**Розрахункова робота №2**

**З предмету: “Чисельні методи ч1”**

**Тема: “Дослідження похибки інтерполяції функцій многочленом Ньютона”**

**Студента групи ПМ-33**

**Венгринюка Олег**

**Варіант - 27**

**Перевірила**

**Тесак І.Є**

**Завдання:**

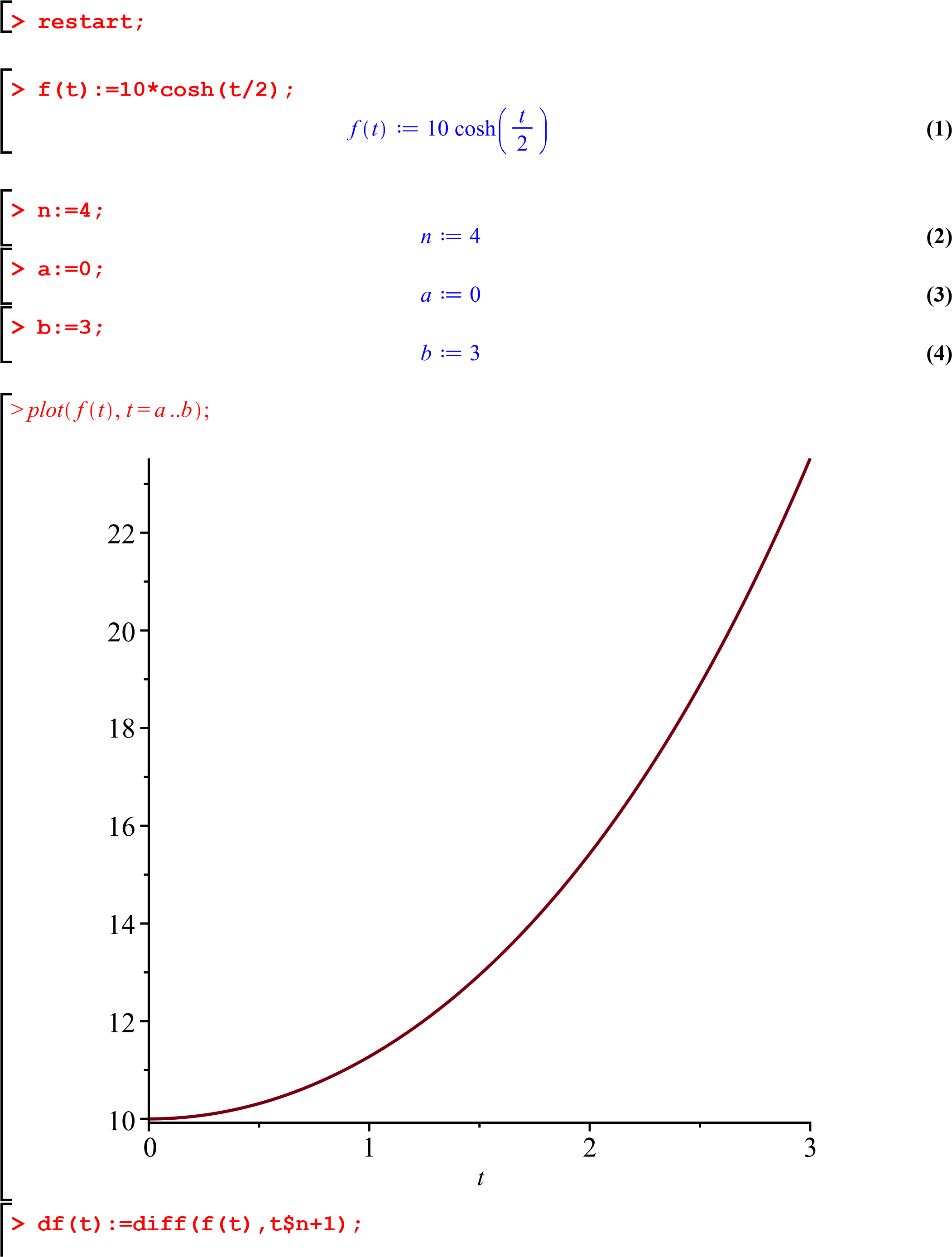
Для функції  заданої на рівномірній сітці в точках  на інтервалі [0, 3]

1. знайти аналітичний вираз залишкового члена інтерполяційного многочлена;
2. обчислити його максимальне значення і значення в точках  ( – не співпадають з вузлами інтерполяції) і в т. – вузлі інтерполяції;
3. побудувати інтерполяційний многочлен Ньютона через розділені різниці (використати програму з лабораторної роботи №3);
4. обчислити похибки інтерполяції в точках , і порівняти їх із значеннями залишкового члена.

