**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕНЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.Шухова»**

**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Дисциплина: Вычислительная математика

Лабораторная работа № 4

Тема: «Численные методы решения задачи Коши»

Выполнила:

Студент группы ВТ-21

Сидорова Ангелина Сергеевна

Проверила: Бондаренко Т. В.

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 которых возвращает приближенное значение решения задачи Коши:   
   в точке x = b с точностью ε, реализующие метод Эйлера, метод Эйлера-Коши, модифицированный метод Эйлера и метод Рунге-Кутты. Оценка точности вычисления должна осуществляться по принципу Рунге.

Таблица 4.1. Оценка погрешности численных методов решения задачи Коши

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Погрешность | Вычислительный метод | | | | |
| Последовательного дифференцирования | Эйлера | Эйлера-Коши | Модифицированный метод Эйлера | Рунге-Кутта |
| h=0,2 | | | | | |
| Δ |  |  |  |  |  |
| δ |  |  |  |  |  |
| h=0,1 | | | | | |
| Δ |  |  |  |  |  |
| δ |  |  |  |  |  |

1. Составить программу для вычисления приближенных значений решения задачи Коши с точностью ε на отрезке [a, b] с шагом h для соответствующего варианта задания с использованием всех функций, описанных 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.

**Вариант 10**

**Безымянный.png**

Задание1

y(x) = y(0) + y`(0)x + x2  + x3

y`(x) = - y2

y`(0) = 0

y```(0) = 4

Подставляем значения в ряд Тейлора:

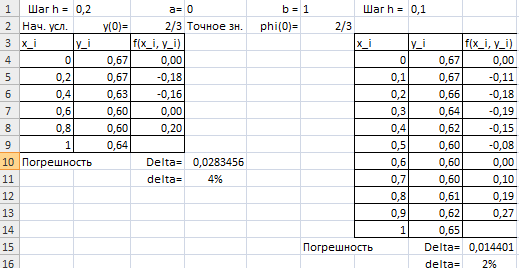
Задание 2

При b=1

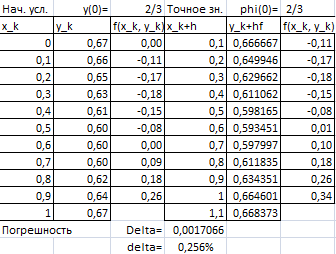
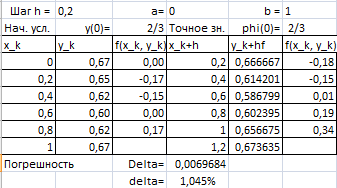
Задание 3-6

∆=δ=0

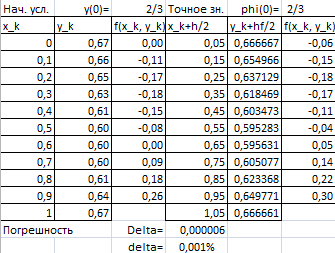
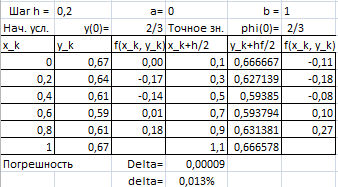
Метод Эйлера



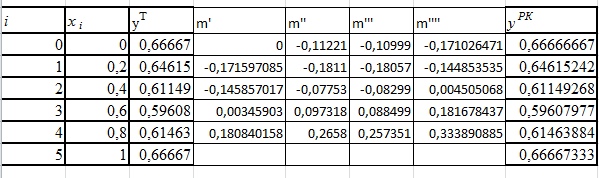
Метод Эйлера-Коши

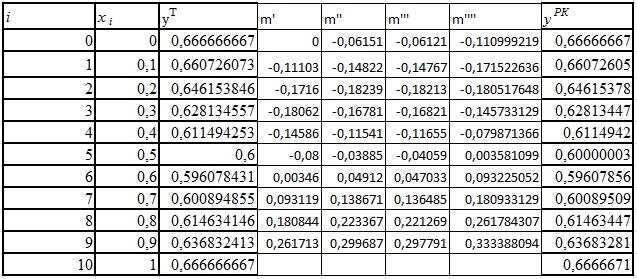


Модифицированный метод Эйлера



Метод Ренге-Кутте





|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Погрешность | Вычислительный метод | | | | |
| Последовательного дифференцирования | Эйлера | Эйлера-Коши | Модифицированный метод Эйлера | Рунге-Кутта |
| h=0,2 | | | | | |
| Δ | 0 | 0,0283 | 0,0069684 | 0,00009 | 0,013975 |
| δ | 0% | 4% | 1,045% | 0,013% | 2,096% |
| h=0,1 | | | | | |
| Δ | 0 | 0,0144 | 0,0017 | 0,000006 | 0,010181 |
| δ | 0% | 2% | 0,256% | 0,001% | 1,527% |

