МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Белгородский государственный технологический университет им. В.Г. Шухова»

**ИНСТИТУТ ЭНЕРГЕТИКИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И УПРАВЛЯЮЩИХ СИСТЕМ Кафедра программного обеспечения вычислительной техники и автоматизированных систем**

Лабораторная работа №4

по дисциплине: Вычислительная математика:

Численные методы решения задачи Коши

Работу выполнил:

Студент группы ПВ-21:

Донцов Александр Алексеевич

Проверил:

Бондаренко Татьяна Владимировна

Белгород – 2018

**Цель работы:** изучить численные методы решения задачи Коши; получить практические навыки приближенного решения дифференциальных уравнений с помощью ЭВМ.

**Задания к работе**

1. Вычислить «вручную» приближенное решение *y(x)* задачи Коши методом последовательного дифференцирования.

*Замечание*. Ряд Тейлора ограничить значением производной третьего порядка.

2. Вычислить значение функции *φ(х)*, которая является точным решением задачи Коши и функции *y(x)*, которая является приближенным решением задачи Коши по методу последовательного дифференцирования, в точке *x = b*.

*Замечание. x* = *b –* правый конец указанного в задании отрезка, которому принадлежит значение *х, a ≤ x ≤b.*

*x = b = x0 + ih, h>0 —* шаг сетки, *x0 = a*.

3. Определить относительную и абсолютную погрешности вычисления приближенного решения задачи Коши методом последовательного дифференцирования.

Значения погрешностей внести в соответствующие ячейки таблицы 4.

4. Вычислить «вручную» приближенное решение *y(x)* задачи Коши четырьмя численными методами решения:

- методом Эйлера;

- методом Эйлера-Коши;

- модифицированным методом Эйлера;

- методом Рунге-Кутты.

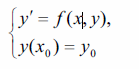
Сначала выполнить вычисления с шагом *h* = 0,2, а затем с шагом *h* = 0,1.

Вычисления вручную можно выполнить с помощью MS Excel или другой программы и *обязательно* их включать в отчет.

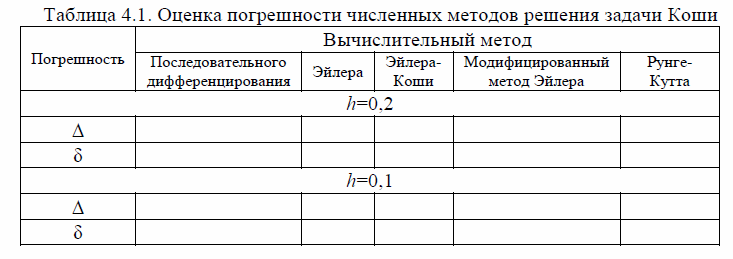
5. Сравнить полученные в пункте 4 значения приближенного решения дифференциального уравнения *y(x)* с точным значением решения дифференциального уравнения *φ(x)* в точке *x = b*.

6. Определить относительную и абсолютную погрешности вычисления приближенного решения задачи Коши заданными численными методами.

Значения погрешностей внести в соответствующие ячейки таблицы 4.1.

7. Описать в модуле функции, каждая из которы*х* возвращает приближенное значение решения задачи Коши:

в точке *x = b* с точностью , реализующие метод Эйлера, метод Эйлера-Коши, модифицированный метод Эйлера и метод Рунге-Кутты. Оценка точности вычисления должна осуществляться по принципу Рунге.

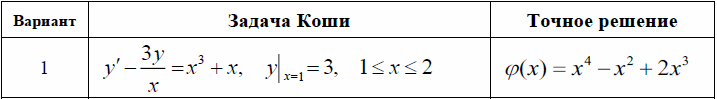


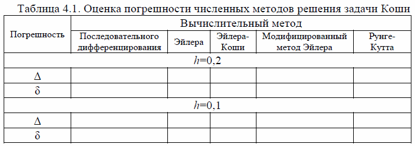
8. Составить программу для вычисления приближенны*х* значений решения задачи Коши с точностью  на отрезке [*a*, *b*] с шагом *h* для соответствующего варианта задания с использованием все*х* функций, описанны*х* в модуле.