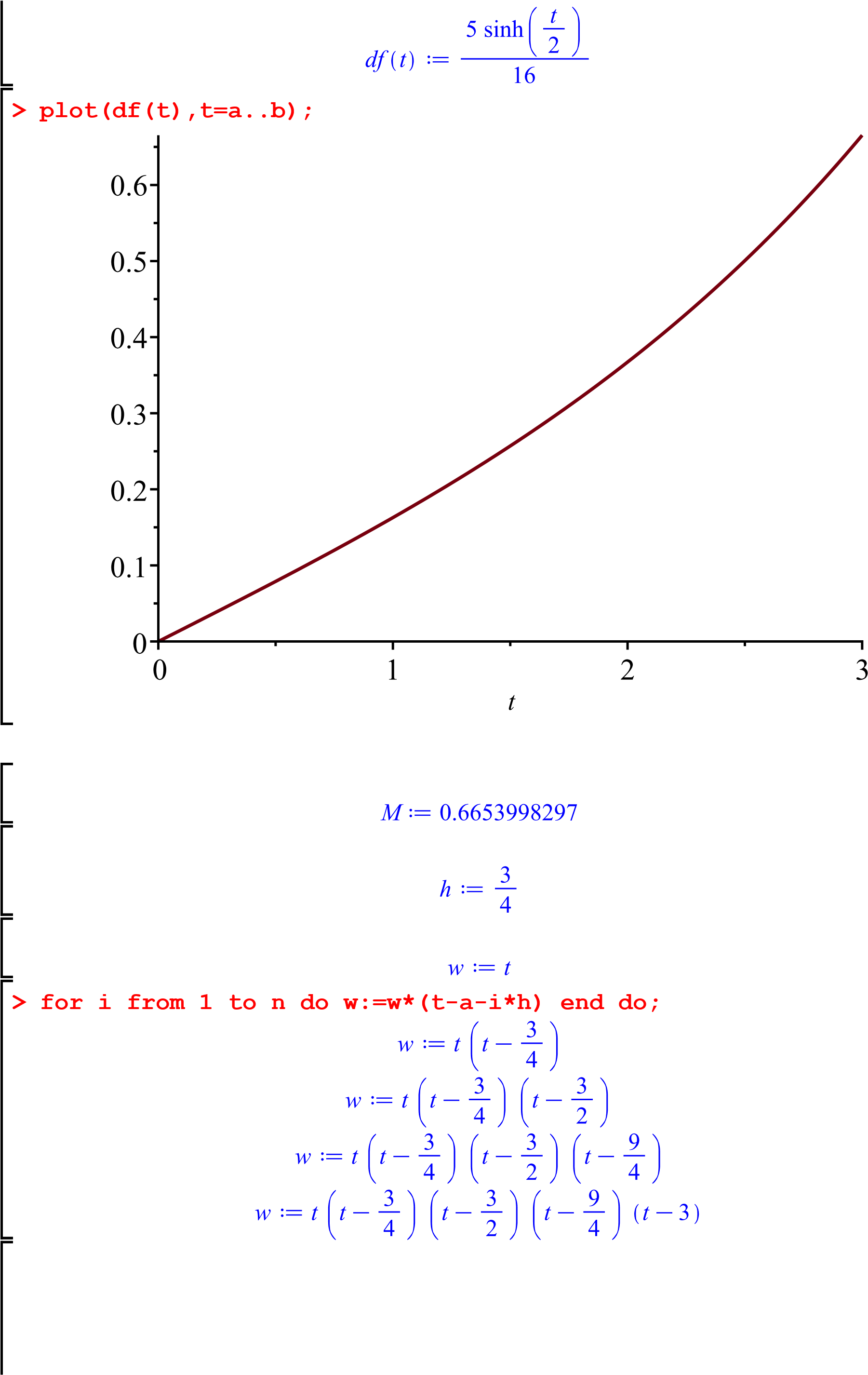
**Виконання:**

1.Знайдемо аналітичний вираз залишкового члена інтерполяційного многочлена

2. Використовуючи засоби Maple обчислимо максимальне значення і значення у точках , що не спіпадають з вузлами інтерполяції і в точці – вузлі інтерполяції;



