Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра інженерії програмного забезпечення в енергетиці

**ДОМАШНЯ РОБОТА №3**

**з дисципліни «Математичні моделі процесів і систем»**

**тема «Однокрокові методи розв’язання задачі Коші»**

**Варіант № \_13\_**

**Виконав:**

**Студент 3 курсу, групи *\_ТІ-01\_\_***

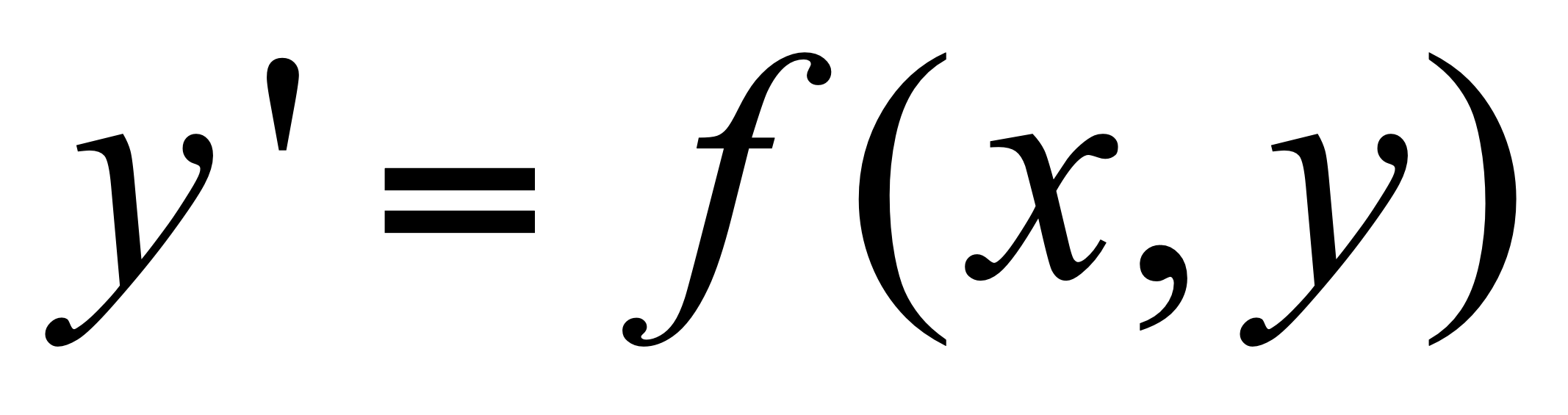
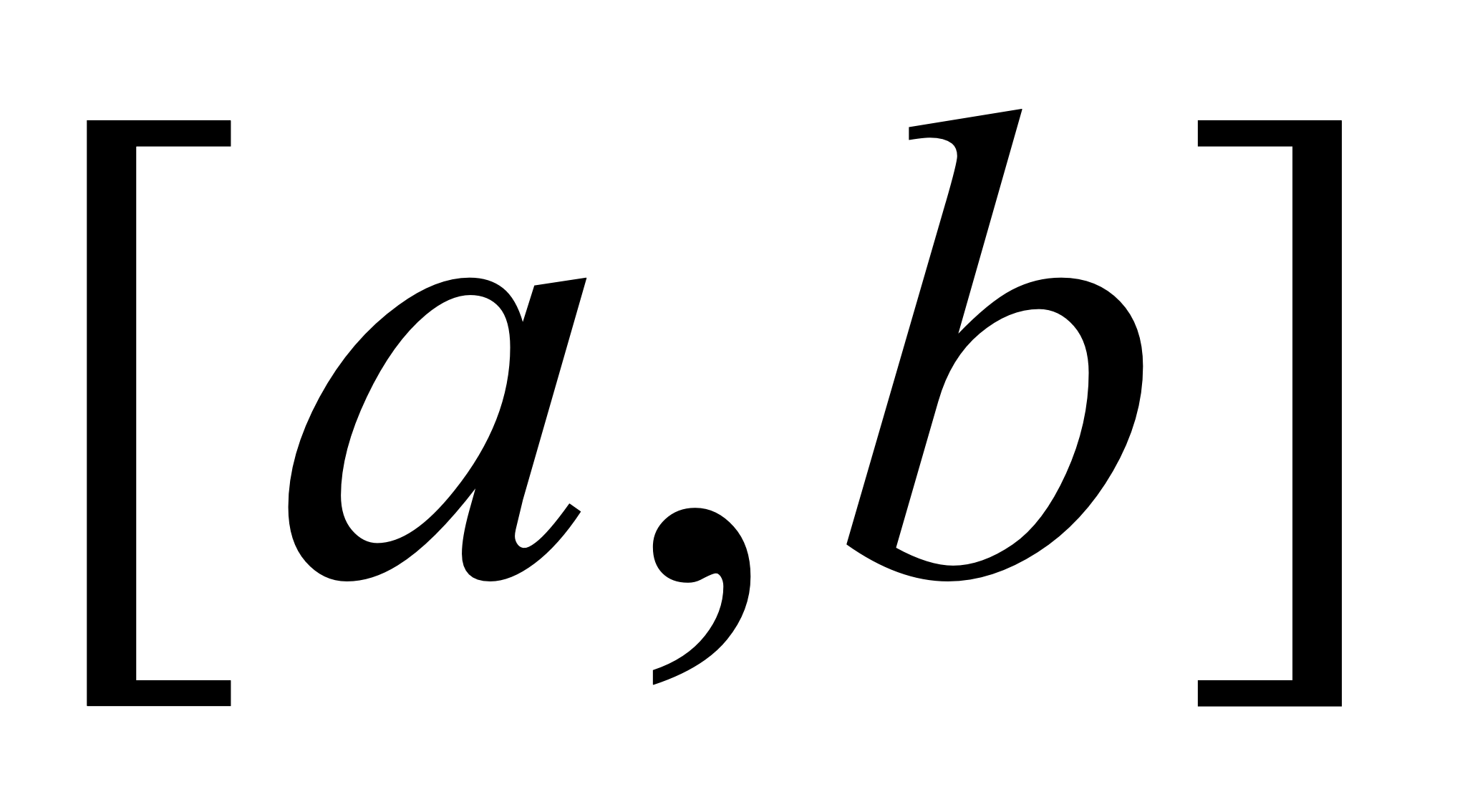
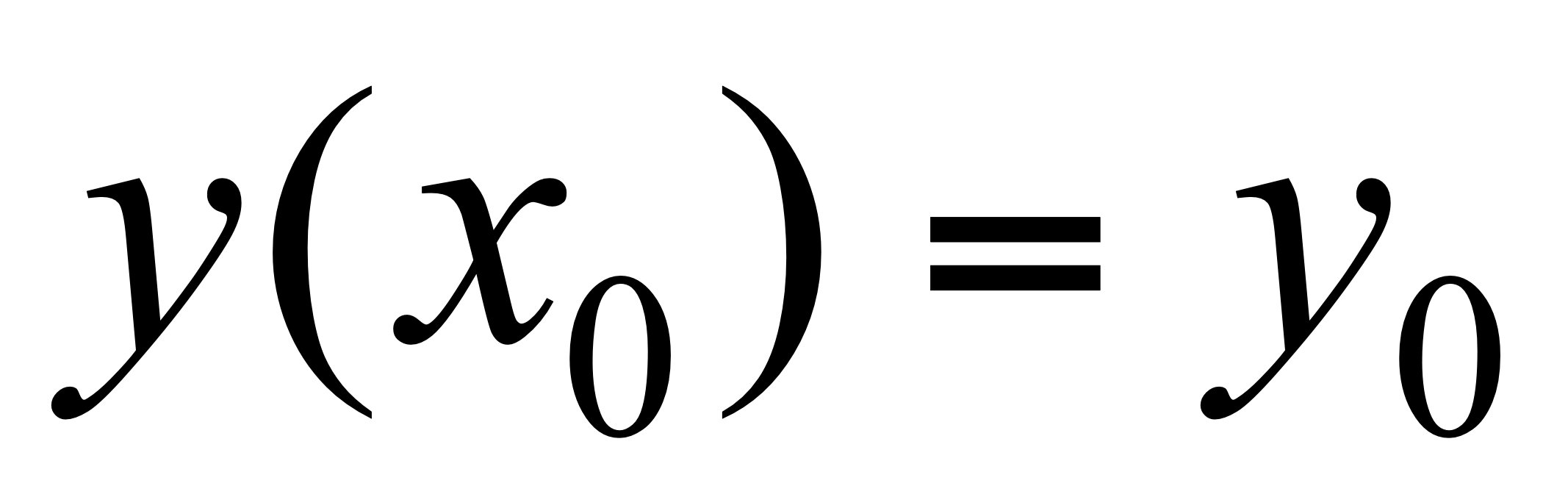
***\_Круть Катерина\_\_***

