Решение диференциального уравнения второго порядка методом разностной прогонки (методом сеток).

Постановка задачи: требуется численно решить линейное неоднородное дифференциальное уравнение второго порядка

$$y''(x) + p(x)y'(x) + q(x)y(x) = f(x) \tag{1}$$

с граничными условиями III типа:

$$y'(a) = \alpha y(a) + A; y'(b) = \beta y(b) + B \tag{2}$$

Используемые формулы:

- Приближенное вычисление производных:

$$y'(a) = \frac{y(a+h) - y(a)}{h} + R \tag{3}$$

$$y'(a) = \frac{y(a) - y(a-h)}{h} + R \tag{4}$$

$$y'(a) = \frac{y(a+h) - y(a-h)}{2h} + R \tag{5}$$

$$y''(a) = \frac{y(a+h) - 2y(a) + y(a-h)}{h^2} + R \tag{6}$$

- Сетка интергирования имеет вид: $x_k = a + h * k$, $k = 0, \dots, n$; $h = \frac{b-a}{n}$, где n - число узлов интегрирования, а h - шаг сетки.
- Численное решение будем искать в виде: $y_k \approx y(x_k)$
- Система уравнений будет иметь вид:

$$\begin{cases} \quad\quad\quad - b_0 y_0 + c_0 y_1 = d_0 \\ a_k y_{k-1} - b_k y_k + c_k y_{k-1} = d_k \\ a_n y_{n-1} - b_n y_n = d_n \end{cases} \tag{7}$$

Где:

$$\begin{array}{llll} \mathbf{b_0} = \frac{1}{h} + \frac{p_0}{2} - \frac{h}{2}q_0 + \alpha & \mathbf{c_0} = \frac{1}{h} + \frac{p_0}{2} & \mathbf{d_0} = A + \frac{h}{2}f_0 & \\ \mathbf{a_k} = \frac{1}{h^2} - \frac{p_k}{2h} & \mathbf{b_k} = \frac{2}{h} - q_k & \mathbf{c_k} = \frac{1}{h} + \frac{p_k}{2h} & \mathbf{d_k} = f_k \\ \mathbf{a_n} = \frac{p_n}{2} - \frac{1}{h} & \mathbf{b_n} = \frac{p_n}{2} - \frac{1}{h} + \frac{h}{2}q_n + \beta & \mathbf{d_n} = B - \frac{h}{2}f_n & \end{array} \tag{8}$$

- Для решения системы уравнений использовались формулы:

$$y_{k-1} = \alpha_{k-1} y_k + \beta_{k-1} \tag{9}$$

$$\alpha_k = \frac{c_k}{b_k - a_k \alpha_{k-1}} \tag{10}$$

$$\beta_k = \frac{a_k \beta_{k-1} - d_k}{b_k - a_k \alpha_{k-1}} \tag{11}$$

Написанная программа (Python):

Главная программа:

```
import math
import sys
import three_diag as td

aa = float(sys.argv[1])
bb = float(sys.argv[2])
n = int(sys.argv[3])
output_f_name = sys.argv[4]
out_file = open(output_f_name, 'w')

def p(x):
    return x + 1 / (1 + x)
def q(x):
    return math.sqrt(1 + x**2)
def f(x):
    return 1 - x**2

alpha = 0
A = 0
beta = -2
B = 1
h = (bb - aa) / n
a = [] ; b = [] ; c = [] ; d = []
a.append(0)
b.append(1/h + p(aa)/2 - q(aa)*h/2 + alpha)
c.append(1 / h + p(aa) / 2)
d.append(A + f(aa) * h / 2)

for k in range(1, n):
    x_k = aa + h * k
    a.append(1 / h**2 - p(x_k) / (2 * h))
    b.append(2 / h**2 - q(x_k))
    c.append(1 / h**2 + p(x_k) / (2 * h))
    d.append(f(x_k))

a.append(p(bb) / 2 - 1 / h)
b.append(p(bb)/2 - 1/h + q(bb)*h/2 + beta)
c.append(0)
d.append(B - f(bb) * h / 2)

y, alphak, betak = td.solution(a, b, c, d)
```

Модуль three_diag, решающий систему уравнений:

```
def solution(a, b, c, d):
    n = len(b) - 1
    y = [0]
    alpha = []
    beta = []
    alpha.append(c[0] / b[0])
    beta.append(-d[0] / b[0])
    for k in range(1, n + 1):
        alpha.append(c[k] / (b[k] - a[k] * alpha[k-1]))
        beta.append((a[k] * beta[k-1] - d[k]) / (b[k] - a[k] * alpha[k-1]))
        y.append(0)

    y[n] = beta[n]
    for k in range(n-1, -1, -1):
        y[k] = alpha[k] * y[k+1] + beta[k]

    return y, alpha, beta
```

Результаты работы программы для сетки с $n = 10, 20, 40$ узлами. Указаны только точки, соответствующие друг другу.

	n = 10	n = 20	n = 40
y_0	0.360613	0.358790	0.358334
y_1	0.363658	0.361870	0.361423
y_2	0.372355	0.370593	0.370152
y_3	0.385716	0.383967	0.383529
y_4	0.402536	0.400789	0.400352
y_5	0.421431	0.419676	0.419237
y_6	0.440877	0.439107	0.438665
y_7	0.459265	0.457475	0.457028
y_8	0.474952	0.473142	0.472690
y_9	0.486322	0.484497	0.484041
y_{10}	0.491845	0.490013	0.489556