НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

про виконання лабораторної роботи №3

по дисципліні «Алгоритми і системи комп’ютерної математики - 1 »

на тему

«Розв’язання звичайних диференціальних рівнянь»

Варіант 5

Виконав: Перевірив:

студент групи КМ-73 асистент кафедри ПМА

Звягін М.С. Дрозденко О.М.

Київ — 2020

Зміст

[1 Вступ](#_30j0zll) 3

[2 Вимоги до ПЗ](#_1fob9te) 4

[3 Тестові випадки](#_3znysh7) 5

[3.1 Програма на мові Python 6](#_2et92p0)

[A. Валідація вхідних даних 6](#_tyjcwt)

[3.2 Програма у середовищі Octave 7](#_3dy6vkm)

[A. Валідація вхідних даних 7](#_1t3h5sf)

[4 Опис методів](#_2s8eyo1) 8

[4.1 Метод Рунге-Кутта 2-го порядку 9](#_17dp8vu)

[4.2 Метод Рунге-Кутта 3-го порядку 9](#_3rdcrjn)

[4.3 Метод Рунге-Кутта 4-го порядку 9](#_26in1rg)

[5 Висновки 10](#_lnxbz9)

[Додаток А. Графи потоку керування для модулів програм 1](#_35nkun2)7

[Додаток Б. Лістинг програми.](#_1ksv4uv) 18

# 1 Вступ

Мета роботи: вивчення правил використання програмних засобів чисельного розв’язання звичайних диференціальних рівнянь, практичне розв’язання рівнянь на комп’ютері з використанням СКМ, порівняльний аналіз методів інтегрування ОДУ.

На мові Python та Octave був реалізований метод Рунге-Кутта четвертого, третього та другого порядку.

Графи потоку керування для кожного модуля надані у додатку А.

Код кожної програми наведено у додатку Б.

# 2 Вимоги до ПЗ

1. **Валідація вхідних даних**

А.1 При порожньому уведенні, програма повинна видати повідомлення про помилку

А.2 У кожному з полів допускаються лише числові значення (цілі та раціональні). Таким чином, при введені у будь-яке з полів значень відмінних від числових, користувач повинен отримати повідомлення про помилку.

А.3 Жодні спеціальні символи не допускаються, окрім «-» і «+» на початку для від’ємних чисел і додатних чисел, відповідно.

А.4 Максимальне можливе введене число 999999, а мінімальне -999999. В іншому випадку користувач отримує повідомлення про помилку.

А.5 Початок інтервалу не може бути більшим за кінець інтервалу

# 3 Тестові випадки

## 3.1 Програма на мові Python

### Валідація вхідних даних

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Сценарій | Кроки | | Вхідні дані | | Очікуваний результат | Експ. результат | Pass? |
| A1 | Перевірити введення порожніх полів | 1. Залишити всі поля порожніми  2. Залишити одне поле порожнім | 1.Усі порожні поля  2. Усі, окрім 1, заповнені поля | | | 1,2. Повідомлення про помилку. | 1,2. Повідомлення про помилку | Так |
| А2 | Перевірити обробку буквенно-числових рядків | 1. Увести рядок, що містить букви. | 1. «івфів»  2. «ы12вав» | | | 1,2. Повідомлення про помилку. | 1,2. Повідомлення про помилку. | Так |
| A3 | Увести спеціальні символи, включно з «-» і «+» | 1. Увести від’ємне число  2. Увести число зі спец. символом | 1. «-432»  2. «64&7»  3. «+324» | | | 1. Нормальна робота – розв’язок СЛАР  2. Повідомлення про помилку.  3. Нормальна робота – розв’язок СЛАР | 1. Нормальна робота – розв’язок СЛАР  2. Повідомлення про помилку.  3. Нормальна робота – розв’язок СЛАР | Так |
| А4 | Перевірити гранично допустимі числа | 1. Увести граничні значення | 1. «-100001»  2. «-100000»  3. «100000»  4. «1000001» | | | 1. Повідомлення про помилку  2. Нормальна робота  3. Нормальна робота  4. Повідомлення про помилку | 1. Повідомлення про помилку  2. Нормальна робота  3. Нормальна робота  4. Повідомлення про помилку | Так |
| A5 | Увести значення інтервалу | 1. Початок інтервалу 10. Кінець інтервалу -5  2. Початок інтервалу 5. Кінець інтервалу 10 | | | 1. 10  2. -5  3. 5  4. 10 | 1. Повідомлення про помилку  2. Нормальна робота | 1.Повідомлення про помилку  2. Нормальна робота | Так |