**(прізвище ім’я)**

**Дата здачі \_\_\_\_\_2023-03-14\_\_\_\_\_\_\_**

**Київ – 2023**

**Завдання.**

З точністю до 0.0001 знайти розв’язок задачі Коші для звичайного диференційного рівняння першого порядку  на відрізку ****** з кроком*h*=0.1 за початкових умов :

1. чисельними методами (розробити програму на одній з мов програмування та в Excel):
   1. методом Ейлера (явним);
   2. методом Ейлера-Коші;
   3. вдосконаленим методом Ейлера;
   4. методом Рунге-Кутта 4-го порядку;
2. використовуючи точний розв’язок диференціального рівняння, отриманий одним з математичних пакетів або онлайн-калькулятора;

|  |  |  |  |
| --- | --- | --- | --- |
| **13** | Изображение | *y*(1.4)=2.5 | [1.4;2.4] |

1. в одній системі координат побудувати для кожного методу графіки наближеного розв’язку та інтегральну криву знайденого точного розв’язку в Excel або в розробленій програмі;
2. порівняти отримані наближені значення розв’язку задачі з точним розв’язком чисельно та графічно (табл. 1 та рис.1).

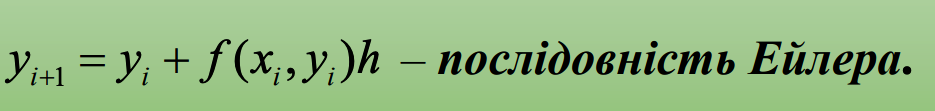
**Дані:**

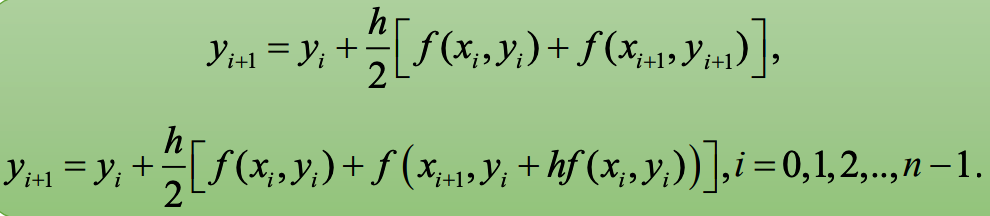
**Алгоритми реалізації:**

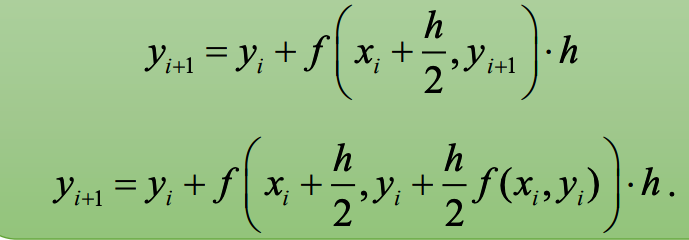
|  |  |
| --- | --- |
| Изображение | Изображение |

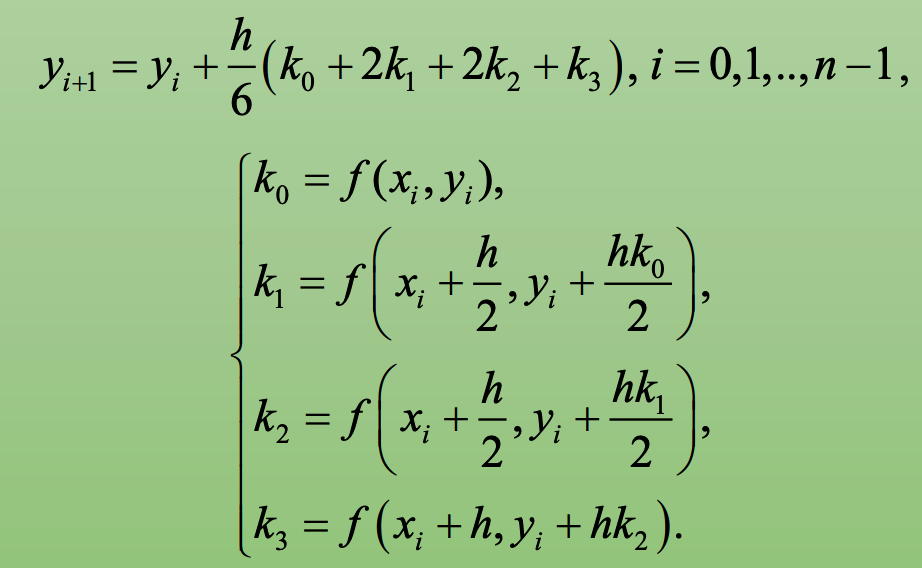
|  |  |
| --- | --- |
| Изображение | Изображение |

