Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы численного анализа

**ОТЧЁТ**

к лабораторной работе

на тему

Интерполяционные многочлены

Выполнил: студент группы 153503

Кончик Денис Сергеевич

Проверил: Анисимов Владимир Яковлевич

Минск 2022

**Содержание**

[1. ЦЕЛЬ РАБОТЫ 3](#_Toc118582428)

[2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 4](#_Toc118582429)

[3. АЛГОРИТМ РЕШЕНИЯ 10](#_Toc118582430)

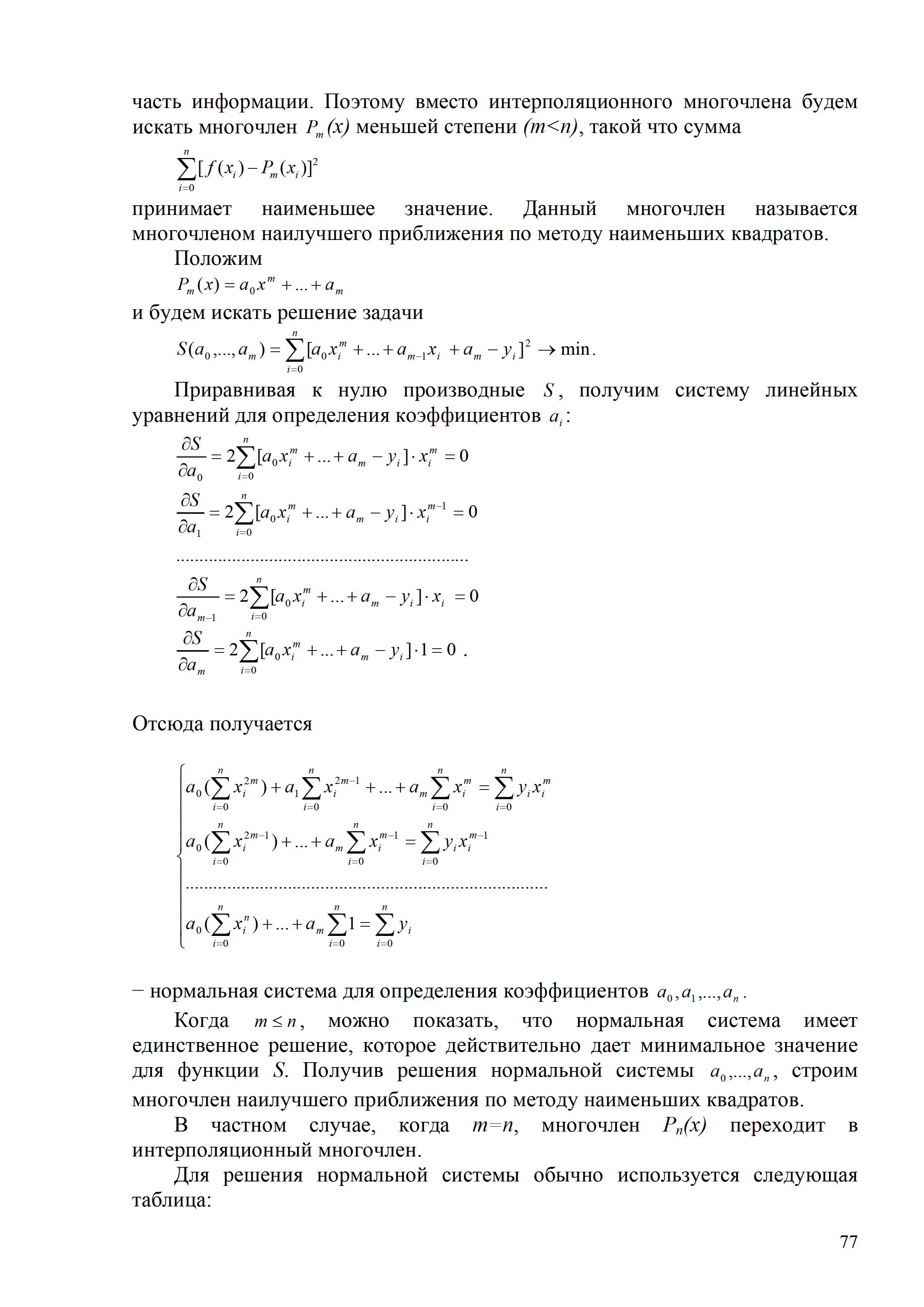