## 3.2 Програма у середовищі Octave

### Валідація вхідних даних

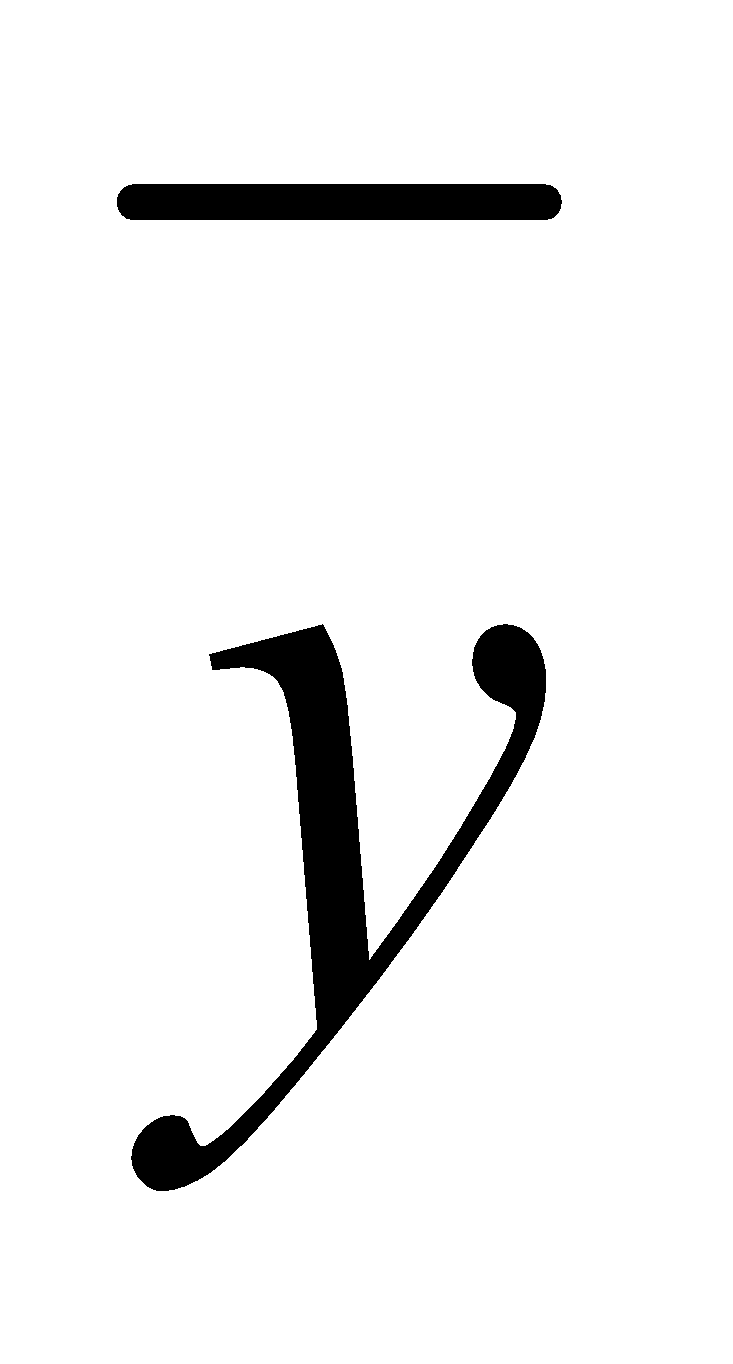
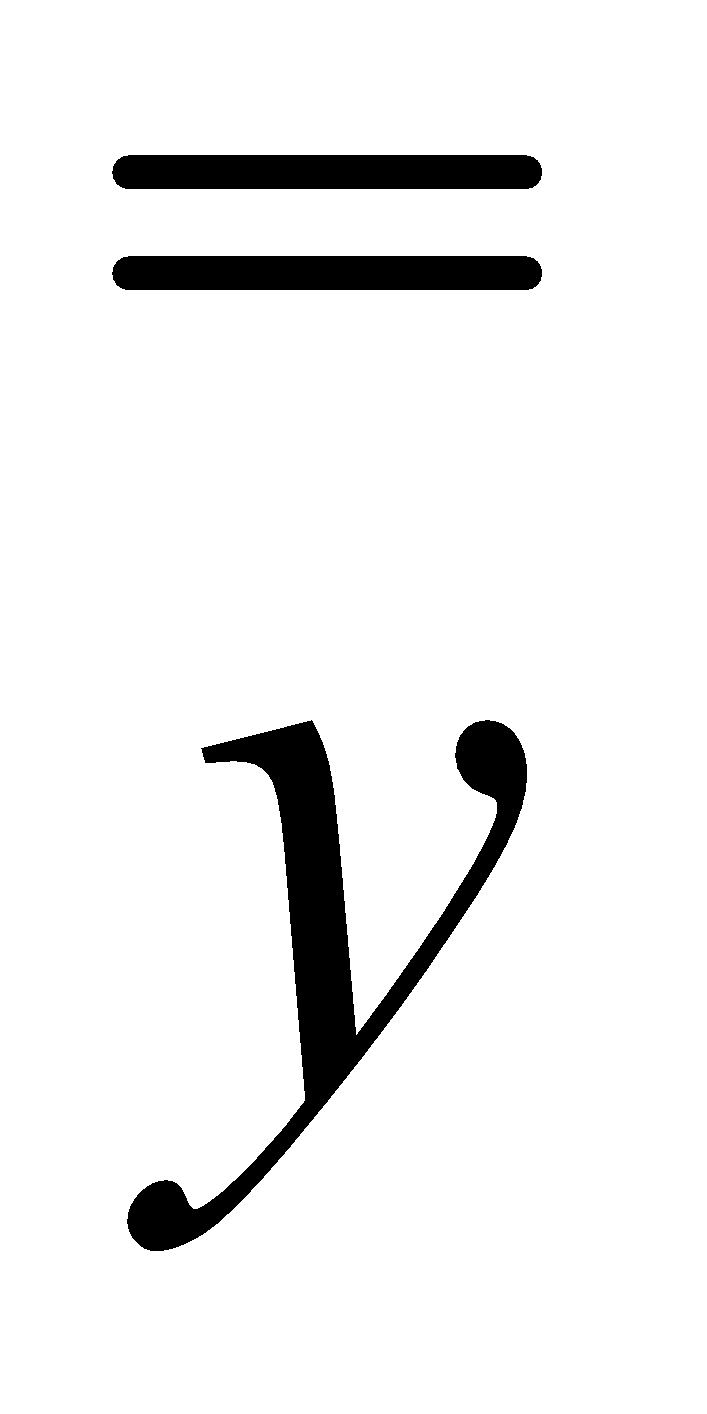
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. ID | Сценарій | Кроки | Вхідні дані | | | Очікуваний результат | Експ. результат | Pass? |
| A1 | Перевірити введення порожніх полів | 1. Залишити всі поля порожніми  2. Залишити одне поле порожнім | 1.Усі порожні поля  2. Усі, окрім 1, заповнені поля | | | 1,2. Повідомлення про помилку. | 1,2. Повідомлення про помилку | Так |
| А2 | Перевірити обробку буквенно-числових рядків | 1. Увести рядок, що містить букви. | 1. «івфів»  2. «ы12вав» | | | 1,2. Повідомлення про помилку. | 1,2. Повідомлення про помилку. | Так |
| A3 | Увести спеціальні символи, включно з «-» і «+» | 1. Увести від’ємне число  2. Увести число зі спец. символом | 1. «-432»  2. «64&7»  3. «+221» | | | 1. Нормальна робота – розв’язок СЛАР  2. Повідомлення про помилку.  3. Нормальна робота – розв’язок СЛАР | 1. Нормальна робота – розв’язок СЛАР  2. Повідомлення про помилку.  3. Нормальна робота – розв’язок СЛАР | Так |
| А4 | Перевірити гранично допустимі числа | 1. Увести граничні значення | 1. «-1000000»  2. «-99999»  3. «999999»  4. «1000000» | | | 1.Повідомлення про помилку  2. Нормальна робота  3. Нормальна робота  4.Повідомлення про помилку | 1. Повідомлення про помилку  2. Нормальна робота  3. Нормальна робота  4. Повідомлення про помилку | Так |
| A5 | Увести значення інтервалу | 1. Початок інтервалу 10. Кінець інтервалу -5  2. Початок інтервалу 5. Кінець інтервалу 10 | | 1. 10  2. -5  3. 5  4. 10 | 1. Повідомлення про помилку  2. Нормальна робота | | 1.Повідомлення про помилку  2. Нормальна робота | Так |

# 4 Опис методів

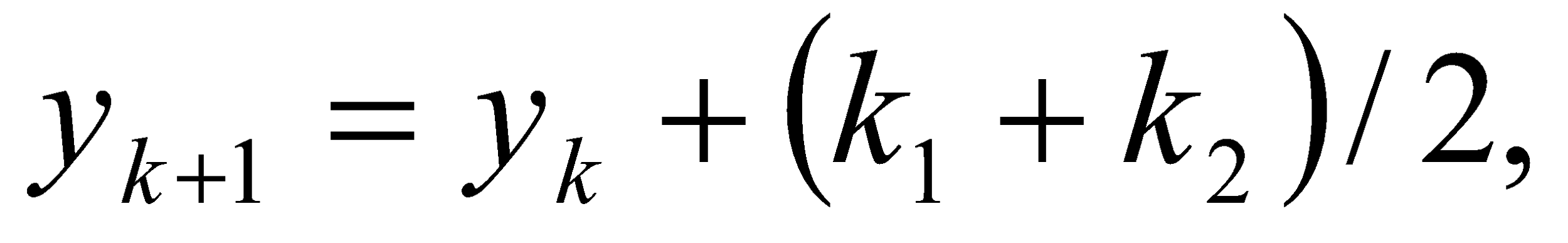
Для багатьох практичних додатків застосовуються програми, розроблені на основі методів Рунге-Кутта. Часто використовуються методи аналізу і корекції.

