Министерство науки и высшего образования Российской Федерации

Российский химико-технологический университет имени Д. И. Менделеева

Факультет цифровых технологий и химического инжиниринга

Кафедра информационных компьютерных технологий

**ОТЧЕТ**

**«**Расчёты дифференциальных уравнений различных типов с помощью численных методов**»**

**Вариант № 6**

**ВЫПОЛНИЛ:**

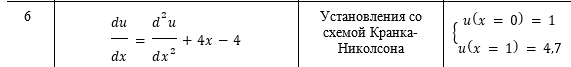
**ПРОВЕРИЛ:** Зинченко.Д.И.

**Москва**

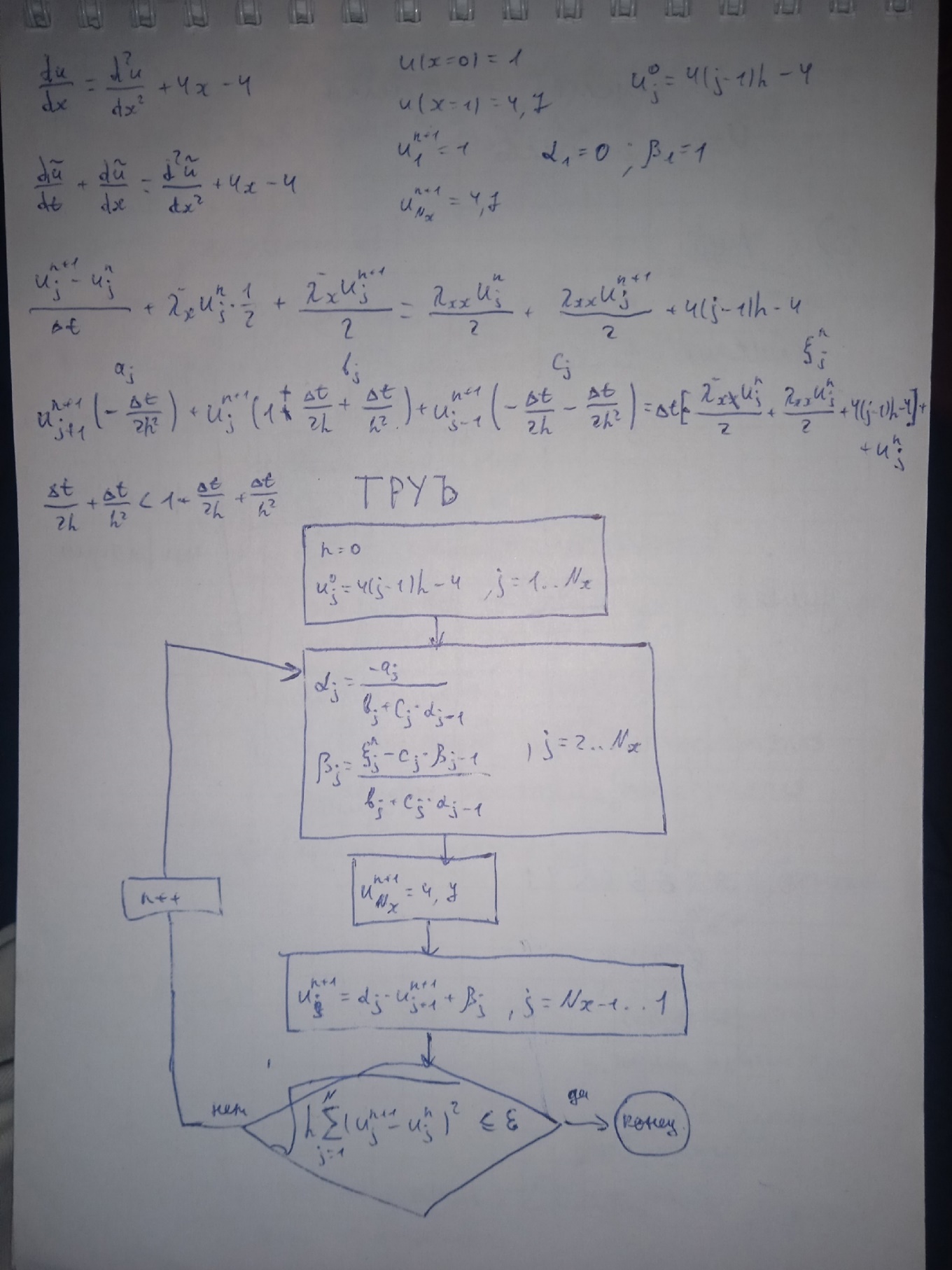
**2024**

**Лабораторная работа 4**

Для заданного уравнения:



1. Решение



1. Код

#include <iostream>

#include <vector>

#include <cmath>

using namespace std;

double my\_left = 0;

double my\_right = 1;

double eps = 0.0001;

double zero\_edge(double h, int counter){

return (4 \* h \* counter - 4);

}

bool quadratic(vector<double> new\_vec, vector<double> old, double h){

double sum = 0;

for (int i = 0; i < new\_vec.size(); i++){

sum += pow((new\_vec[i] - old[i]), 2);

}

return !(sqrt(sum \* h) <= eps);

}

vector<double> algo(double step\_x, double step\_t){

int n\_x = (my\_right - my\_left) / step\_x + 1;

int n\_t = (my\_right - my\_left) / step\_t + 1;

double a = -step\_t / (2 \* step\_x \* step\_x);

double c = -step\_t / (2 \* step\_x) - step\_t / (2 \* step\_x \* step\_x);

double b = 1 + step\_t / (2 \* step\_x) + step\_t / (step\_x \* step\_x);

vector<double> old\_answer(n\_x);

vector<double> new\_answer(n\_x);

vector<double> alphas(n\_x);

vector<double> betas(n\_x);

for (int i = 0; i < n\_x; i++){

new\_answer[i] = zero\_edge(step\_x, i);

