**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа природных ресурсов

Направление подготовки Химическая технология

Отделение химической инженерии

**ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ СИСТЕМ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ**

**Лабораторная работа по дисциплине «Углубленный курс информатики»**

Выполнил студент гр. 2Д91 Г. Н. Омельченко

(Подпись)

\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ 2020 г.

Отчет принят:

Преподаватель

доцент ОХИ ИШПР, к.т.н. В.А. Чузлов

(Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Томск 2020 г.

**Цель работы:** научиться составлять программы для численных методов решения систем обыкновенных дифференциальных уравнений.

**Теоретическая часть**

**Метод Эйлера:**

Пусть дано дифференциальное уравнение:

с начальными условиями:



Формула Эйлера:

где ***h*** – шаг вычисления;  ***f(xi, yi)*** – правая часть дифференциального уравнения.

**Метод Рунге-Кутты:**

где ***h*** – шаг вычисления;  ***f(xi, yi)*** – правая часть дифференциального уравнения.

Решить систему дифференциальных уравнений первого порядка:



методами Эйлера и Рунге-Кутты на отрезке [0; 1] с шагом

***h*** = 0,1. Начальные условия: *y1(0) = 0; y2(0) = 0*.

**Program** lab\_13\_example;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := y2;

result[1] := exp(-x \* y1)

**end**;

**function** eyler\_method(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**procedure** print\_results(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

print\_results(eyler\_method(0, 1, 0.1, 0, 0))

**end**.

**Ответ:**

0.0000 0.0000 0.0000

0.1000 0.0000 0.1000

0.2000 0.0100 0.2000

0.3000 0.0300 0.2998

0.4000 0.0600 0.3989

0.5000 0.0999 0.4965

0.6000 0.1495 0.5917

0.7000 0.2087 0.6831

0.8000 0.2770 0.7695

0.9000 0.3539 0.8496

1.0000 0.4389 0.9223

**Метод Рунге-Кутты:**

**Program** lab\_13\_example;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := y2;

result[1] := exp(-x \* y1)

**end**;

**function** runge\_kutt\_method(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**procedure** print\_results(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

print\_results(runge\_kutt\_method(0, 1, 0.1, 0, 0))