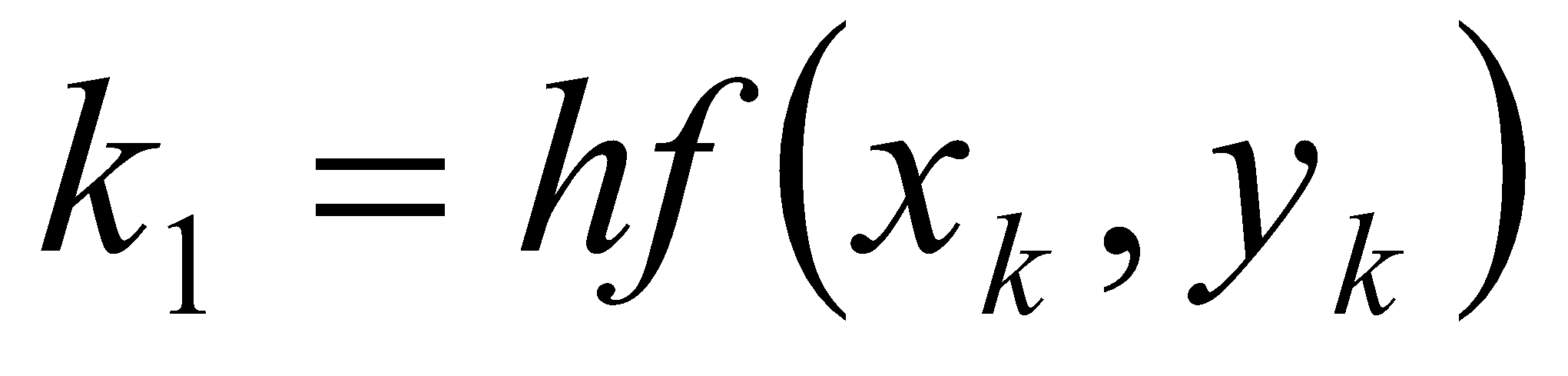
Метод Рунге-Кутта має досить високу точність і легко програмується. Оскільки для обчислення *yi+1* необхідно знати тільки одне значення *yi*, за допомогою цього методу можна починати розв’язання рівняння. Величину кроку можна змінити на будь-якому етапі розрахунків. Одним з недоліків методу є необхідність багаторазового обчислення значення функції на кожному кроці, що призводить до витрати великої кількості машинного часу.

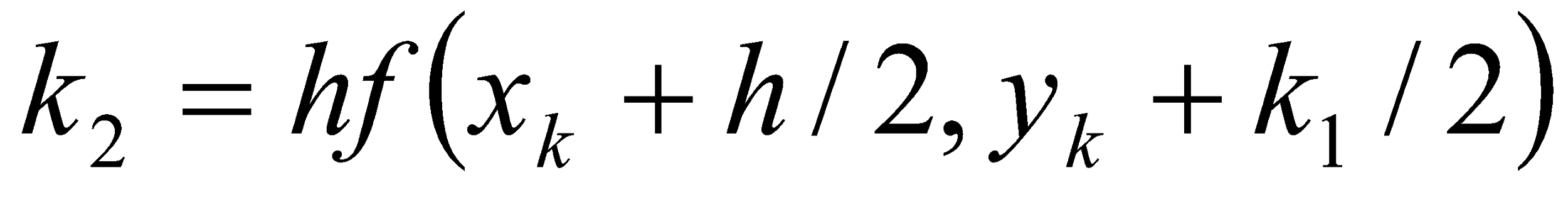
Можна рекомендувати наступні правила автоматичного обрання кроку.

У вузлі *х0* узяти *h=h0*, де *h0*− заданий початковий крок, знайти наближені значення розв'язку  і , обчислені методом Рунге-Кутта в точці *x0 + h* із кроком відповідно *h* і *h0*. Розв'язок в наступному вузлі х2, виходячи з вузла х1, одержують аналогічним шляхом.

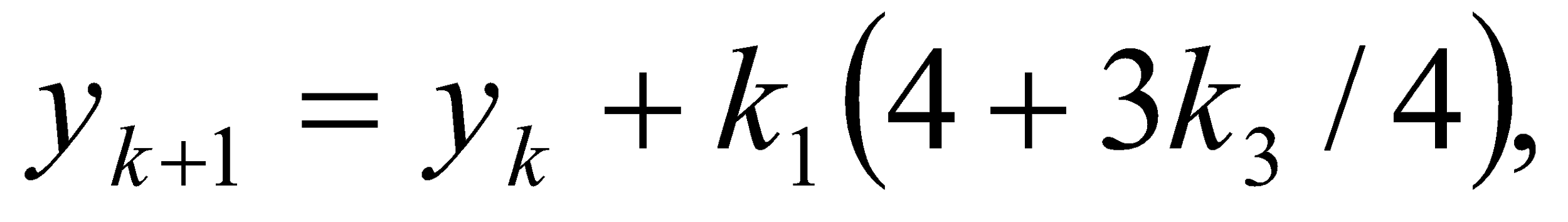
## **4.1 Метод Рунге-Кутта 2-го порядку**