}

do {

old\_answer = new\_answer;

alphas[0] = 0;

betas[0] = 1;

for (int j = 1; j < n\_x - 1; j++){

alphas[j] = (-a)/ (b + c \* alphas[j - 1]);

betas[j] = (step\_t \* (-((old\_answer[j] - old\_answer[j - 1])/(2 \* step\_x))

+ ((old\_answer[j + 1] - 2 \* old\_answer[j] + old\_answer[j - 1])/(2 \* step\_x \* step\_x)) +

4 \* j \* step\_x - 4) + old\_answer[j] - c \* betas[j - 1])

/ (b + c \* alphas[j - 1]);

}

new\_answer[n\_x - 1] = 4.7;

for (int j = n\_x - 2; j >= 0; j--){

new\_answer[j] = (alphas[j] \* new\_answer[j + 1] + betas[j]);

}

} while (quadratic(new\_answer, old\_answer, step\_x));

cout.fill(' ');

cout.precision(3);

for (int i = 0; i < n\_x; i++){

cout << new\_answer[i] << fixed << " ";

}

cout << endl;

return new\_answer;

}

int main()

{

algo(0.1, 0.1);

cout << endl;

algo(0.1, 0.01);

cout << endl;

algo(0.1, 0.001);

cout << endl;

algo(0.01, 0.1);

cout << endl;

algo(0.01, 0.01);

cout << endl;

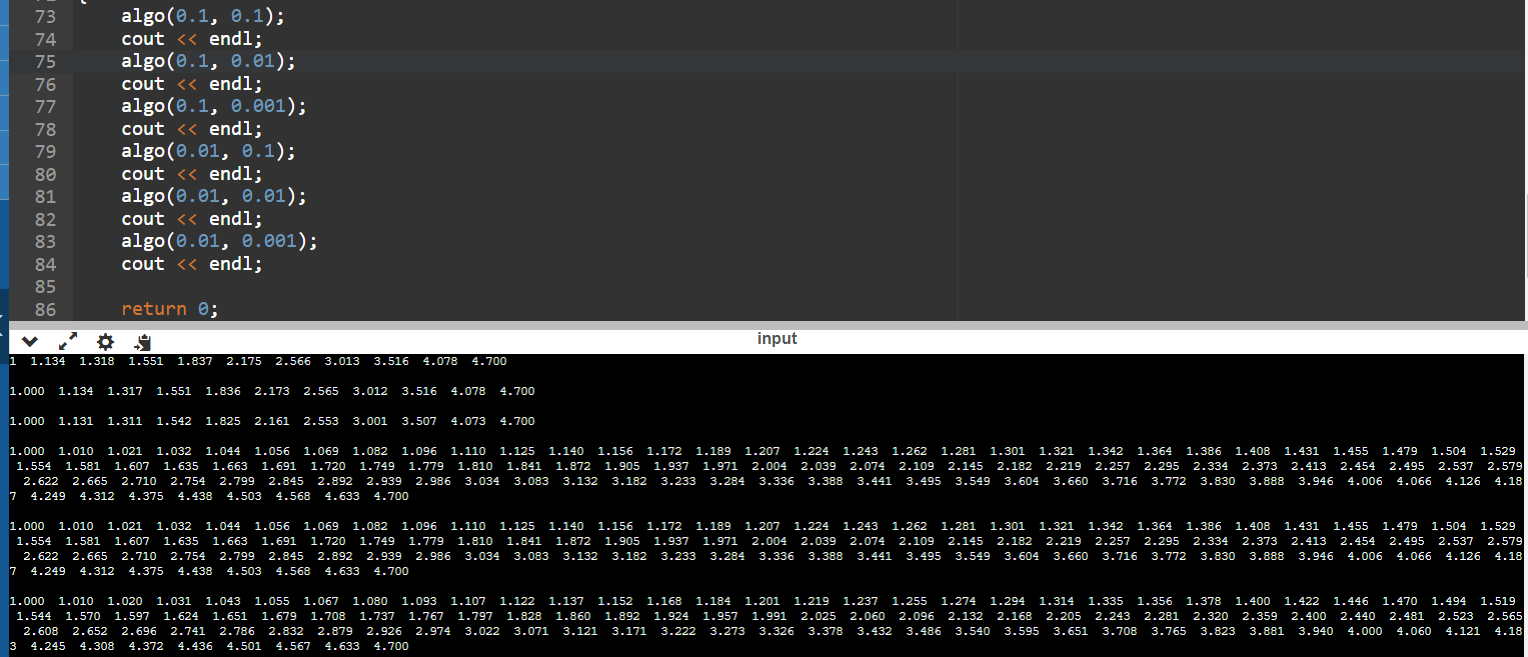
algo(0.01, 0.001);

cout << endl;

return 0;

}

Проведенные эксперименты для различных ∆t и h:



Вывод:

Схема Кранка-Николсона абсолютно устойчива, т.е. вне зависимости от выбора интервала деления на разностной сетке (или, иначе говоря, выбора расчетного шага по независимым переменным) погрешность решения схемы в процессе вычислений возрастать не будет. Благодаря условию с проверкой на точность, мы можем решить задачу за меньшее количество итераций.