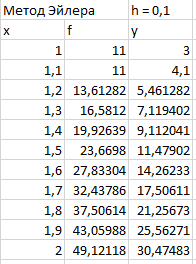
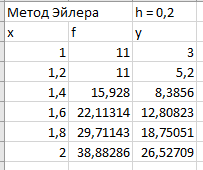
Результат работы программы таблица значений приближенного решения задачи Коши для заданного отрезка *a* ≤ *x* ≤ *b*.

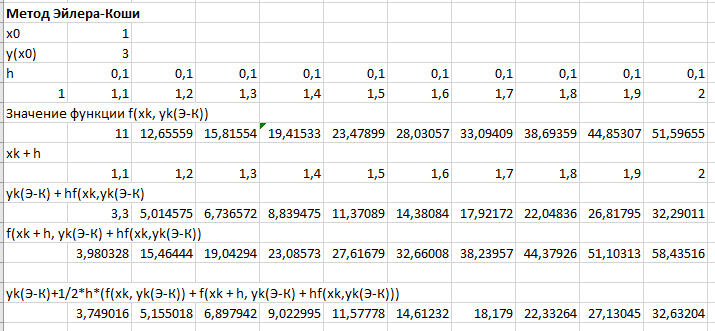
Предусмотреть возможность сохранения результата работы программы в файл.