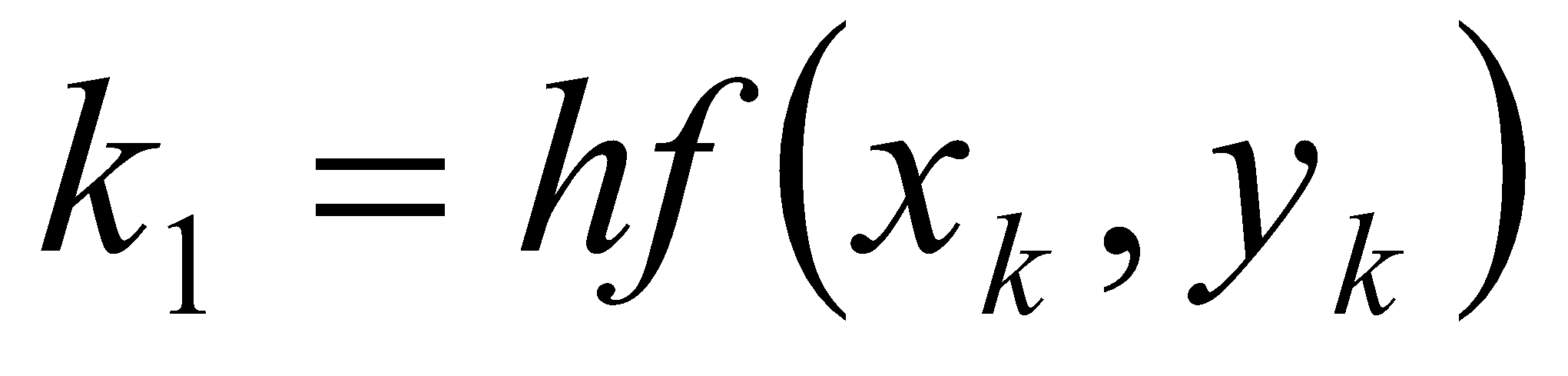


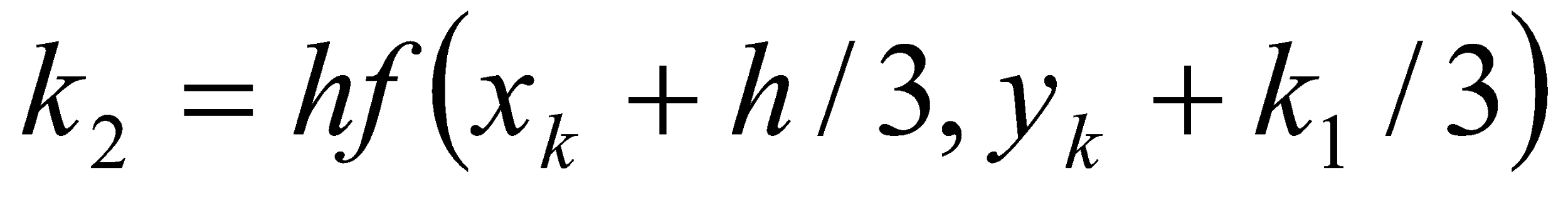
де ;

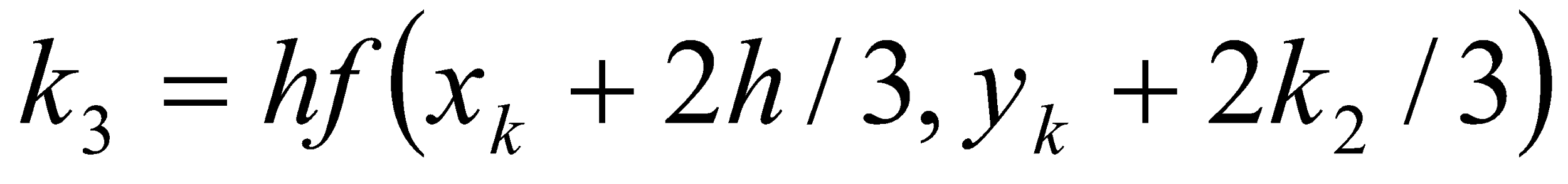
.

## **4.2 Метод Рунге-Кутта 3-го порядку**

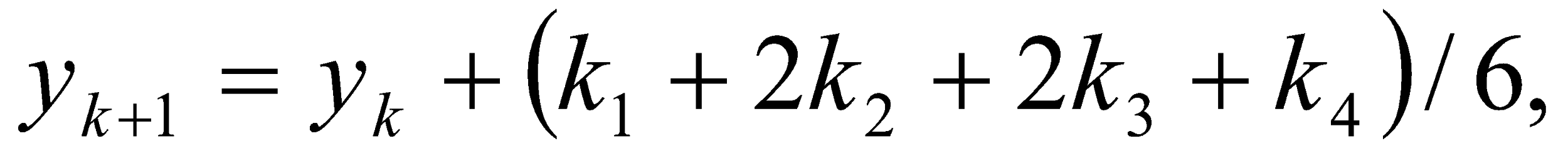


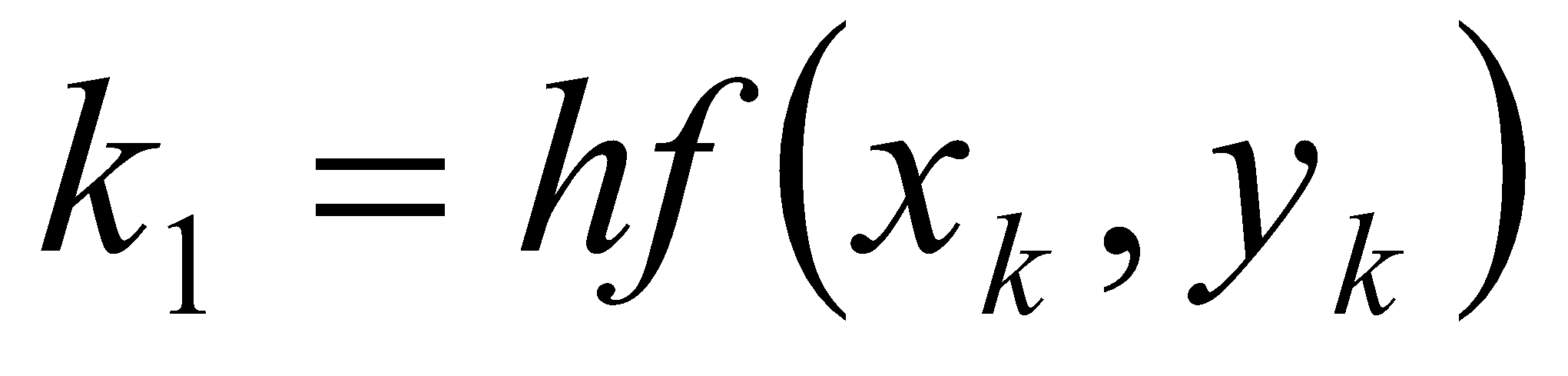
де ;

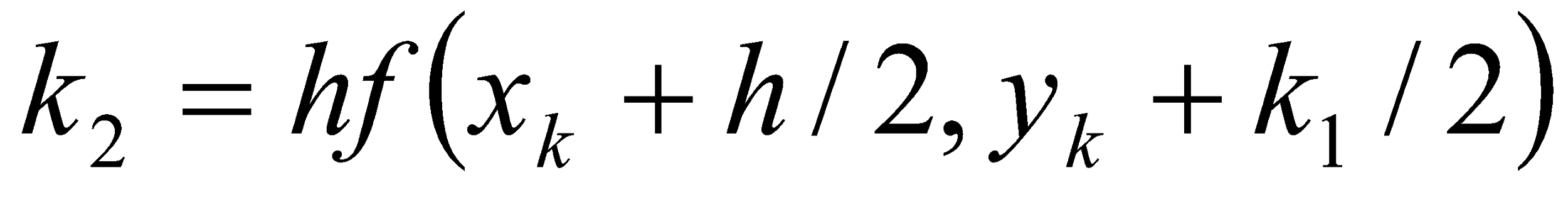
;