**Формули:**

Явний метод Ейлера:

Метод Ейлера-Коші:

Вдосконалений метод Ейлера:

Метод Рунге-Кутта 4-го порядку:

**Програмна реалізація:**

import math

import matplotlib.pyplot as plt

import numpy as np

"""

З точністю до 0.0001 знайти розв’язок задачі Коші для звичайного

диференційного рівняння першого порядку y' = y + cos(x / 13) на відрізку [1.4; 2.4] з

кроком h=0.1 за початкових умов y(1.4) = 2.2

"""

def f(x, y):

""" f(x, y) = y + cos(x/13) """

return y + math.cos(x / 13)

def df(x):

"""

y' = y + cos(x/13)

y' = Ce^x + 13sin(x/13) / 170 - 169cos(x/13) / 170

x0 = 1.4; y0 = 2.5 =>

C = - (13 \* sin(7/65) - 169 \* cos(7/65) - 425) / 170 \* e^1.4

y' = (1/170) \* (e\*sin(x/e) - e^2 \* cos(x/ e) - e^(x -1.4)\*(e\*sin(7/(5e)) - e^2\*cos(7/(5e)) - 5))

"""

return -((13 \* math.sin(7 / 65) - 169 \* math.cos(7 / 65) - 425) \* math.e \*\* (x - 1.4)) / 170 + 13 \* math.sin(

x / 13) / 170 - 169 \* math.cos(x / 13) / 170

def precise\_solution(x0, xn, y0, h):

n = (xn - x0) / h

x = x0

y = y0

result = {'x': [], 'y': []}

for \_ in range(int(n)):

result['x'].append(round(x, 4))

result['y'].append(round(y, 4))

x = x + h

y = df(x)

result['x'].append(round(x, 4))

result['y'].append(round(y, 4))

return result

def euler(x0, xn, y0, h):

n = (xn - x0) / h

x = x0

y = y0

result = {'x': [], 'y': []}

for \_ in range(int(n)):

result['x'].append(round(x, 4))

result['y'].append(round(y, 4))

x = x + h

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

result['x'].append(round(x, 4))

result['y'].append(round(y, 4))

return result

def euler\_cauchy(x0, xn, y0, h):

n = (xn - x0) / h

xi = x0 # x0 = 1.4

yi = y0 # y0 = 2.2

result = {'x': [], 'y': []}

for \_ in range(int(n)):

result['x'].append(round(xi, 4)) # записуємо результуюче значення xi

|  |  |  |  |
| --- | --- | --- | --- |
| **13** | Изображение | *y*(1.4)=2.5 | [1.4;2.4] |

result['y'].append(round(yi, 4)) # записуємо результуюче значення yi

yi1\_ = yi + h \* f(xi, yi) # ‾yi+1 = yi + hƒ(xi, yi)

yi1 = yi + 0.5 \* h \* (f(xi, yi) + f(xi + h, yi1\_)) # yi+1 = yi + h / 2 [ ƒ(xi, yi) + ƒ(xi+1, ‾yi+1) ]

xi = xi + h # перевизначаємо x | x = x + h

yi = yi1 # перевизначаємо yi

result['x'].append(round(xi, 4)) # записуємо результуюче значення xn

result['y'].append(round(yi, 4)) # записуємо результуюче значення yn

return result

def euler\_enhanced(x0, xn, y0, h):

n = (xn - x0) / h

xi = x0 # x0 = 1.4

yi = y0 # y0 = 2.2

result = {'x': [], 'y': []}

for \_ in range(int(n)):

result['x'].append(round(xi, 4)) # записуємо результуюче значення xi

result['y'].append(round(yi, 4)) # записуємо результуюче значення yi

yi1 = yi + h \* f(xi + h / 2, yi + h / 2 \* f(xi, yi)) # yi+1 = yi + hƒ(xi + h / 2, yi + h / 2 ƒ(xi, yi))

xi = xi + h # перевизначаємо x | x = x + h

yi = yi1 # перевизначаємо yi

result['x'].append(round(xi, 4)) # записуємо результуюче значення xn

result['y'].append(round(yi, 4)) # записуємо результуюче значення yn

return result

def runge\_kutte(x0, xn, y0, h):