**(5)**

**> M:=evalf(maximize(abs(diff(f(t),t$n+1)),t=a..b));**

**(6)**

**> h:=(b-a)/n;**

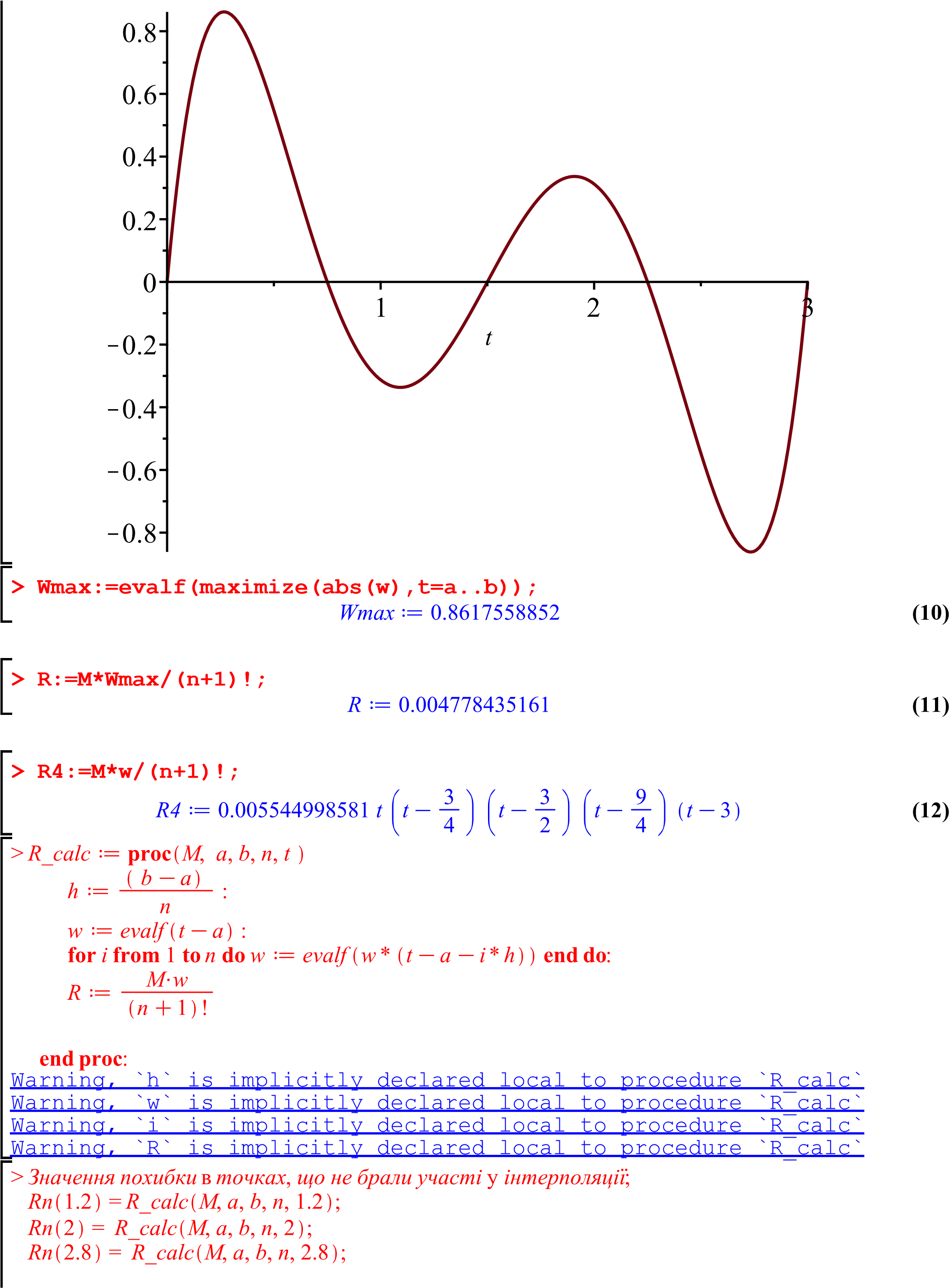
**(7)**

**> w:=t-a;**

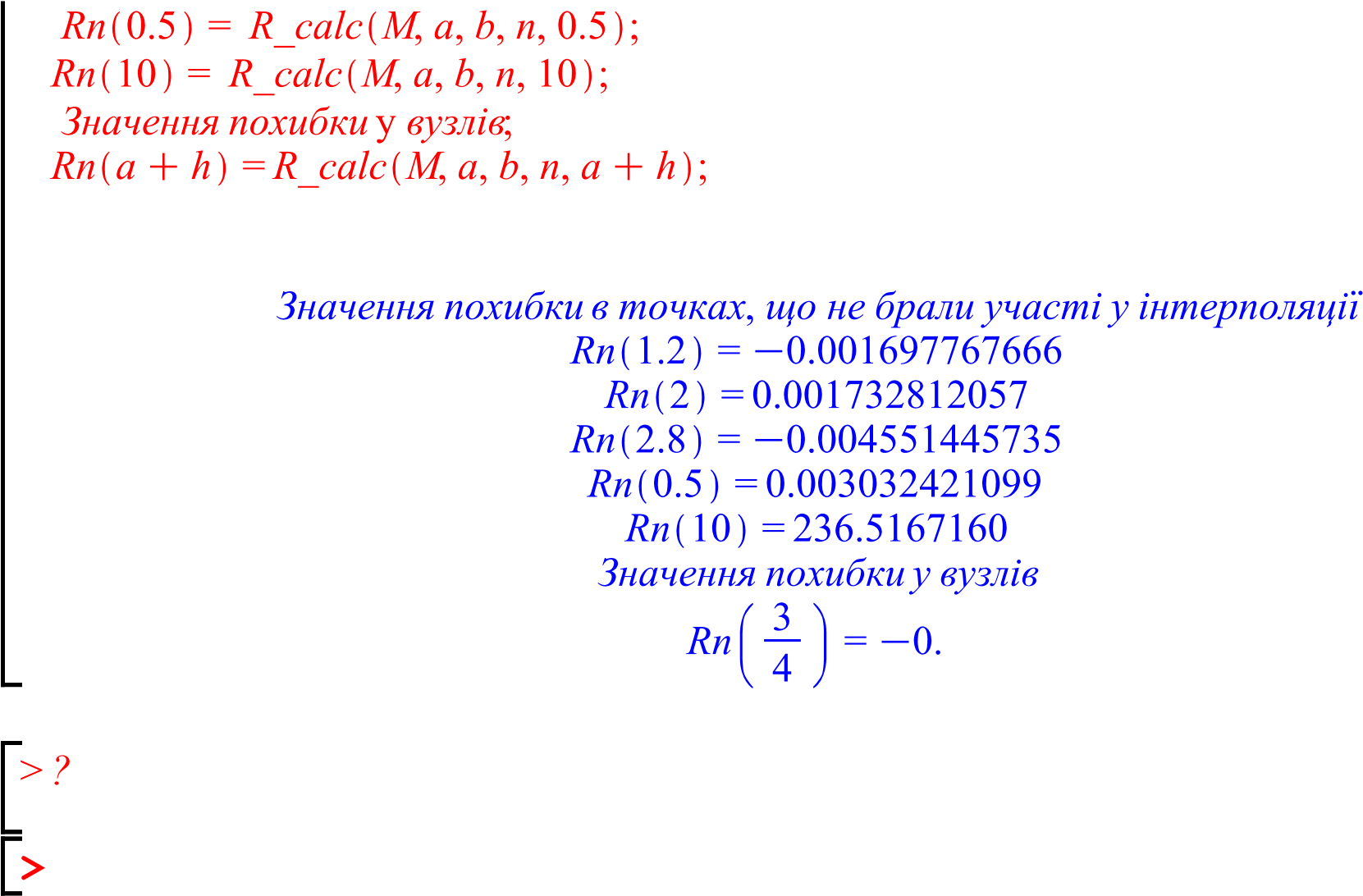
**(8)**

**(9)**

**> plot(w,t=a..b);**



**(13)**

 *`?`* **(14)**

1. побудуємо інтерполяційний многочлен Ньютона через розділені різниці та порівняємо отримані похибки зі значенням аналітично знайденого залишкового члена. Використовуючи мову програмування Python3 та пакети numpy, matplotlib, виконаємо поставлене завдання.

**Програмний код**

import matplotlib.pyplot as plt

import numpy as np

import texttable as tt

def divided\_diff(x: np.ndarray, y: np.ndarray) -> float:

if y.shape != y.shape:

raise ValueError("Y and X should have the same shape, got X.shape = {}, Y.shape = {}".format(x.shape, y.shape))

result = 0

for j in range(y.shape[0]):

indexes = np.arange(y.shape[0]) != j

denominator = np.prod(x[j] - x[indexes])

result += y[j]/denominator

return result

def L(x: float, x\_i: np.ndarray, y\_i: np.ndarray) -> float:

result = y\_i[0]

for i in range(1, y\_i.shape[0]):

indexes = np.arange(i)

indexes\_diff = np.append(indexes, indexes[-1]+1)

result += np.prod(x - x\_i[indexes])\*divided\_diff(x\_i[indexes\_diff], y\_i[indexes\_diff])

return result

def print\_divided\_diff\_table(x: np.ndarray, y: np.ndarray) -> None:

tab = tt.Texttable()

# ----------- set headers

headings = ['X', 'Y']

for i in range(1, y.shape[0]):

head = 'f\_x\_i...f\_x\_i+{}'.format(i)

headings.append(head)

tab.header(headings)

values = [x, y]

# --------- cal divided differences

for i in range(1, y.shape[0]):

divided\_diff\_s = np.array([divided\_diff(x[j: j+i+1], y[j: j+i+1]) for j in range(y.shape[0] - i)])

divided\_diff\_s = np.hstack([divided\_diff\_s, [""]\*(y.shape[0]-1-divided\_diff\_s.shape[0])])

values.append(divided\_diff\_s)

for row in zip(\*values):

tab.add\_row(row)

s = tab.draw()

print(s)

def f(x: float) -> float:

return np.cosh(x/2)/10

def print\_polynomial(X, Y) -> None:

template = '(x-{:.3f})'

polinomial = 'Ln(x) = {:.3f}+'