Текст программы

#ifndef Z\_KOSHI\_H

#define Z\_KOSHI\_H

typedef double (\* t\_func)(double x, double y);

void free\_array(double \*\*a, unsigned n);

double \*\*euler(double a, double b, unsigned n, double y\_0, t\_func f);

double \*\*euler\_koshi(double a, double b, unsigned n, double y\_0, t\_func f);

double \*\*modified\_euler(double a, double b, unsigned n, double y\_0, t\_func f);

double \*\*runge\_kutta(double a, double b, unsigned n, double y\_0, t\_func f);

double \*\*runge(double \*\*(\*koshi\_func)(double, double, unsigned, double, t\_func), double a, double b, unsigned \*n, double y\_0, t\_func f, double eps, int accuracy);

#endif /\* Z\_KOSHI\_H \*/

#include <stdlib.h>

#include <math.h>

#include "z\_koshi.h"

/\*

Освобождает память, выделенную под массив

\*/

void free\_array(double \*\*a, unsigned n){

for (unsigned i = 0; i<n; i++){

free(a[i]);

}

free(a);

}

/\*

Возвращает решение задачи Коши, найденное с точностью eps, оценка погрешности выполняется по методу Рунге

\*/

double \*\*runge(double \*\*(\*koshi\_func)(double, double, unsigned, double, t\_func), double a, double b, unsigned \*n, double y\_0, t\_func f, double eps, int accuracy){

double \*\*result1, \*\*result2;

double delta;

result1 = koshi\_func(a, b, \*n, y\_0, f);

(\*n)\*=2;

result2 = koshi\_func(a, b, \*n, y\_0, f);

delta = fabs(result2[\*n][1]-result1[\*n/2][1])/((1<<accuracy)-1);

while (delta > eps){

free\_array(result1, \*n/2+1);

result1=result2;

(\*n)\*=2;

result2 = koshi\_func(a, b, \*n, y\_0, f);

delta = abs(result2[\*n][1]-result1[\*n/2][1])/((1<<accuracy)-1);

}

free\_array(result1, \*n/2+1);

return result2;

}

/\*

Метод Эйлера

\*/

double \*\*euler(double a, double b, unsigned n, double y\_0, t\_func f){

double x = a;

double y = y\_0;

double h = (b-a)/n;

double \*\*result = (double \*\*)malloc(sizeof(double \*)\*(n+1));

for (unsigned i = 0; i<=n; i++){

result[i] = (double \*)malloc(sizeof(double)\*2);

}

result[0][0] = x;

result[0][1] = y;

for (unsigned i = 0; i < n; i++){

y = y+h\*f(x, y);

x+=h;

result[i+1][0] = x;

result[i+1][1] = y;

}

return result;

}

/\*

Метод Эйлера-Коши

\*/

double \*\*euler\_koshi(double a, double b, unsigned n, double y\_0, t\_func f){

double x = a;

double y = y\_0;

double h = (b-a)/n;

double \*\*result = (double \*\*)malloc(sizeof(double \*)\*(n+1));

for (unsigned i = 0; i<=n; i++){

result[i] = (double \*)malloc(sizeof(double)\*2);

}

result[0][0] = x;

result[0][1] = y;

for (unsigned k = 0; k < n; k++){

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

x+=h;

result[k+1][0] = x;

result[k+1][1] = y;

}

return result;

}

/\*

Модифицированный метод Эйлера

\*/

double \*\*modified\_euler(double a, double b, unsigned n, double y\_0, t\_func f){

double x = a;

double y = y\_0;

double h = (b-a)/n;

double \*\*result = (double \*\*)malloc(sizeof(double \*)\*(n+1));

for (unsigned i = 0; i<=n; i++){

result[i] = (double \*)malloc(sizeof(double)\*2);

}

result[0][0] = x;

result[0][1] = y;

for (unsigned k = 0; k < n; k++){

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

result[k+1][0] = x;

result[k+1][1] = y;

x+=h;

}

return result;

}

/\*

Метод Рунге-Кутты

\*/

double \*\*runge\_kutta(double a, double b, unsigned n, double y\_0, t\_func f){

double x = a;

double y = y\_0;

double h = (b-a)/n;

double m1, m2, m3, m4;

double \*\*result = (double \*\*)malloc(sizeof(double \*)\*(n+1));

for (unsigned i = 0; i<=n; i++){

result[i] = (double \*)malloc(sizeof(double)\*2);

}

result[0][0] = x;

result[0][1] = y;

for (unsigned k = 0; k < n; k++){

m1 = f(x, y);

m2 = f(x+h/2, y+h\*m1/2);

m3 = f(x+h/2, y+h\*m2/2);

m4 = f(x+h, y+h\*m3);

y = y+h/6\*(m1+2\*m2+2\*m3+m4);

x+=h;

result[k+1][0] = x;

result[k+1][1] = y;

}

return result;

}