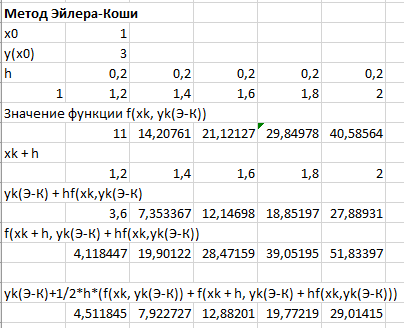
Отчет

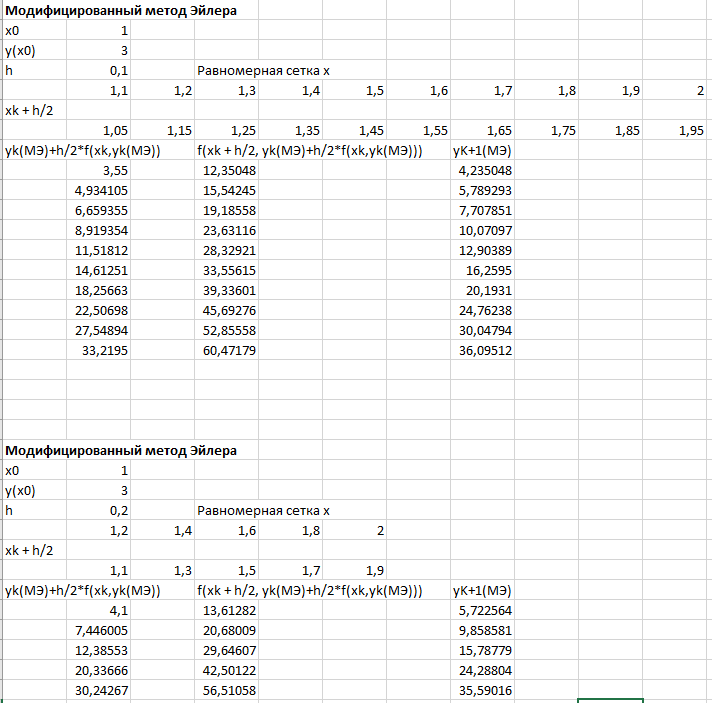


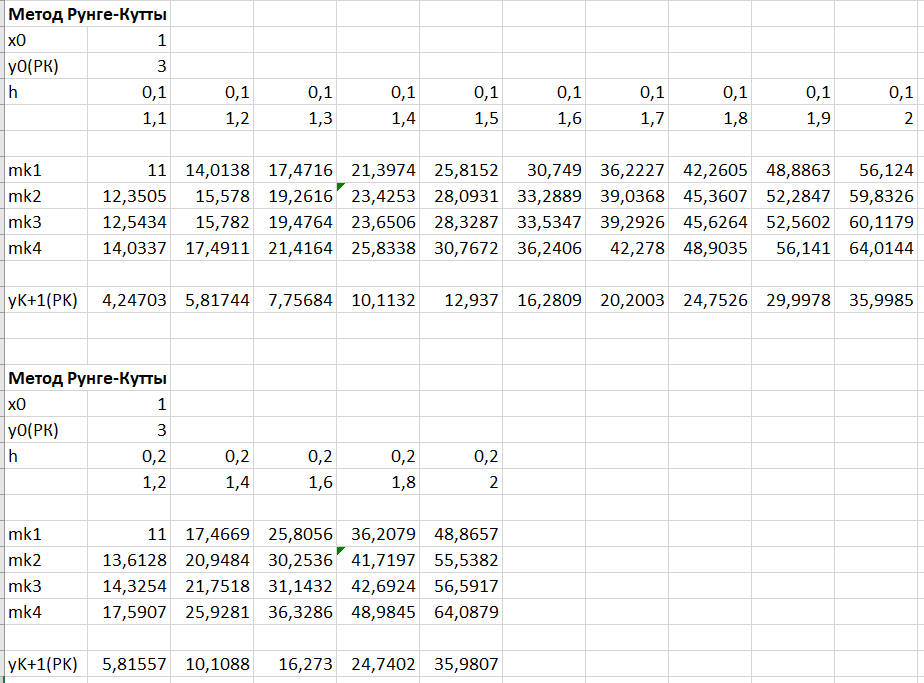












**Заголовочный файл mod\_solution\_koshi.h**

#pragma once

#ifndef MOD\_SOLUTION\_KOSHI\_H\_

#define MOD\_SOLUTION\_KOSHI\_H\_

#define x\_0 1

#define x\_n 2

#define y\_x0 3

double metod\_eilera(double, double);// Решение Задачи Коши методом Эйлера

double metod\_eilera\_koshi(double, double);// Решение Задачи Коши методом Эйлера-Коши

double metod\_eilera\_mod(double, double);// Решение Задачи Коши модифицированным методом Эйлера

double metod\_runge\_kutti(double, double);// Решение Задачи Коши методом Рунге-Кутты

double y\_sh(double, double); // y'

double toch\_znach(double); // Точное значение

#endif

**Исходный файл solution\_koshi.cpp**

#include "mod\_solution\_koshi.h";

#include "pch.h"

#include "stdio.h"

#include "math.h"

#define x\_0 1

#define x\_n 2

#define y\_x0 3

double toch\_znach(double x) {

double znach = x \* (2 / PI - cos(x));

return znach;

}

// y'

double y\_sh(double x, double y) {

double y\_shtrih = x \* sin(x) + (y / x);

return y\_shtrih;

}

//Решение задачи Коши методом Эйлера

double metod\_eilera(double h, double exp) {

double y\_pred;// у\_i-1

double y = y\_x0;//y\_i

double x\_pred = x\_0, x = x\_pred + h;//Предыдущий и настоящий x

while (x <= (x\_n)) {

y\_pred = y;

y = y\_pred + h \* (y\_sh(x\_pred, y\_pred));

x\_pred = x;

x = x + h;

}

if ((toch\_znach(x\_n) - y) > exp) {

double h\_dop = h / 2;

y = metod\_eilera(h\_dop, exp);

}

return y;

}

// Решение Задачи Коши методом Эйлера-Коши

double metod\_eilera\_koshi(double h, double exp) {

double y\_pred;// у\_i

double y = y\_x0;//y\_i

double x\_pred = x\_0, x = x\_pred + h;//Предыдущий и настоящий x

while (x <= (x\_n)) {

y\_pred = y;

y = y\_pred + (h / 2) \* (y\_sh(x\_pred, y\_pred) + y\_sh((x\_pred + h), (y\_pred + h \* y\_sh(x\_pred, y\_pred))));

x\_pred = x;

x = x + h;