**end**.

**Ответ:**

0.0000 0.0000 0.0000

0.1000 0.0050 0.1000

0.2000 0.0200 0.1998

0.3000 0.0449 0.2990

0.4000 0.0797 0.3968

0.5000 0.1242 0.4924

0.6000 0.1781 0.5844

0.7000 0.2409 0.6717

0.8000 0.3122 0.7529

0.9000 0.3913 0.8271

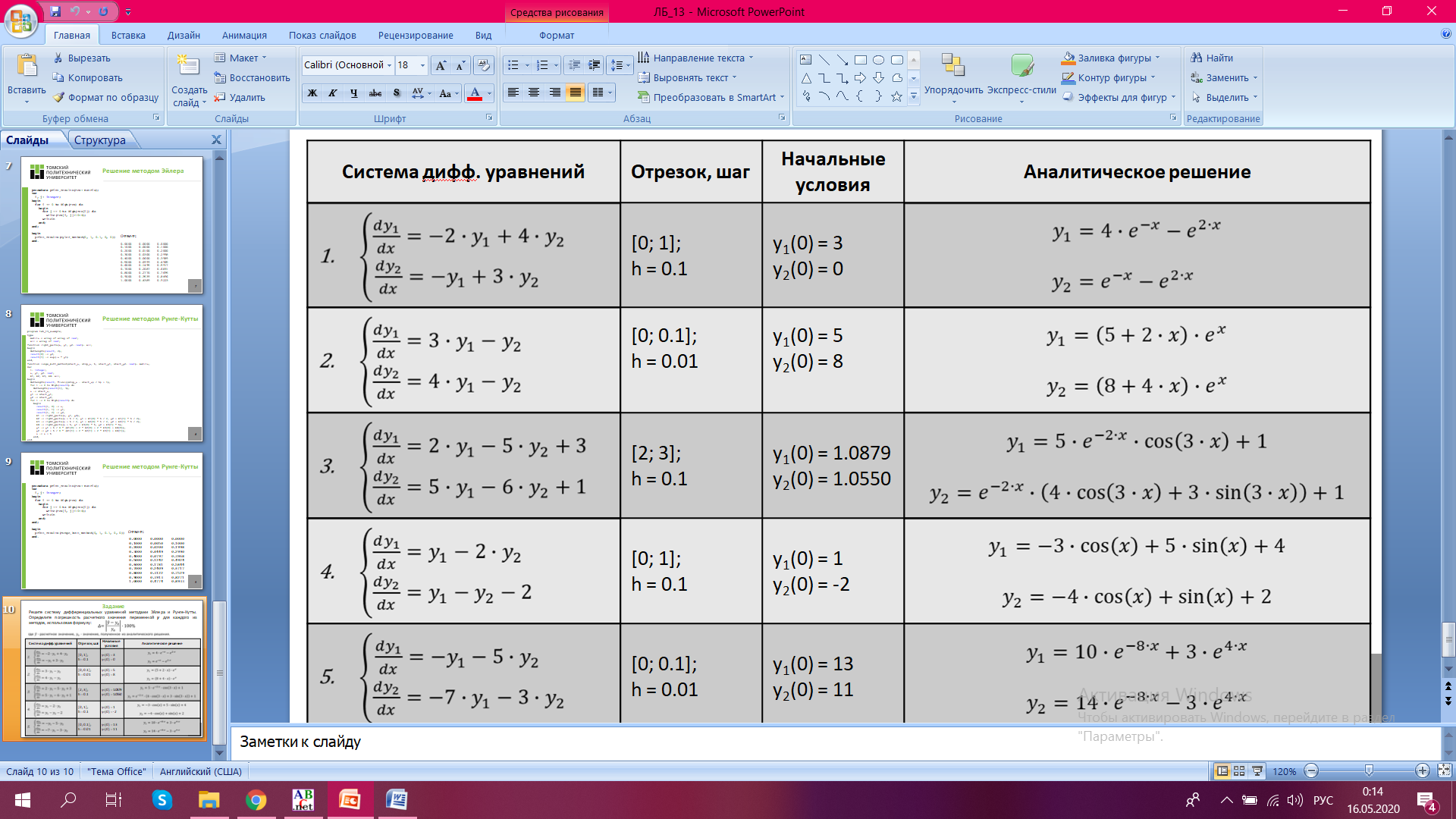
1.0000 0.4774 0.8933

**Практическая часть**

**Задание**

**Задание:** Решите систему дифференциальных уравнений методами Эйлера и Рунге-Кутты. Определите погрешность расчетного значения переменной ***y*** для каждого из методов, использовав формулу:

Где – расчётное значение, - значение, полученное из аналитического решения.

****

**Программная реализация:**

**Первая система:**

**1 способ:**

**Program** gtr;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := - 2 \* y1 + 4 \* y2;

result[1] := - y1 + 3 \* y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 4 \* exp(- x) - exp(2 \* x);

result[1] := exp(- x) - exp(2 \* x)

**end**;

**function** eiler(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(eiler(0, 1, 0.1, 3, 0)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.1000 2.4000 2.3979 0.0856 -0.3000 -0.3166 5.2328

0.2000 1.8000 1.7831 0.9479 -0.6300 -0.6731 6.4024

0.3000 1.1880 1.1412 4.1051 -0.9990 -1.0813 7.6113

0.4000 0.5508 0.4557 20.8586 -1.4175 -1.5552 8.8554

0.5000 -0.1264 -0.2922 56.7496 -1.8978 -2.1118 10.1300

0.6000 -0.8602 -1.1249 23.5272 -2.4545 -2.7713 11.4301

0.7000 -1.6700 -2.0689 19.2795 -3.1049 -3.5586 12.7502

0.8000 -2.5779 -3.1557 18.3086 -3.8693 -4.5037 14.0852

0.9000 -3.6101 -4.4234 18.3858 -4.7724 -5.6431 15.4298

1.0000 -4.7970 -5.9175 18.9355 -5.8431 -7.0212 16.7795

**2 способ:**

**Program** wfv;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := - 2 \* y1 + 4 \* y2;

result[1] := - y1 + 3 \* y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 4 \* exp(- x) - exp(2 \* x);

result[1] := exp(- x) - exp(2 \* x)

**end**;

**function** kutt(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(kutt(0, 1, 0.1, 3, 0)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.1000 2.3980 2.3979 0.0001 -0.3166 -0.3166 0.0009