args = [L(X[0], X, Y)]

for i in range(1, X.shape[0]):

polinomial += template\*i + '{:.3f}+'

args.extend(X[:i].tolist())

args.append(divided\_diff(X[:i+1], Y[:i+1]))

polinomial = polinomial[:-1]

print(polinomial.format(\*args))

def print\_difference(X\_test, X, Y) -> None:

L\_x = np.array([L(x, X, Y) for x in X\_test])

difference = f(X\_test) - L\_x

tab = tt.Texttable()

headings = ['x', 'f(x)', 'L(x)', 'R(x)']

tab.header(headings)

values = [X\_test, f(X\_test), L\_x, difference]

for row in zip(\*values):

tab.add\_row(row)

s = tab.draw()

print(s)

if \_\_name\_\_ == '\_\_main\_\_':

a, b = 0, 3

n = 5

X = np.linspace(a, b, n)

Y = f(X)

X\_test = np.array([1.2, 2., 2.8, 0.5, 0.75, 3.4, 10])

print("Netwon's polynomial")

print\_polynomial(X, Y)

print("Table of divided differences")

print\_divided\_diff\_table(X, Y)

print('Difference between function and interpolation polynomial')

print\_difference(X\_test, X, Y)

debug = True

if debug:

fig = plt.figure()

plt.plot(X, Y, "ob", markersize=5)