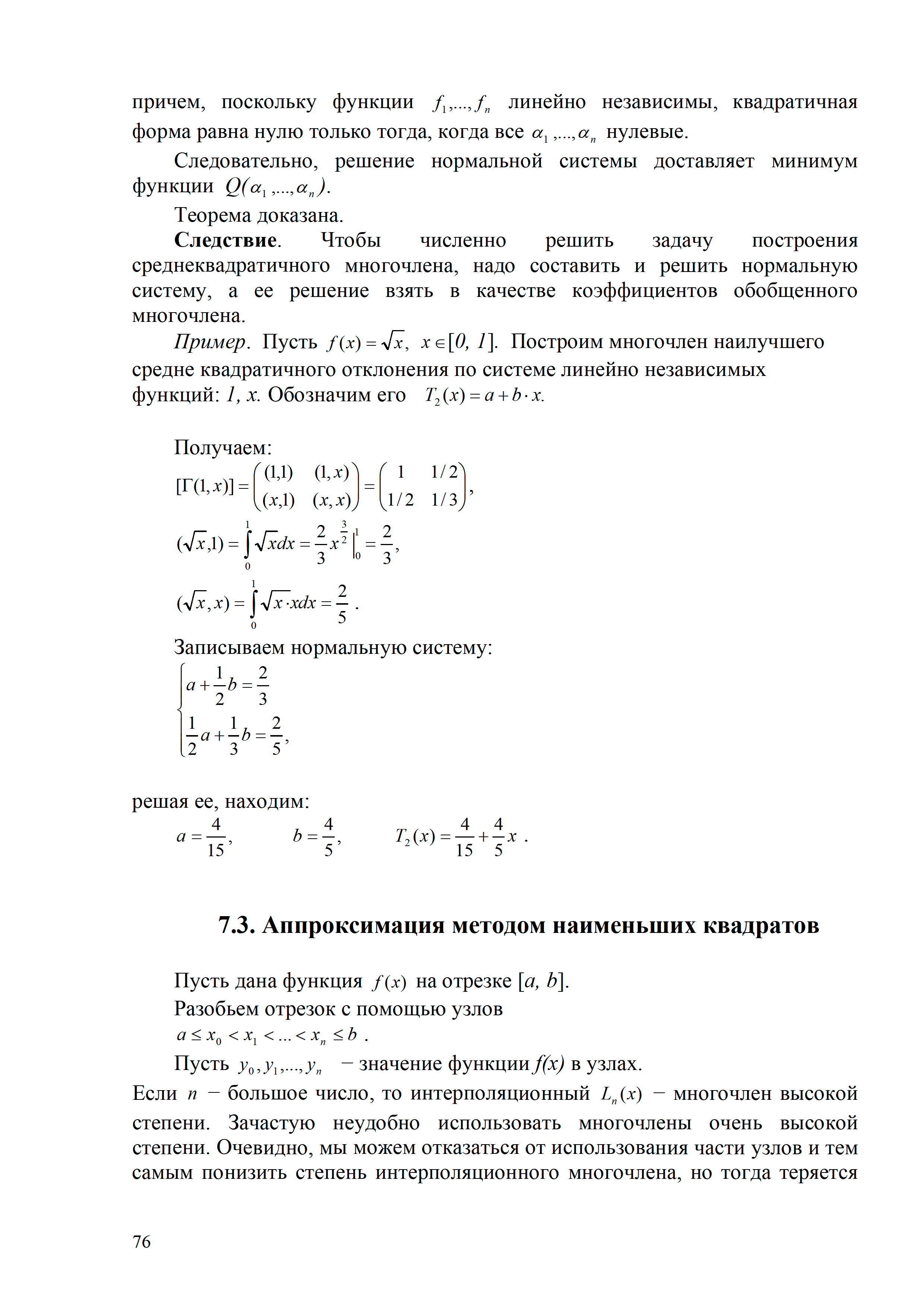
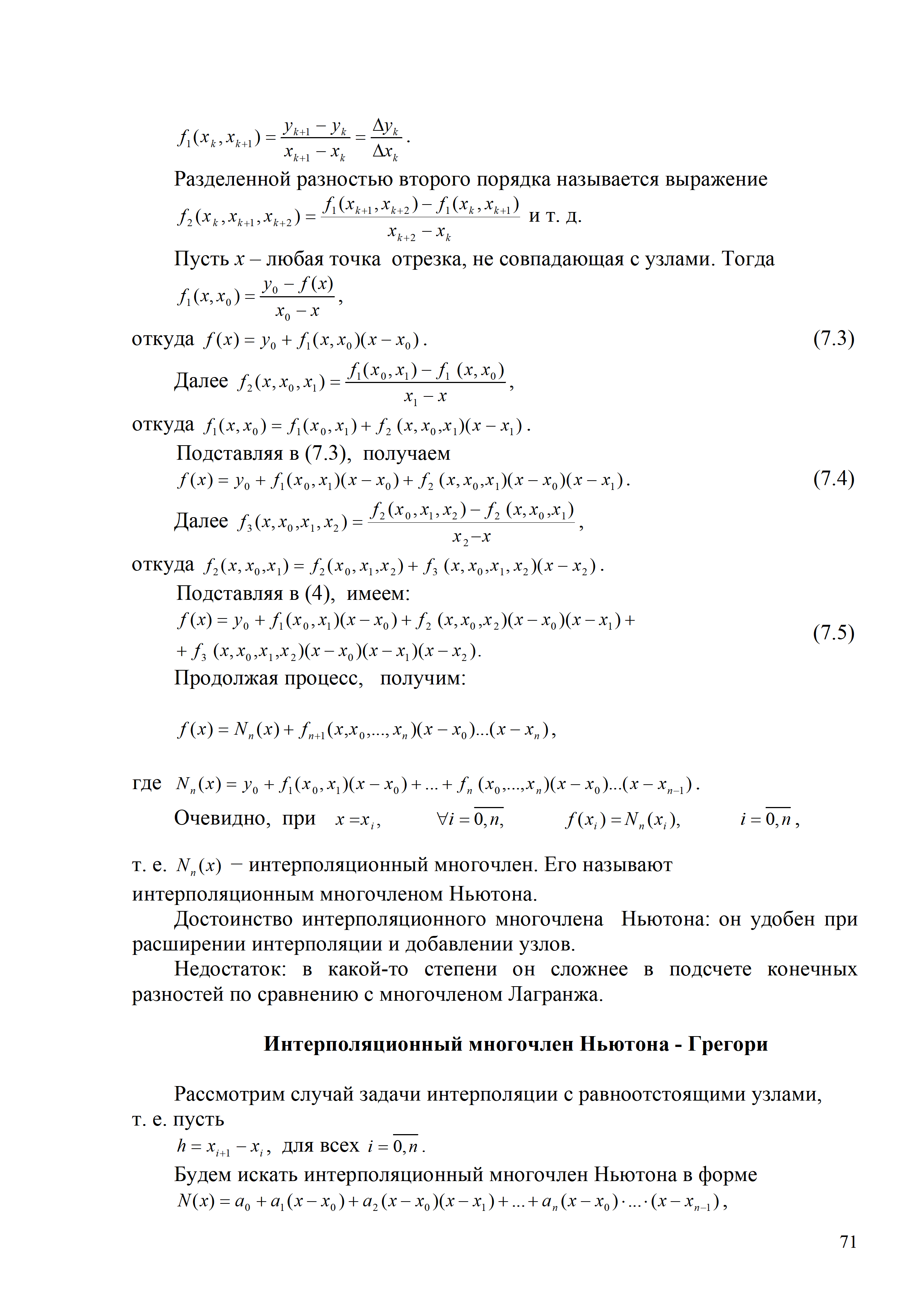
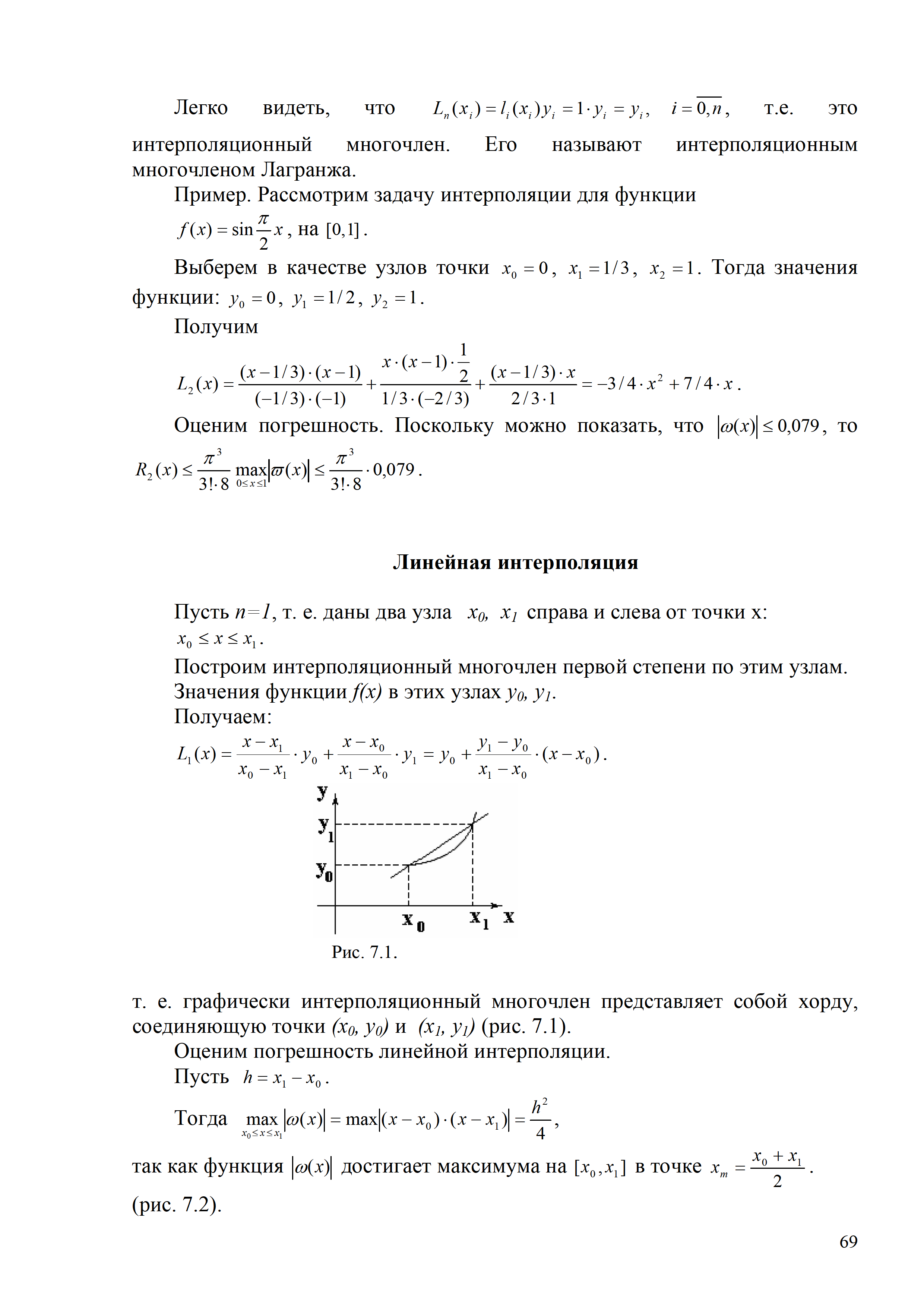
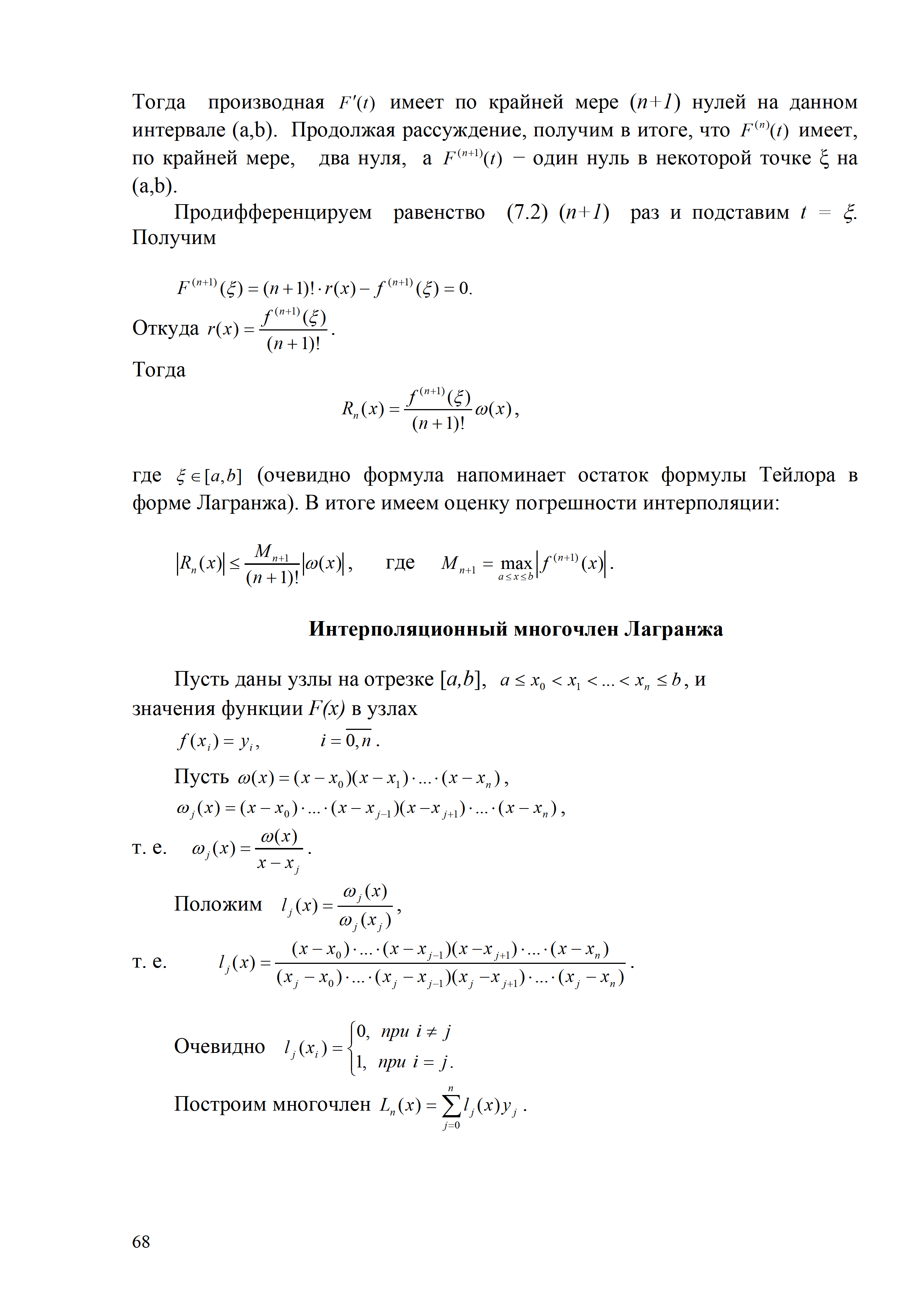