0.2000 1.7831 1.7831 0.0004 -0.6731 -0.6731 0.0010

0.3000 1.1412 1.1412 0.0012 -1.0813 -1.0813 0.0012

0.4000 0.4558 0.4557 0.0046 -1.5552 -1.5552 0.0013

0.5000 -0.2921 -0.2922 0.0109 -2.1117 -2.1118 0.0015

0.6000 -1.1248 -1.1249 0.0041 -2.7713 -2.7713 0.0016

0.7000 -2.0688 -2.0689 0.0032 -3.5586 -3.5586 0.0018

0.8000 -3.1556 -3.1557 0.0029 -4.5036 -4.5037 0.0020

0.9000 -4.4232 -4.4234 0.0028 -5.6430 -5.6431 0.0022

1.0000 -5.9174 -5.9175 0.0028 -7.0210 -7.0212 0.0024

**Вторая система:**

**1 способ:**

**Program** bdfbed;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := 3 \* y1 - y2;

result[1] := 4 \* y1 - y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := (5 + 2 \* x) \* exp(x);

result[1] := (8 + 4 \* x) \* exp(x)

**end**;

**function** eiler(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(eiler(0, 0.1, 0.01, 5, 8)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0100 5.0700 5.0705 0.0089 8.1200 8.1208 0.0099

0.0200 5.1409 5.1418 0.0178 8.2416 8.2432 0.0197

0.0300 5.2127 5.2141 0.0266 8.3648 8.3673 0.0295

0.0400 5.2854 5.2873 0.0355 8.4897 8.4930 0.0393

0.0500 5.3591 5.3615 0.0442 8.6162 8.6204 0.0490

0.0600 5.4337 5.4366 0.0530 8.7444 8.7495 0.0586

0.0700 5.5093 5.5127 0.0617 8.8743 8.8804 0.0682

0.0800 5.5858 5.5898 0.0704 9.0059 9.0129 0.0778

0.0900 5.6633 5.6678 0.0791 9.1393 9.1473 0.0873

0.1000 5.7418 5.7469 0.0877 9.2745 9.2834 0.0968

**2 способ:**

**Program** sfbvcxb;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := 3 \* y1 - y2;

result[1] := 4 \* y1 - y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := (5 + 2 \* x) \* exp(x);

result[1] := (8 + 4 \* x) \* exp(x)

**end**;

**function** kutt(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(kutt(0, 0.1, 0.01, 5, 8)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0100 5.0705 5.0705 0.0000 8.1208 8.1208 0.0000

0.0200 5.1418 5.1418 0.0000 8.2432 8.2432 0.0000

0.0300 5.2141 5.2141 0.0000 8.3673 8.3673 0.0000

0.0400 5.2873 5.2873 0.0000 8.4930 8.4930 0.0000

0.0500 5.3615 5.3615 0.0000 8.6204 8.6204 0.0000

0.0600 5.4366 5.4366 0.0000 8.7495 8.7495 0.0000

0.0700 5.5127 5.5127 0.0000 8.8804 8.8804 0.0000

0.0800 5.5898 5.5898 0.0000 9.0129 9.0129 0.0000

0.0900 5.6678 5.6678 0.0000 9.1473 9.1473 0.0000

0.1000 5.7469 5.7469 0.0000 9.2834 9.2834 0.0000

**Третья система:**

**1 способ:**

**Program** skvlnj;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := 2 \* y1 - 5 \* y2 + 3;

result[1] := 5 \* y1 - 6 \* y2 + 1

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 5 \* exp(- 2 \* x) \* cos(3 \* x) + 1;

result[1] := exp(- 2 \* x) \* (4 \* cos(3 \* x) + 3 \* sin(3 \* x)) + 1

**end**;

**function** eiler(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(eiler(2, 3, 0.1, 1.0879, 1.0550)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

2.1000 1.0780 1.0750 0.2803 1.0660 1.0607 0.4921

2.2000 1.0606 1.0583 0.2144 1.0654 1.0581 0.6833

2.3000 1.0400 1.0410 0.0924 1.0564 1.0502 0.5910

2.4000 1.0198 1.0250 0.5086 1.0426 1.0396 0.2863

2.5000 1.0025 1.0117 0.9088 1.0269 1.0283 0.1317

2.6000 0.9895 1.0015 1.1964 1.0120 1.0177 0.5595

2.7000 0.9814 0.9945 1.3175 0.9996 1.0087 0.9100

2.8000 0.9779 0.9904 1.2624 0.9905 1.0018 1.1256

2.9000 0.9782 0.9887 1.0574 0.9852 0.9970 1.1834

3.0000 0.9813 0.9887 0.7515 0.9832 0.9940 1.0926

**2 способ:**

**Program** sdkmv;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := 2 \* y1 - 5 \* y2 + 3;

result[1] := 5 \* y1 - 6 \* y2 + 1

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 5 \* exp(- 2 \* x) \* cos(3 \* x) + 1;

result[1] := exp(- 2 \* x) \* (4 \* cos(3 \* x) + 3 \* sin(3 \* x)) + 1

**end**;

**function** kutt(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(kutt(2, 3, 0.1, 1.0879, 1.0550)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