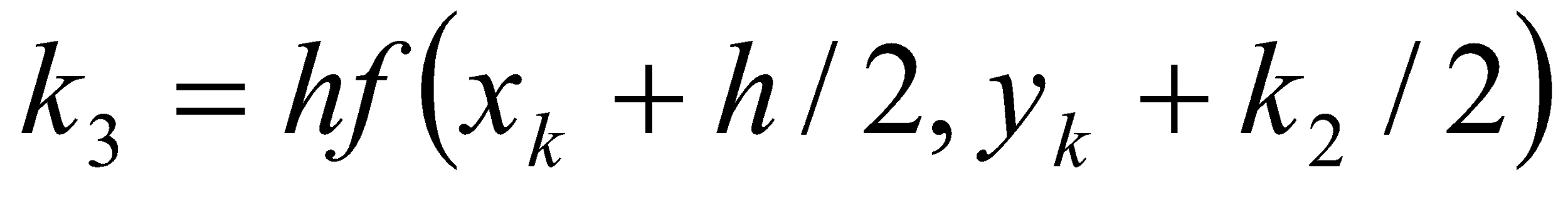
.

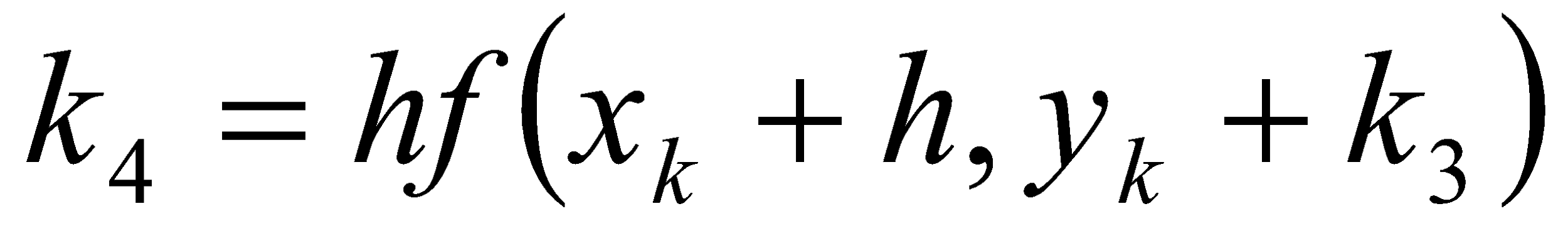
## **4.3 Метод Рунге-Кутта 4-го порядку**



де ;

;

;

.

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 5 Висновки

Насамкінець, продемонструємо роботу програм на індивідуальному завданні для варіанту 5, а саме таких функціях:



При виконанні даної лабораторної роботи вивчені правил використання програмних засобів чисельного розв’язання звичайних диференціальних рівнянь, практично розв’язане рівняння на комп’ютері з використанням СКМ, проведено порівняльний аналіз методів інтегрування ОДУ.

Похибка між значеннями отриманими методом Рунге-Кутти та даними отриманими вбудованою функцією становить:

0.000103 для 4-го порядку

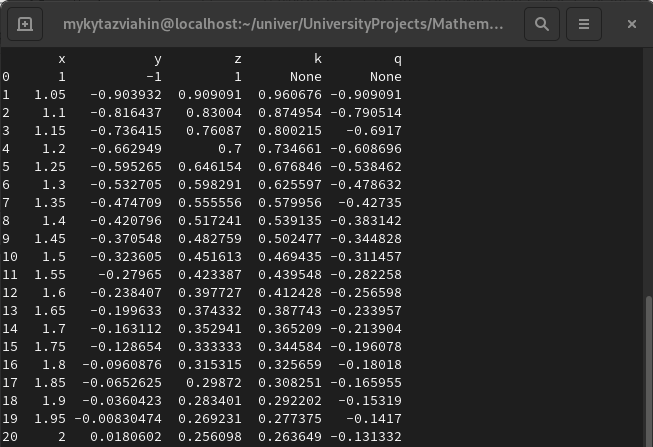
0.000289 для 3-го порядку

0.000344 для 2-го порядку

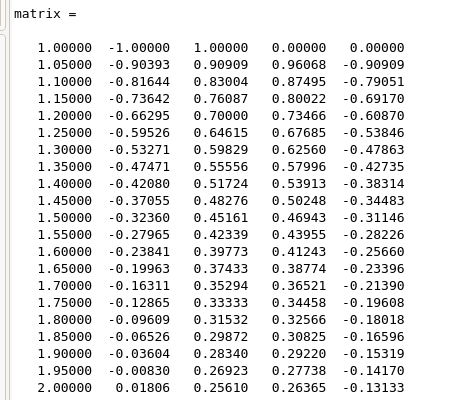
Тобото найточніший - Рунге-Кутта 4-го порядку

Табличні значення:

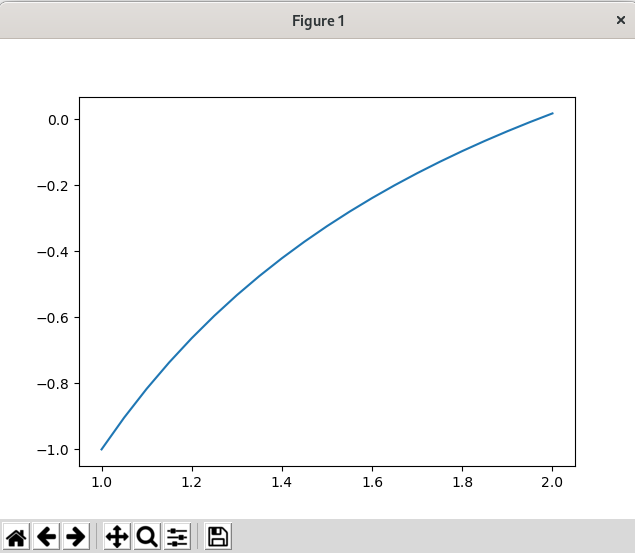
Метод Рунге-Кутта 4-го порядку Python:



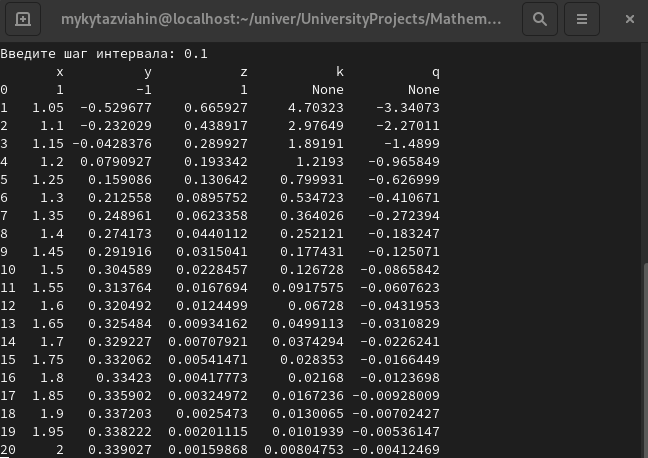
Метод Рунге-Кутта 4-го порядку Octave:



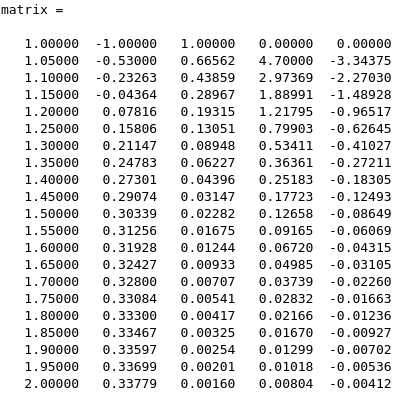
Графік функції



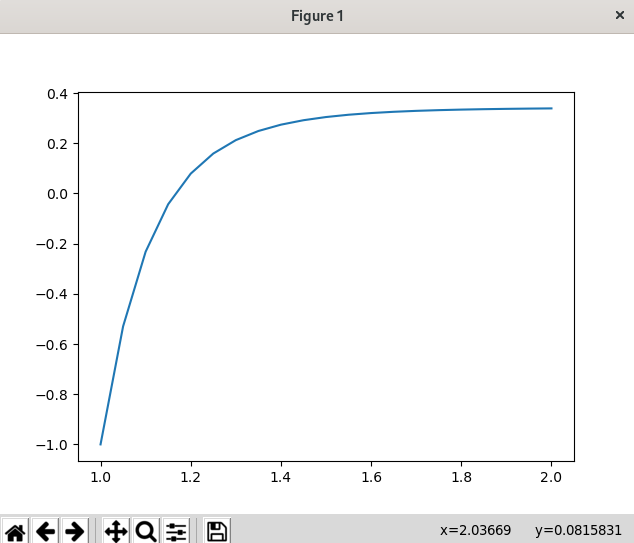
Метод Рунге-Кутта 3-го порядку Python:



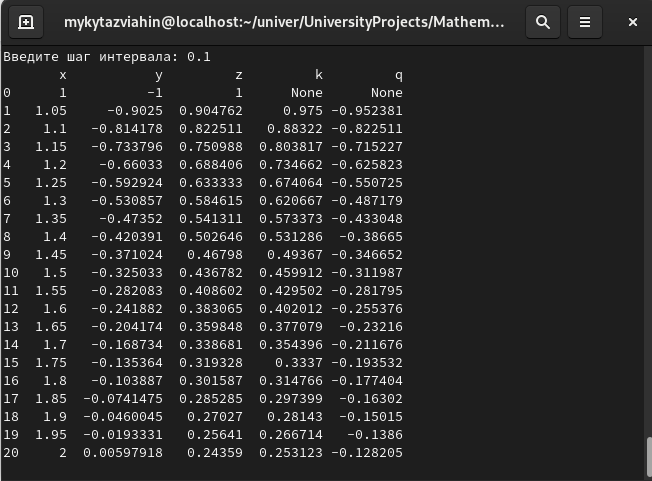
Метод Рунге-Кутта 3-го порядку Octave:



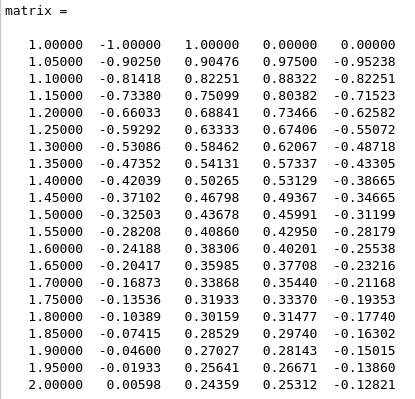
Графік функції



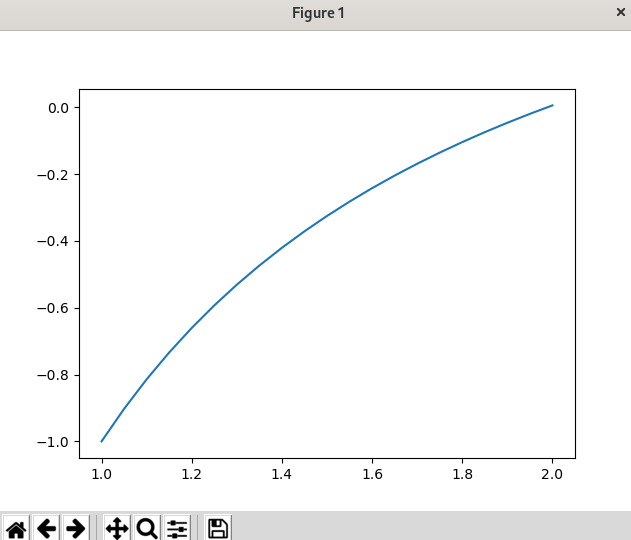
Метод Рунге-Кутта 2-го порядку Python:



Метод Рунге-Кутта 2-го порядку Octave:

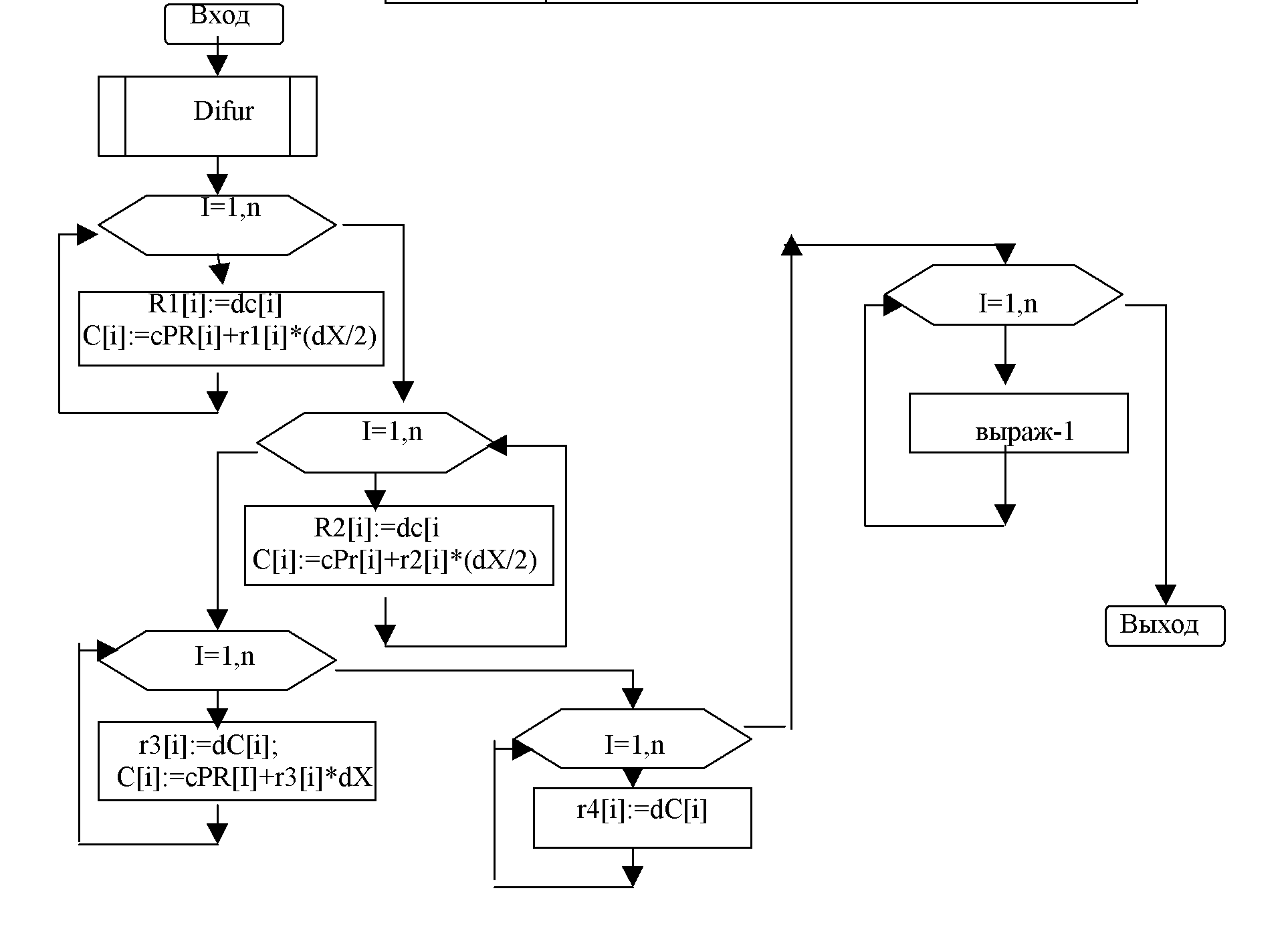


Графік функції



# Додаток А. Графи потоку керування для модулів програм

Метод Рунге-Кутта 4-го порядку



# 

# 

# 

# 

# 

# 

# 

# Додаток Б. Лістинг програми.

# **runge4.py**

import pandas as pd

import numpy as np

import pylab

def func(z, x):

return -z / x

try:

a = float(input('Ввежите начальную границу промежутка: '))

b = float(input('Введите конечную границу промежутка: '))

if a>=b:

print('Начальная граница не может быьб больше конечной')

raise SystemExit

elif a<-999999 or a>999999 or b>999999 or b<-999999:

print('Интервал в не границ диапазона. Интервал должен находится в пределах -999999 до 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

try:

h = float(input('Введите шаг интервала: '))

if h<=0 or h > 999999:

print('Интервал должен быть между -999999 и 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

n = int((b-a)/(h/2))

matrix = [[1, -1, 1, None, None]]

for i in range(1, n + 1):

matrix.append([None, None, None, None, None])

k = []

q = []

matrix[i][0] = (matrix[i-1][0] + h/2)

q.append(func(matrix[i-1][2], matrix[i-1][0]))

q.append(func(matrix[i-1][2] + q[0]\*h/2, matrix[i-1][0] + h/2))

q.append(func(matrix[i-1][2] + q[1]\*h/2, matrix[i-1][0] + h/2))

q.append(func(matrix[i-1][2] + q[2]\*h, matrix[i-1][0] + h))

matrix[i][4] = (q[0] + 2\*q[1] + 2\*q[2] + q[3])/6

k.append(matrix[i-1][2])

k.append(matrix[i-1][2] + q[0]\*h/2)

k.append(matrix[i-1][2] + q[1]\*h/2)

k.append(matrix[i-1][2] + q[2]\*h/2)

matrix[i][3] = (k[0] + 2\*k[1] + 2\*k[2] + k[3])/6

matrix[i][1] = (matrix[i-1][1] + matrix[i][3]\*h)

matrix[i][2] = (matrix[i-1][2] + matrix[i][4]\*h)

matrixStruct = pd.DataFrame(np.array(matrix),

columns=['x', 'y', 'z', 'k', 'q'])

print(matrixStruct)

diffenation = []

for i in range(n):

R1 = abs(matrix[i][0]-matrix[i+1][0])/((2\*\*4)-1)

R2 = abs(matrix[i][1]-matrix[i+1][1])/((2\*\*4)-1)

diffenation.append(max([R1,R2]))

print(max(diffenation))

xlist = [i for i in matrixStruct['x']]

ylist = [i for i in matrixStruct['y']]

pylab.plot (xlist, ylist)

pylab.show()

**runge3.py**

import pandas as pd

import numpy as np

import pylab

def func(z, x):

return -z / x

try:

a = float(input('Введите начальную границу промежутка: '))

b = float(input('Введите конечную границу промежутка: '))

if a>=b:

print('Начальная граница не может быьб больше конечной')

raise SystemExit

elif a<-999999 or a>999999 or b>999999 or b<-999999:

print('Интервал в не границ диапазона. Интервал должен находится в пределах -999999 до 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

try:

h = float(input('Введите шаг интервала: '))

if h<=0 or h > 999999:

print('Интервал должен быть между -999999 и 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

n = int((b-a)/(h/2))

matrix = [[1, -1, 1, None, None]]

for i in range(1, n + 1):

matrix.append([None, None, None, None, None])

k = []

q = []

matrix[i][0] = (matrix[i-1][0] + h/2)

q.append(func(matrix[i-1][2], matrix[i-1][0]))

q.append(func(matrix[i-1][2] + q[0]\*h/3, matrix[i-1][0] + h/3))

q.append(func(matrix[i-1][2] + q[1]\*2\*h/3, matrix[i-1][0] + 2\*h/3))

matrix[i][4] = q[0]\*(4 + 3\*q[2]/4)

k.append(matrix[i-1][2])

k.append(matrix[i-1][2] + q[0]\*h/3)

k.append(matrix[i-1][2] + q[1]\*2\*h/3)

matrix[i][3] = k[0]\*(4 + 3\*k[2]/4)

matrix[i][1] = (matrix[i-1][1] + matrix[i][3]\*h)

matrix[i][2] = (matrix[i-1][2] + matrix[i][4]\*h)

matrix = pd.DataFrame(np.array(matrix),

columns=['x', 'y', 'z', 'k', 'q'])

print(matrix)

xlist = [i for i in matrix['x']]

ylist = [i for i in matrix['y']]

pylab.plot (xlist, ylist)

pylab.show()