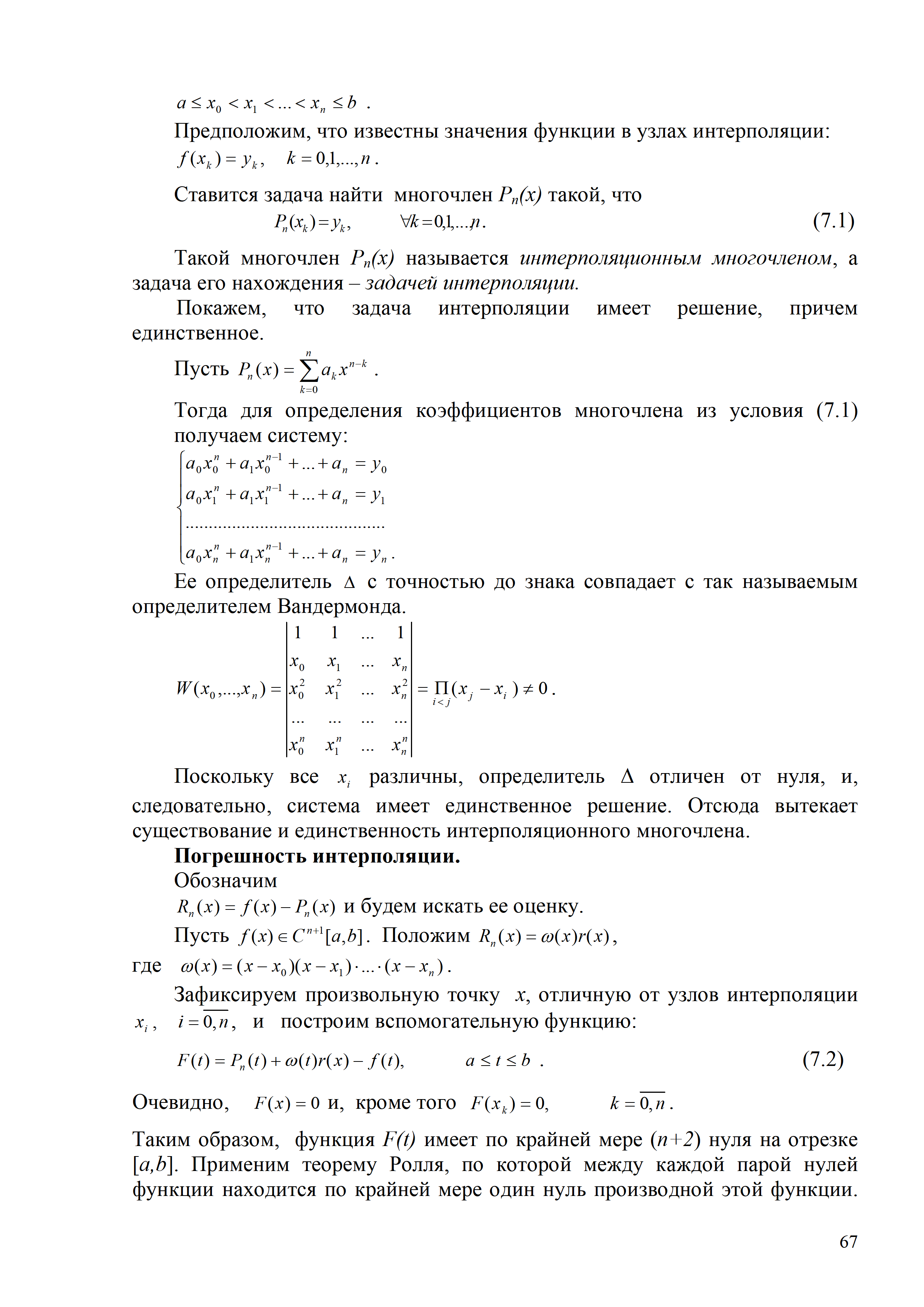
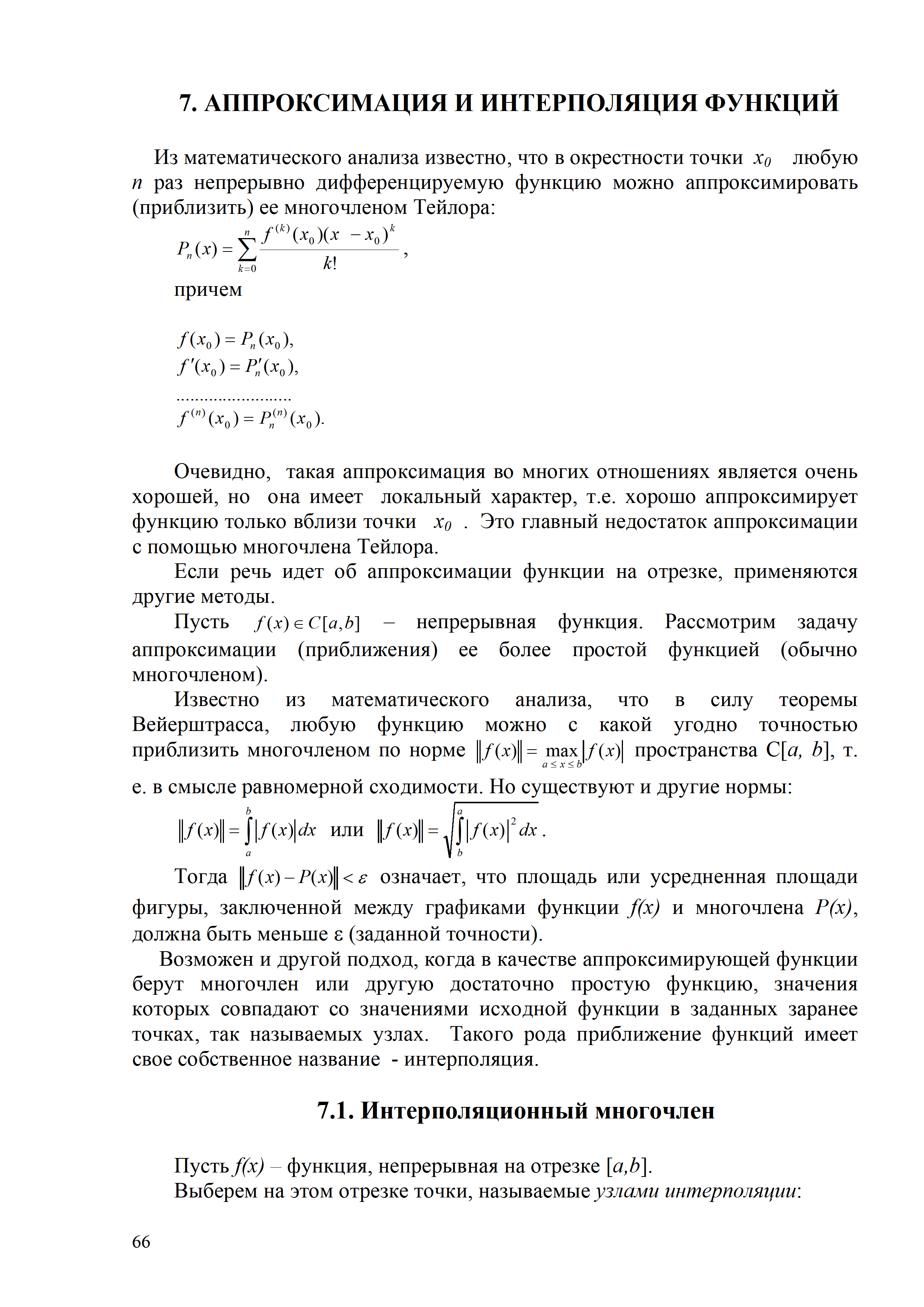
[4. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ 12](#_Toc118582431)