2.1000 1.0749 1.0750 0.0033 1.0607 1.0607 0.0004

2.2000 1.0583 1.0583 0.0035 1.0581 1.0581 0.0013

2.3000 1.0410 1.0410 0.0034 1.0502 1.0502 0.0019

2.4000 1.0250 1.0250 0.0032 1.0396 1.0396 0.0022

2.5000 1.0117 1.0117 0.0028 1.0283 1.0283 0.0023

2.6000 1.0015 1.0015 0.0023 1.0177 1.0177 0.0023

2.7000 0.9945 0.9945 0.0017 1.0087 1.0087 0.0020

2.8000 0.9904 0.9904 0.0012 1.0018 1.0018 0.0017

2.9000 0.9887 0.9887 0.0007 0.9969 0.9970 0.0014

3.0000 0.9887 0.9887 0.0003 0.9940 0.9940 0.0010

**Четвёртая система:**

**1 способ:**

**Program** sgc;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := y1 - 2 \* y2;

result[1] := y1 - y2 - 2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := - 3 \* cos(x) + 5 \* sin(x) + 4;

result[1] := - 4 \* cos(x) + sin(x) + 2

**end**;

**function** eiler(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(eiler(0, 1, 0.1, 1, -2)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.1000 1.5000 1.5142 0.9348 -1.9000 -1.8802 1.0540

0.2000 2.0300 2.0531 1.1274 -1.7600 -1.7216 2.2307

0.3000 2.5850 2.6116 1.0182 -1.5810 -1.5258 3.6160

0.4000 3.1597 3.1839 0.7603 -1.3644 -1.2948 5.3733

0.5000 3.7486 3.7644 0.4205 -1.1120 -1.0309 7.8654

0.6000 4.3458 4.3472 0.0323 -0.8259 -0.7367 12.1129

0.7000 4.9456 4.9266 0.3858 -0.5088 -0.4152 22.5487

0.8000 5.5419 5.4967 0.8227 -0.1633 -0.0695 135.1045

0.9000 6.1287 6.0518 1.2712 0.2072 0.2969 30.2118

1.0000 6.7002 6.5864 1.7266 0.5993 0.6803 11.8948

**2 способ:**

**Program** dsdf;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := y1 - 2 \* y2;

result[1] := y1 - y2 - 2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := - 3 \* cos(x) + 5 \* sin(x) + 4;

result[1] := - 4 \* cos(x) + sin(x) + 2

**end**;

**function** kutt(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(kutt(0, 1, 0.1, 1, -2)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.1000 1.5142 1.5142 0.0000 -1.8802 -1.8802 0.0000

0.2000 2.0531 2.0531 0.0000 -1.7216 -1.7216 0.0000

0.3000 2.6116 2.6116 0.0001 -1.5258 -1.5258 0.0000

0.4000 3.1839 3.1839 0.0001 -1.2948 -1.2948 0.0001

0.5000 3.7644 3.7644 0.0001 -1.0309 -1.0309 0.0001

0.6000 4.3472 4.3472 0.0001 -0.7367 -0.7367 0.0002

0.7000 4.9266 4.9266 0.0001 -0.4152 -0.4152 0.0004

0.8000 5.4967 5.4967 0.0001 -0.0695 -0.0695 0.0032

0.9000 6.0518 6.0518 0.0001 0.2969 0.2969 0.0009

1.0000 6.5864 6.5864 0.0001 0.6803 0.6803 0.0005

**Пятая система:**

**1 способ:**

**Program** dfsdf;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := - y1 - 5 \* y2;

result[1] := - 7 \* y1 - 3 \* y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 10 \* exp(- 8 \* x) + 3 \* exp(4 \* x);

result[1] := 14 \* exp(- 8 \* x) - 3 \* exp(4 \* x)

**end**;

**function** eiler(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

f: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

f := right\_parts(x, y1, y2);

y1 := y1 + h \* f[0];

y2 := y2 + h \* f[1];

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(eiler(0, 0.1, 0.01, 13, 11)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0100 12.3200 12.3536 0.2720 9.7600 9.8012 0.4203