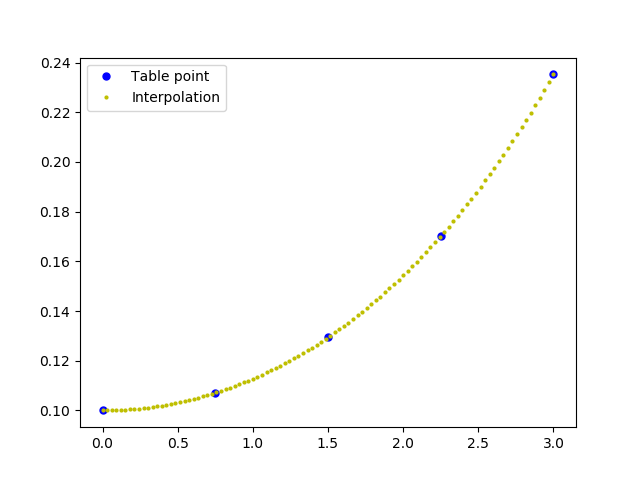
X\_interp = np.linspace(X[0], X[-1], 100)

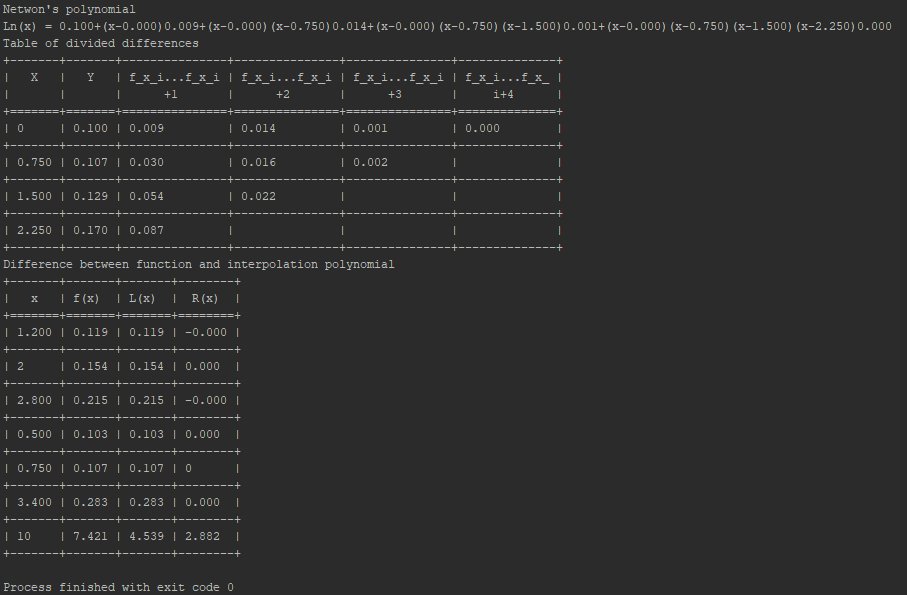
plt.plot(X\_interp, [L(x, X, Y) for x in X\_interp], 'oy', markersize=2)

plt.legend(["Table point", "Interpolation"])

plt.show()

**Результат роботи програми:**

****



**Висновок:** в ході виконання розрахункової роботи було освоєно дослідження похибки інтерполяції функцій многочленом Ньютона. Було виведено вигляд многочлена із коефіціентами з певною кількістю знаків після коми, задля спрощення вигляду многочлена Ньютона. Було виведено таблицю розділених різниць, комірки, що не мали значення відповіднї розділеної різниці, містили порожній рядок. Було виведено в таблицю, для наочності, похибку у точках спостереження, що відповідають точкам спостержень із завдання 2. Було навмисно обрано крайню точку спостерження x = 10 – точку поза межами відрізку інтерполяції, на якому було проведено спостереження, оскільки інтерполяційний многочлен давав нульову похибку. Похибка практичного характеру не перевищує значення аналітичного залишкового члена у розгляданих точках, що дає підгрунття для прийняття даної програми як правильної.