n = (xn - x0) / h

xi = x0 # x0 = 1.4

yi = y0 # y0 = 2.2

result = {'x': [], 'y': []}

for \_ in range(int(n)):

result['x'].append(round(xi, 4)) # записуємо результуюче значення xi

result['y'].append(round(yi, 4)) # записуємо результуюче значення yi

k0 = f(xi, yi)

k1 = f(xi + 0.5 \* h, yi + 0.5 \* h \* k0)

k2 = f(xi + 0.5 \* h, yi + 0.5 \* h \* k1)

k3 = f(xi + h, yi + h \* k2)

yi1 = yi + h / 6 \* (k0 + 2 \* k1 + 2 \* k2 + k3)

xi = xi + h # перевизначаємо x | x = x + h

yi = yi1 # перевизначаємо yi

result['x'].append(round(xi, 4)) # записуємо результуюче значення xn

result['y'].append(round(yi, 4)) # записуємо результуюче значення yn

return result

def draw(data):

"""Function for drawing graphic"""

x = np.linspace(1.4, 2.4, len(data))

y = data

plt.plot(x, y, 'green', label='y=sin(x)')

plt.xlabel('x')

plt.ylabel('y')

plt.grid(color='black', linestyle='--', linewidth=0.5)

plt.legend(loc='best')

plt.show()

def draw\_all(euler\_, euler\_cauchy\_, euler\_enh, runge, precise):

"""Function for drawing graphic"""

x = np.linspace(1.4, 2.4, len(euler\_))

plt.plot(x, euler\_, 'green', linewidth=0.2, label='Явний метод Ейлера')

plt.plot(x, euler\_cauchy\_, 'black', linewidth=0.2, label='метод Ейлера-Коші')

plt.plot(x, euler\_enh, 'yellow', linewidth=0.2, label='Вдосконалений метод Ейлера')

plt.plot(x, runge, 'red', linewidth=0.2, label='Рунге-Кутта')

plt.plot(x, precise, 'orange', linewidth=0.2, label='Точне')

plt.xlabel('x')

plt.ylabel('y')

plt.grid(color='black', linestyle='--', linewidth=0.5)

plt.legend(loc='best')

plt.show()

def print\_euler(data):

for i in range(len(data['y'])):

print(f'x{i}={data["x"][i]}; y{i}={data["y"][i]}')

def print\_data\_table(euler\_, euler\_cauchy\_, euler\_enhanced\_, runge\_kutte\_, precise\_):

data = [euler\_['x'], euler\_['y'], euler\_cauchy\_['y'], euler\_enhanced\_['y'], runge\_kutte\_['y'], precise\_['y']]

rows = [f'n = {x}' for x in range(len(data[0]))]

data = np.transpose(data)

columns = ('x', 'y Ейлер Явний', 'y Ейлера-Коші', 'y Ейлер Вд.', 'y Рунге-Кутта', 'y Точне')

colors = plt.cm.BuPu(np.linspace(0, 0.5, len(rows)))

cell\_text = [[f'{x}' for x in data[row]] for row in range(len(data))]

plt.box(on=None)

plt.table(cellText=cell\_text, rowLabels=rows, rowColours=colors, colLabels=columns, loc='center')

plt.xticks([])

plt.yticks([])

ax = plt.gca()

ax.get\_xaxis().set\_visible(False)

ax.get\_yaxis().set\_visible(False)

plt.show()

x0\_ = 1.4

xn\_ = 2.4

y0\_ = 2.5

h\_ = 0.1

euler\_data = euler(x0\_, xn\_, y0\_, h\_)

euler\_cauchy\_data = euler\_cauchy(x0\_, xn\_, y0\_, h\_)

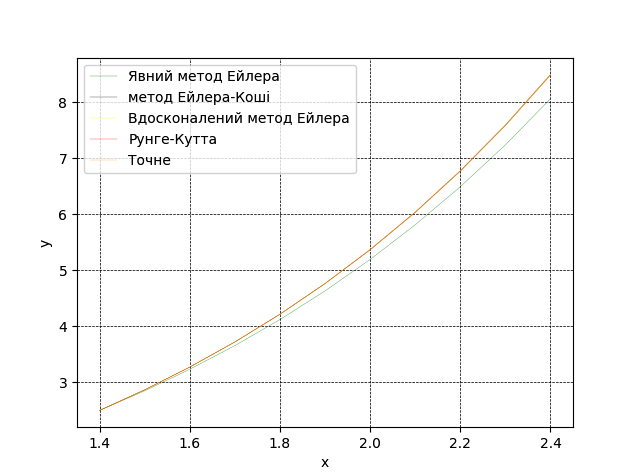
euler\_enhanced\_data = euler\_enhanced(x0\_, xn\_, y0\_, h\_)