[5. ТЕСТОВЫЕ ПРИМЕРЫ 15](#_Toc118582432)

[6. ЗАДАНИЕ 17](#_Toc118582433)

[7. ВЫВОД 20](#_Toc118582434)

1. **ЦЕЛЬ РАБОТЫ**
2. Изучить интерполяцию функций с помощью интерполяционных многочленов Лагранжа и Ньютона, аппроксимацию функций методом наименьших квадратов.
3. Составить алгоритм и программу нахождения многочленов соответствующими методами.
4. Проверить правильность работы программы на тестовых примерах.
5. Решить задание заданного варианта.
6. **ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**



# **АЛГОРИТМ РЕШЕНИЯ**





# **ПРОГРАММНАЯ РЕАЛИЗАЦИЯ**

import numpy as np

import matplotlib.pyplot as plt

from numpy.core.multiarray import dot

def input():

x = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

p = [0.0, 0.41, 0.79, 1.13, 1.46, 1.76, 2.04, 2.3, 2.55, 2.79, 3.01]

k = 11

m = 2.53

y = [ (p[i] + ((-1) \*\* k) \* m) for i in range(len(x))]

dots = list(zip(x, y))

return dots

# Многочлен Лагранжа

def Lagrange(dots):

n = len(dots) # Число точек

(x, y) = map(list, zip(\*dots)) # Списки X и Y отдельно

polynom = np.poly1d([0]) # polynom = 0

for i in range(n):

p = np.poly1d([1]) # p = 1

for j in range(n):

if j != i: # пропускаем j-ое

p \*= np.poly1d([ 1, -x[j] ]) / (x[i] - x[j]) # p \*= (X-Xj)/(Xi-Xj)

polynom += y[i] \* p # polynom += P\*Yi

return polynom

# Разделенные разности

def DividedDifferences(x):

n = len(x)

diffs = [[None for j in range(n - i)] for i in range(n)]

for i in range(n):

diffs[i][0] = y[i]

for j in range(1, n):

for i in range(n - j):

diffs[i][j] = ((diffs[i + 1][j - 1] - diffs[i][j - 1]) / (x[i + j] - x[i]))

return diffs

# Многочлен Ньютона

def Newton(dots):

n = len(dots) # Число точек

(x, y) = map(list, zip(\*dots)) # Списки X и Y отдельно

diffs = DividedDifferences(x) # Разделенные разности

polynom = np.poly1d([0]) # polynom = 0

for i in range(n):

p = np.poly1d([1]) # p = 1

for j in range(i):

p \*= np.poly1d([ 1, -x[j] ]) # p \*= (X - Xj)

polynom += p \* diffs[0][i] # polynom += p \* fn(x0,..,xi)

return polyno

# МНК при m == n

def Simple(dots):

n = len(dots)

(x, y) = map(list, zip(\*dots))

A = []

for i in range(n):

A.append([])

for j in range(n):

A[i].append(x[i] \*\* j)

polynom = np.poly1d(np.linalg.solve(A, y)[::-1])

return polynom

# Аппроксимация методом наименьших квадратов

def Squares(dots, m = None):

n = len(dots) - 1

if m is None:

m = n

assert 0 <= m <= n

if m == n:

return Simple(dots)

(x, y) = map(list, zip(\*dots))

b = []

for k in range(m + 1):

s = 0

for i in range(n + 1):

s += y[i] \* (x[i] \*\* (m - k))

b.append(s)

A = []

for k in range(m + 1):

A.append([])

for j in range(m + 1):

s = 0

for i in range(n + 1):

s += x[i] \*\* (2 \* m - k - j)

A[k].append(s)

polynom = np.poly1d(np.linalg.solve(A, b))

return polynom

# Общая погрешность

def get\_error(method, dots):

func = method(dots) - Squares(dots, 10)

der = np.polyder(func)

max\_error = 0.0

for root in np.roots(der):

if x[0] <= root <= x[-1]:

max\_error = max(max\_error, abs(np.polyval(func, root)))

return max\_error

dots = input()