**runge2.py**

import pandas as pd

import numpy as np

import pylab

def func(z, x):

return -z / x

try:

a = float(input('Ввежите начальную границу промежутка: '))

b = float(input('Введите конечную границу промежутка: '))

if a>=b:

print('Начальная граница не может быьб больше конечной')

raise SystemExit

elif a<-999999 or a>999999 or b>999999 or b<-999999:

print('Интервал в не границ диапазона. Интервал должен находится в пределах -999999 до 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

try:

h = float(input('Введите шаг интервала: '))

if h<=0 or h > 999999:

print('Интервал должен быть между -999999 и 999999')

raise SystemExit

except:

print('Введена строка, а не число')

raise SystemExit

n = int((b-a)/(h/2))

matrix = [[1, -1, 1, None, None]]

for i in range(1, n + 1):

matrix.append([None, None, None, None, None])

k = []

q = []

matrix[i][0] = (matrix[i-1][0] + h/2)

q.append(func(matrix[i-1][2], matrix[i-1][0]))

q.append(func(matrix[i-1][2] + q[0]\*h/2, matrix[i-1][0] + h/2))

matrix[i][4] = (q[0] + q[1])/2

k.append(matrix[i-1][2])

k.append(matrix[i-1][2] + q[0]\*h/2)

matrix[i][3] = (k[0] + k[1])/2

matrix[i][1] = (matrix[i-1][1] + matrix[i][3]\*h)

matrix[i][2] = (matrix[i-1][2] + matrix[i][4]\*h)

matrix = pd.DataFrame(np.array(matrix),

columns=['x', 'y', 'z', 'k', 'q'])

print(matrix)

xlist = [i for i in matrix['x']]

ylist = [i for i in matrix['y']]

pylab.plot (xlist, ylist)

pylab.show()

**runge4.m**

try

a = input('Enter the start boundary of the gap: ');

b = input('Enter the end boundary of the gap: ');

if (a>=b)

errordlg('The initial boundary cannot be greater than the final');

return

elseif (a<-999999 || a>999999 || b>999999 || b<-999999)

errordlg('The interval is not within the range. The interval must be between -999999 and 999999');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

try

h = input('Insert the number of breaking parts: ');

if (h<=0 || h>999999)

errordlg ('The number of partitions cannot be less than 1');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

n = (b-a)/(h/2)

matrix = zeros(n, 5);

matrix(1,1) = 1

matrix(1,2) = -1

matrix(1,3) = 1

function res = func(z,x)

res = -z/x;

endfunction

for i = 2:n + 1

k = zeros(4, 1);

q = zeros(4, 1);

matrix(i,1) = (matrix(i-1,1) + h/2);

q(1) = (func(matrix(i-1,3), matrix(i-1,1)));

q(2) = (func(matrix(i-1,3) + q(1)\*h/2, matrix(i-1,1) + h/2));

q(3) = (func(matrix(i-1,3) + q(2)\*h/2, matrix(i-1,1) + h/2));

q(4) = (func(matrix(i-1,3) + q(3)\*h, matrix(i-1,1) + h));

matrix(i,5) = (q(1) + 2\*q(2) + 2\*q(3) + q(4))/6;

k(1) = matrix(i-1, 3);

k(2) = (matrix(i-1, 3) + q(1)\*h/2);

k(3) = (matrix(i-1, 3) + q(2)\*h/2);

k(4) = (matrix(i-1, 3) + q(3)\*h/2);

matrix(i,4) = (k(1) + 2\*k(2) + 2\*k(3) + k(4))/6;

matrix(i,2) = (matrix(i-1,2) + matrix(i,4)\*h);

matrix(i,3) = (matrix(i-1,3) + matrix(i,5)\*h);

endfor

matrix

**runge3.m**

try

a = input('Enter the start boundary of the gap: ');

b = input('Enter the end boundary of the gap: ');

if (a>=b)

errordlg('The initial boundary cannot be greater than the final');

return

elseif (a<-999999 || a>999999 || b>999999 || b<-999999)

errordlg('The interval is not within the range. The interval must be between -999999 and 999999');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

try

h = input('Insert the number of breaking parts: ');

if (h<0 || h>999999)

errordlg ('The number of partitions cannot be less than 1');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

n = (b-a)/(h/2)

matrix = zeros(n, 5);

matrix(1,1) = 1

matrix(1,2) = -1

matrix(1,3) = 1

function res = func(z,x)

res = -z/x;

endfunction

for i = 2:n+1

k = zeros(3, 1);

q = zeros(3, 1);

matrix(i,1) = (matrix(i-1,1) + h/2);

q(1) = (func(matrix(i-1,3), matrix(i-1,1)));

q(2) = (func(matrix(i-1,3) + q(1)\*h/3, matrix(i-1,1) + h/3))

q(3) = (func(matrix(i-1,3) + q(1)\*2\*h/3, matrix(i-1,1) + 2\*h/3));

matrix(i,5) = q(1)\*(4 + 3\*q(3)/4);

k(1) = matrix(i-1, 3);

k(2) = (matrix(i-1,3) + q(1)\*h/3);

k(3) = (matrix(i-1,3) + q(1)\*2\*h/3);

matrix(i,4) = k(1)\*(4 + 3\*k(3)/4);

matrix(i,2) = (matrix(i-1,2) + matrix(i,4)\*h);

matrix(i,3) = (matrix(i-1,3) + matrix(i,5)\*h);

endfor

matrix

**runge2.m**

try

a = input('Enter the start boundary of the gap: ');

b = input('Enter the end boundary of the gap: ');

if (a>=b)

errordlg('The initial boundary cannot be greater than the final');

return

elseif (a<-999999 || a>999999 || b>999999 || b<-999999)

errordlg('The interval is not within the range. The interval must be between -999999 and 999999');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

try

h = input('Insert the number of breaking parts: ');

if (h<=0 || h>999999)

errordlg ('The number of partitions cannot be less than 1');

return

endif

catch

errordlg ('Introduced string');

return

end\_try\_catch

n = (b-a)/(h/2)

matrix = zeros(n, 5);

matrix(1,1) = 1

matrix(1,2) = -1

matrix(1,3) = 1

function res = func(z,x)

res = -z/x;

endfunction

for i = 2:n + 1

k = zeros(2, 1);

q = zeros(2, 1);

matrix(i,1) = (matrix(i-1,1) + h/2);

q(1) = (func(matrix(i-1,3), matrix(i-1,1)));

q(2) = (func(matrix(i-1,3) + q(1)\*h/2, matrix(i-1,1) + h/2));

matrix(i,5) = (q(1) + q(2))/2;

k(1) = matrix(i-1, 3);

k(2) = (matrix(i-1, 3) + q(1)\*h/2);

matrix(i,4) = (k(1) + k(2))/2;;

matrix(i,2) = (matrix(i-1,2) + matrix(i,4)\*h);

matrix(i,3) = (matrix(i-1,3) + matrix(i,5)\*h);

endfor

matrix