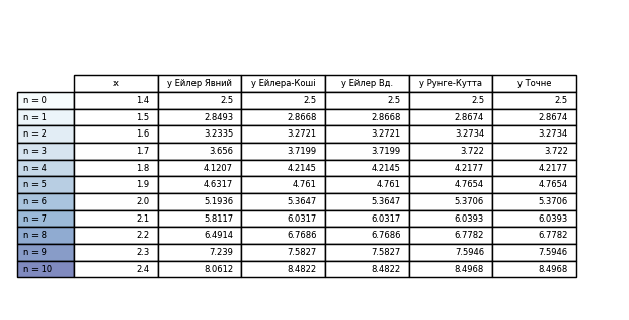
runge\_kutte\_data = runge\_kutte(x0\_, xn\_, y0\_, h\_)

precise\_data = precise\_solution(x0\_, xn\_, y0\_, h\_)

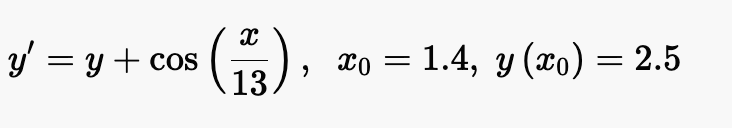
draw\_all(euler\_data['y'], euler\_cauchy\_data['y'], euler\_enhanced\_data['y'], runge\_kutte\_data['y'], precise\_data['y'])

print\_data\_table(euler\_data, euler\_cauchy\_data, euler\_enhanced\_data, runge\_kutte\_data, precise\_data)

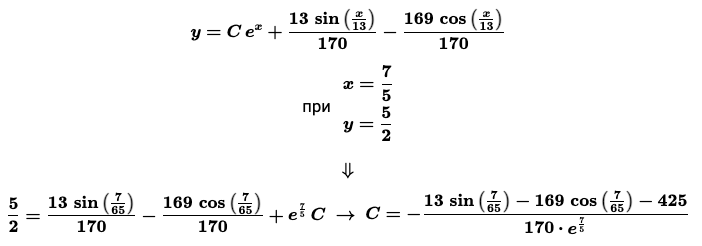
**Результат роботи програми:**

****

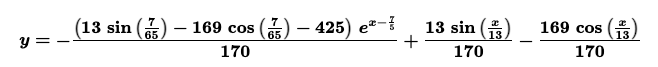
**Ананлітичний розвʼязок задачі Коші:**

1. Умова

2. Загальний розвʼязокИзображение

3. Знаходимо констатнту

2. Точний розвʼязок



**Висновки:**

Під час виконання домашньої роботи біло опрацьовано лекцію №5 «Однокрокові методи розв’язування задачі Коші для звичайних диференціальних рівнянь», набуто навичок для роботи з однокроковими методами розвʼязання задач Коші для звичайних диференціальних рівнянь та розроблено програмний продукт для реалізації алгоритмів вирішення таких задач. Крім того, проведено порівнняня отриманих даних у графічному й табличному вигляді.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **xi** | **Ейлера Неяв.** | **y Точне** | **Рунге-Кутта** | **Ейлера-Коші** | **Ейлера Вдосконалений** |
| 1,4 | 2,5 | 2,5 | 2,5 | 2,5 | 2,5 |
| 1,5 | 2,8493 | 2,8674 | 2,8674 | 2,8668 | 2,8668 |
| 1,6 | 3,2335 | 3,2734 | 3,2734 | 3,2721 | 3,2721 |
| 1,7 | 3,6560 | 3,722 | 3,722 | 3,7199 | 3,7199 |
| 1,8 | 4,1207 | 4,2177 | 4,2177 | 4,2145 | 4,2145 |
| 1,9 | 4,6317 | 4,7654 | 4,7654 | 4,761 | 4,761 |
| 2 | 5,1936 | 5,3706 | 5,3706 | 5,3647 | 5,3647 |
| 2,1 | 5,8117 | 6,0393 | 6,0393 | 6,0317 | 6,0317 |
| 2,2 | 6,4914 | 6,7782 | 6,7782 | 6,7686 | 6,7686 |
| 2,3 | 7,2390 | 7,5946 | 7,5946 | 7,5827 | 7,5827 |
| 2,4 | 8.0612 |  | 8,4968 | 8,4822 | 8,4822 |