(x, y) = map(list, zip(\*dots))

print("(x,y) =", dots, '\n')

lagrange = Lagrange(dots)

print("Полином Лагранжа =")

print(lagrange, '\n')

newton = Newton(dots)

print("Полином Ньютона =")

print(newton, '\n')

squares = Squares(dots)

print("Полином МНК =")

print(squares, '\n')

xdot = 0.47

width = 25

print(f"Полином Лагранжа ({xdot}) =".ljust(width), lagrange(xdot))

print(f"Полином Ньютона ({xdot}) =".ljust(width), newton(xdot))

print(f"Молином МНК ({xdot}) =".ljust(width), squares(xdot))

#print(f"Молином МНК ({xdot}) =", "{:.4f}".format(squares(xdot)))

print("|Лагранж - Ньютон| =".ljust(width), abs(lagrange(xdot) - newton(xdot)))

print("|Лагранж - МНК| =".ljust(width), abs(lagrange(xdot) - squares(xdot)))

print("|Ньютон - МНК| =".ljust(width), abs(newton(xdot) - squares(xdot)))

print("Погрешность Лагранж: ".ljust(width), get\_error(Lagrange, dots))

print("Погрешность Ньютон: ".ljust(width), get\_error(Newton, dots))

#print(f"Inaccuracy({xdot}) = ", Inaccuracy(x, xdot))

plotdots = 10\*\*4

plt.plot(x, y, 'og',linewidth=5)

xplot = np.linspace(min(x), max(x), plotdots)

yplot = [squares(xdot) for xdot in xplot]

plt.plot(xplot, yplot, 'r',linewidth=4) # Красный

yplot = [lagrange(xdot) for xdot in xplot]

plt.plot(xplot, yplot, 'b--',linewidth=4) # Пунктир голубой

yplot = [newton(xdot) for xdot in xplot]

plt.plot(xplot, yplot, 'm:', linewidth=4) # Фиолетовый точками

plt.show()

**5. ТЕСТОВЫЕ ПРИМЕРЫ**

Построить интерполяционные многочлены в формах Лагранжа1 и Ньютона2, построить многочлен наилучшего приближения по методу наименьших квадратов3.

Цвета: 1 – голубой, 2 – фиолетовый, 3 – красный.

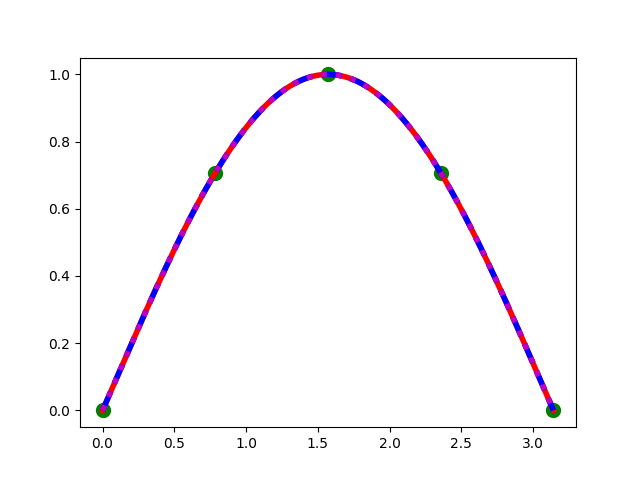
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Тестовый пример 1**   |  |  |  | | --- | --- | --- | | *x* | *0* | *5* | | *y* | *1* | *5* |   Вывод программы: | **Тестовый пример 2**   |  |  |  |  | | --- | --- | --- | --- | | *x* | *0* | *1* | *3* | | *y* | *-1* | *-2* | *2* |   Вывод программы: |
| **Тестовый пример 3**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | *x* | *-5* | *-3* | *2* | *7* | *8* | | *y* | *8* | *7* | *-2* | *-3* | *5* |   Вывод программы: | **Тестовый пример 4**   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | *x* | *-9* | *-7* | *-4* | *0* | *2* | *7* | *4* | | *y* | *-3* | *2* | *1* | *9* | *15* | *9* | *2* |   Вывод программы: |

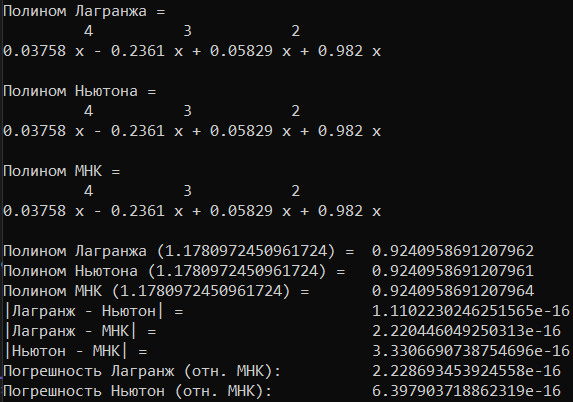
**Тестовый пример 5**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x* | *0* | *π/4* | *π /2* | *3 π/4* | *π* |
|  | | | | | |

Построить многочлены в формах Лагранжа и Ньютона, многочлен наилучшего приближения. Найти значение многочлена в точке .