}

if ((toch\_znach(x\_n) - y) > exp) {

double h\_dop = h / 2;

y = metod\_eilera(h\_dop, exp);

}

return y;

}

//Модифицированный метод Эйлера

double metod\_eilera\_mod(double h, double exp) {

double y\_pred;// у\_i

double y = y\_x0;//y\_i

double x\_pred = x\_0, x = x\_pred + h;//Предыдущий и настоящий x

while (x <= (x\_n)) {

y\_pred = y;

y = y\_pred + h \* y\_sh((x\_pred + h / 2), (y\_pred + (h / 2) \* y\_sh(x\_pred, y\_pred)));

x\_pred = x;

x = x + h;

}

if ((toch\_znach(x\_n) - y) > exp) {

double h\_dop = h / 2;

y = metod\_eilera(h\_dop, exp);

}

return y;

}

//Метод Рунге Кутты

double metod\_runge\_kutti(double h, double exp) {

double y\_pred;// у\_i

double y = y\_x0;//y\_i

double x\_pred = x\_0, x = x\_pred + h;//Предыдущий и настоящий x

double k\_1, k\_2, k\_3, k\_4;

while (x <= (x\_n)) {

y\_pred = y;

k\_1 = y\_sh(x\_pred, y\_pred);

k\_2 = y\_sh((x\_pred + h / 2), (y\_pred + h / 2 \* k\_1));

k\_3 = y\_sh((x\_pred + h / 2), (y\_pred + h / 2 \* k\_2));

k\_4 = y\_sh(x, (y\_pred + h \* k\_3));

y = y\_pred + h / 6 \* (k\_1 + 2 \* k\_2 + 2 \* k\_3 + k\_4);

x\_pred = x;

x = x + h;

}

if ((toch\_znach(x\_n) - y) > exp) {

double h\_dop = h / 2;

y = metod\_eilera(h\_dop, exp);

}

return y;

}

**Solution\_koshi**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "pch.h"

#include <iostream>

#include "windows.h"

#include "stdio.h"

#include "mod\_solution\_koshi.h"

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

double exp;// Точность решения

double h; // Первоначальный шаг

printf("Введите требуюмую точность решения E: ");

scanf("%lf", &exp);

printf("Введите начальный шаг h: ");

scanf("%lf", &h);

printf("\ny(0): %lf", toch\_znach(x\_0));

printf("\nТочное решение: %lf\n", toch\_znach(x\_n));

printf("\nРешение задачи методом Эйлера: %lf \n", metod\_eilera(h, exp));

printf("\nРешение задачи методом Эйлера-Коши: %lf \n", metod\_eilera\_koshi(h, exp));

printf("\nРешение задачи модифицированным методом Эйлера: %lf \n", metod\_eilera\_mod(h, exp));

printf("\nРешение задачи методом Рунге Кутты: %lf \n", metod\_runge\_kutti(h, exp));

int flag;

printf("\nЗаписать результат работы программы в файл? 1 - да, 0 - нет: ");

scanf("%d", &flag);

if (flag) {

FILE \*f;

if ((f = fopen("koshi.txt", "w")) != NULL) {

fprintf(f, "\nМетод Эйлера: %lf \n", metod\_eilera(h, exp));

fprintf(f, "\nМетод Эйлера-Коши: %lf \n", metod\_eilera\_koshi(h, exp));

fprintf(f, "\nМодиф. метод Эйлера: %lf \n", metod\_eilera\_mod(h, exp));

fprintf(f, "\nМетод Рунге-Кутты: %lf \n", metod\_runge\_kutti(h, exp));

printf("\nЗапись результатов в файл прошла успешно!\n");

}

}

getchar();

return 0;

}