0.0200 11.7088 11.7713 0.5309 8.6048 8.6802 0.8681

0.0300 11.1615 11.2488 0.7761 7.5270 7.6303 1.3533

0.0400 10.6735 10.7820 1.0065 6.5199 6.6456 1.8904

0.0500 10.2408 10.3674 1.2215 5.5772 5.7203 2.5014

0.0600 9.8595 10.0016 1.4205 4.6930 4.8492 3.2213

0.0700 9.5263 9.6815 1.6033 3.8621 4.0275 4.1087

0.0800 9.2379 9.4043 1.7695 3.0794 3.2507 5.2713

0.0900 8.9915 9.1675 1.9194 2.3403 2.5145 6.9285

0.1000 8.7846 8.9688 2.0532 1.6407 1.8151 9.6095

**2 способ:**

**Program** sdvsd;

**type**

matrix = **array of array of** real;

arr = **array of** real;

**function** right\_parts(x, y1, y2: real): arr;

**begin**

SetLength(result, 2);

result[0] := - y1 - 5 \* y2;

result[1] := - 7 \* y1 - 3 \* y2

**end**;

**function** f1(x: real): arr;

**begin**

SetLength(result, 2);

result[0] := 10 \* exp(- 8 \* x) + 3 \* exp(4 \* x);

result[1] := 14 \* exp(- 8 \* x) - 3 \* exp(4 \* x)

**end**;

**function** kutt(start\_x, stop\_x, h, start\_y1, start\_y2: real): matrix;

**var**

i: integer;

x, y1, y2: real;

k1, k2, k3, k4: arr;

**begin**

SetLength(result, Trunc((stop\_x - start\_x) / h) + 1);

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 3);

x := start\_x;

y1 := start\_y1;

y2 := start\_y2;

**for** i := 0 **to** High(result) **do**

**begin**

result[i, 0] := x;

result[i, 1] := y1;

result[i, 2] := y2;

k1 := right\_parts(x, y1, y2);

k2 := right\_parts(x + h / 2, y1 + k1[0] \* h / 2, y2 + k1[1] \* h / 2);

k3 := right\_parts(x + h / 2, y1 + k2[0] \* h / 2, y2 + k2[1] \* h / 2);

k4 := right\_parts(x + h, y1 + k3[0] \* h, y2 + k3[1] \* h);

y1 := y1 + h / 6 \* (k1[0] + 2 \* k2[0] + 2 \* k3[0] + k4[0]);

y2 := y2 + h / 6 \* (k1[1] + 2 \* k2[1] + 2 \* k3[1] + k4[1]);

x := x + h

**end**;

**end**;

**function** err(res: matrix): matrix;

**var**

i: integer;

delta1, delta2: real;

ya: arr;

**begin**

SetLength(result, Length(res));

**for** i := 0 **to** High(result) **do**

SetLength(result[i], 7);

**for** i := 1 **to** High(result) **do**

**begin**

ya := f1(res[i, 0]);

delta1 := abs((res[i, 1] - ya[0]) / ya[0]) \* 100;

delta2 := abs((res[i, 2] - ya[1]) / ya[1]) \* 100;

result[i, 0] := res[i, 0];

result[i, 1] := res[i, 1];

result[i, 2] := ya[0];

result[i, 3] := delta1;

result[i, 4] := res[i, 2];

result[i, 5] := ya[1];

result[i, 6] := delta2

**end**;

**end**;

**procedure** resultat(res: matrix);

**var**

i, j: integer;

**begin**

**for** i := 0 **to** High(res) **do**

**begin**

**for** j := 0 **to** High(res[i]) **do**

write(res[i, j]:10:4);

writeln

**end**;

**end**;

**begin**

resultat(err(kutt(0, 0.1, 0.01, 13, 11)))

**end**.

**Ответ:**

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0100 12.3536 12.3536 0.0000 9.8012 9.8012 0.0000

0.0200 11.7713 11.7713 0.0000 8.6802 8.6802 0.0000

0.0300 11.2488 11.2488 0.0000 7.6303 7.6303 0.0000

0.0400 10.7820 10.7820 0.0000 6.6456 6.6456 0.0000

0.0500 10.3674 10.3674 0.0000 5.7203 5.7203 0.0000

0.0600 10.0016 10.0016 0.0000 4.8492 4.8492 0.0000

0.0700 9.6815 9.6815 0.0000 4.0275 4.0275 0.0000

0.0800 9.4043 9.4043 0.0000 3.2507 3.2507 0.0001

0.0900 9.1675 9.1675 0.0000 2.5145 2.5145 0.0001

0.1000 8.9688 8.9688 0.0000 1.8151 1.8151 0.0001

**Выводы**

В ходе работы были успешно составлены программы для численных методов решения систем обыкновенных дифференциальных уравнений.