Вывод программы:



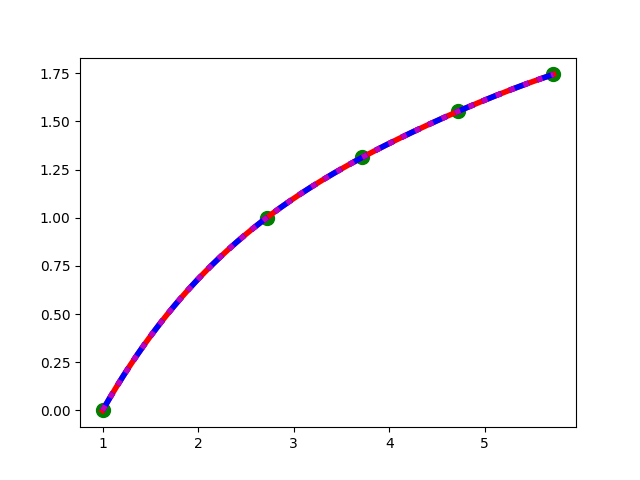


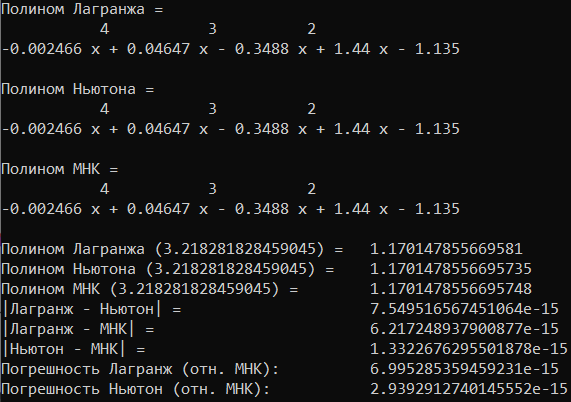
**Тестовый пример 6**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *x* | *1* | *e* | *e+1* | *e+2* | *e+3* |
|  | | | | | |

Построить многочлены в формах Лагранжа и Ньютона, многочлен наилучшего приближения. Найти значение многочлена в точке .

Вывод программы:





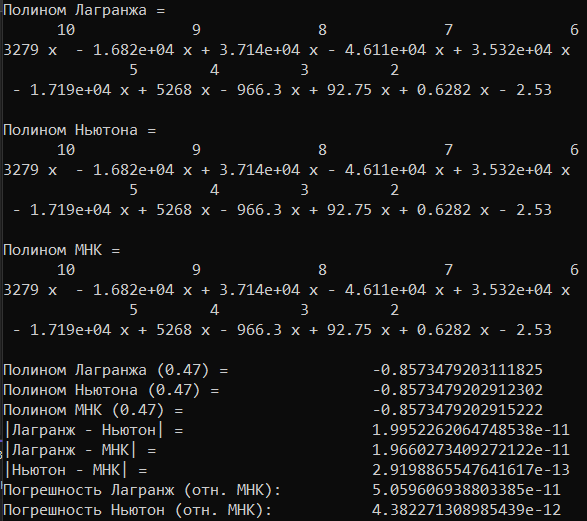
6. ЗАДАНИЕ

Вариант 11

Построить интерполяционные многочлены в форме Лагранжа1 и Ньютона2. Оценить погрешность. Вычислить значение функции в точке 0.47 с помощью интерполяционного многочлена и многочлена наилучшего приближения3. Сравнить значения.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *x* | *0.0* | *0.1* | *0.2* | *0.3* | *0.4* | *0.5* | *0.6* | *0.7* | *0.8* | *0.9* | *1.0* |
| *y* | *-2.53* | *-2.12* | *-1.74* | *-1.40* | *-1.07* | *-0.77* | *-0.49* | *-0.23* | *0.02* | *0.26* | *0.48* |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Значение в точке x = 0.47 | | | | | | | |
| Полином Лагранжа | | | | Полином Ньютона | | | |
| ≈ - 0.8573 | | | | ≈ - 0.8573 | | | |
| Многочлен наилучшего приближения | | | | | | | |
| Степень | 1 | 2 | 3 | | 5 | 8 | 10 |
| Значение ≈ | -0.9612 | -0.8611 | -0.8587 | | -0.8596 | -0.8570 | -0.8573 |
| |МНП – Лагранж| ≈ | 0.1039 | 0.0038 | 0.0014 | | 0.0023 | 0.0003 | 0.0000 |
| |МНП –Ньютон| ≈ | 0.1039 | 0.0038 | 0.0014 | | 0.0023 | 0.0003 | 0.0000 |
| task Цвета: 1 – голубой, 2 – фиолетовый, 3 – красный | | | | | | | |

Вывод программы: 

# **7. ВЫВОД**

Таким образом, в ходе выполнения лабораторной работы была изучена интерполяция функций с помощью интерполяционных многочленов Лагранжа и Ньютона, аппроксимация методом наименьших квадратов. Составлен алгоритм и программа нахождения многочленов соответствующими методами, проверена правильность работы на тестовых примерах. Согласно заданному варианту построены интерполяционные многочлены в форме Лагранжа и Ньютона, построен многочлен наилучшего приближения по методу наименьших квадратов, вычислено значение функции в точке.