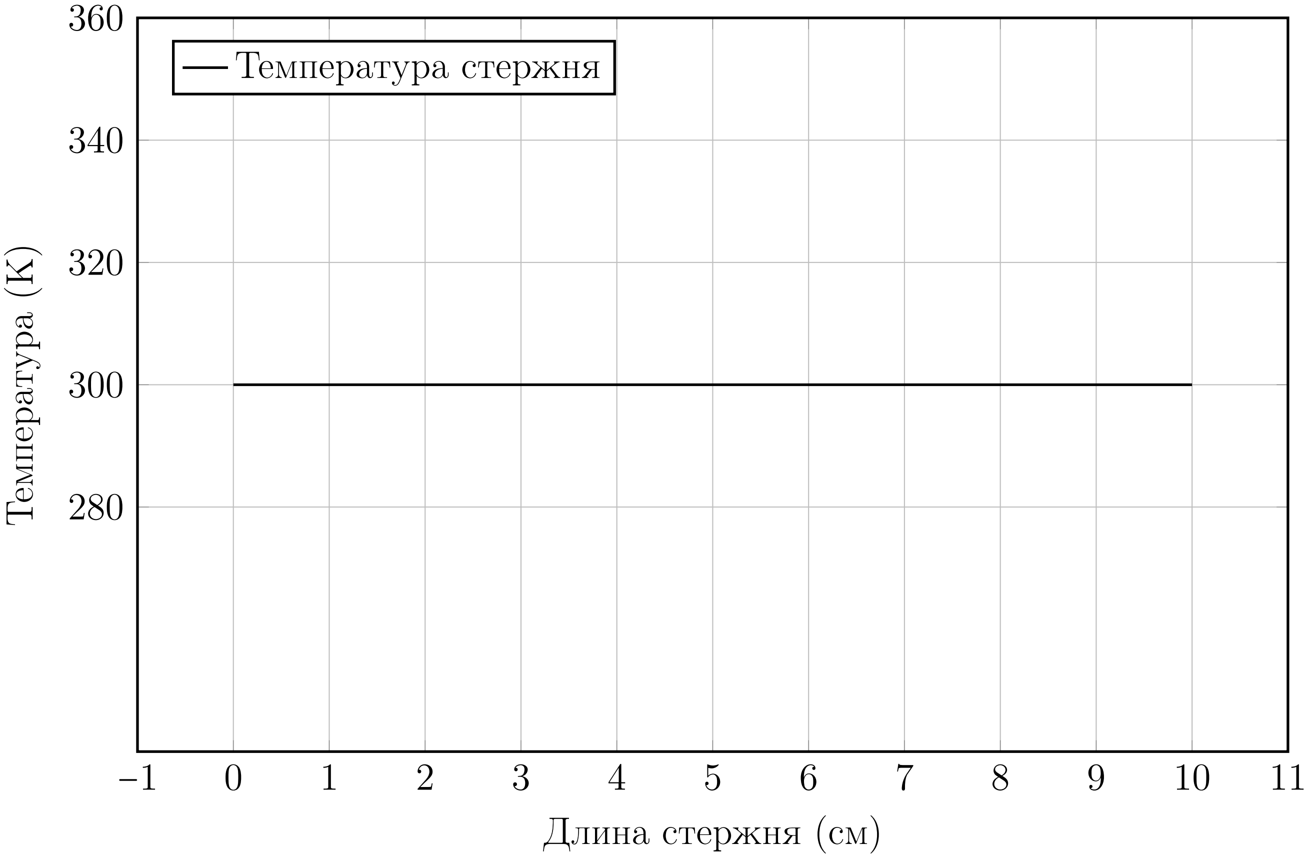

Рис. 6: $F_0 = 0$

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 $x = l$

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

, 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 $0 \le n \le p+1$ будет использоваться правая прогонка, а на промежутке $p \le n \le N$ – левая.

Тогда коэффициенты для правой прогонки будут

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

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

$$0 \le n \le p+1$$

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

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

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

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