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

Численное дифференцирование

и интегрирование функций

Выполнил: студент группы 153503

Кончик Денис Сергеевич

Проверил: Анисимов Владимир Яковлевич

Минск 2022

**Содержание**

[1. ЦЕЛЬ РАБОТЫ 3](#_Toc120059950)

[2. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ 3](#_Toc120059951)

[3. АЛГОРИТМ РЕШЕНИЯ 10](#_Toc120059952)

[4. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ 11](#_Toc120059953)

[5. ТЕСТОВЫЕ ПРИМЕРЫ 14](#_Toc120059954)

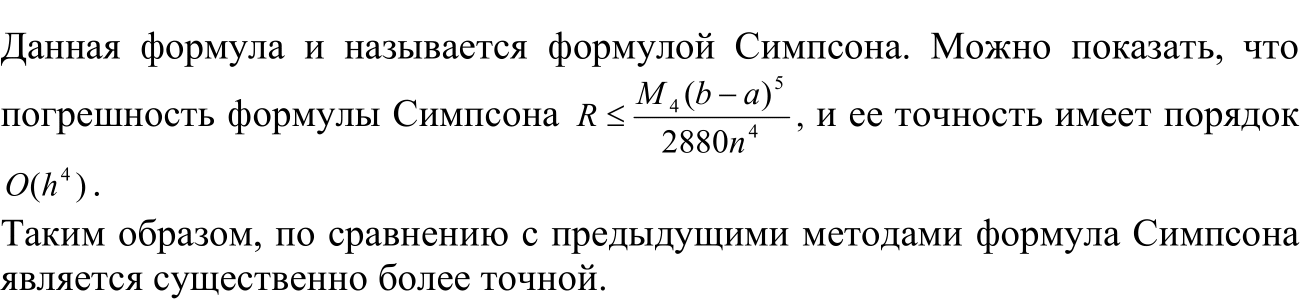
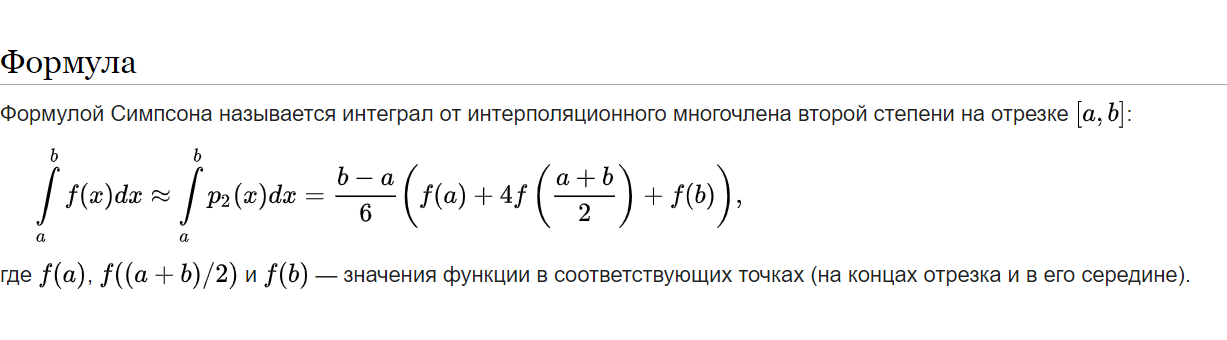
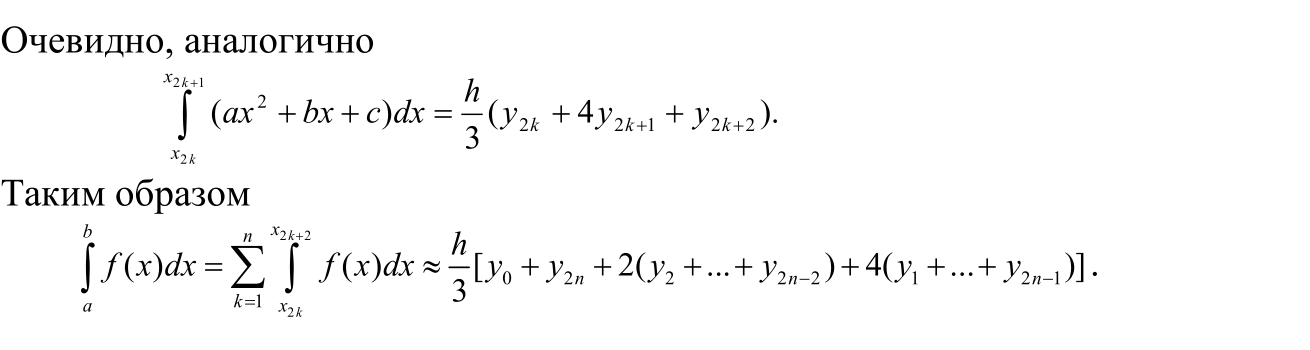
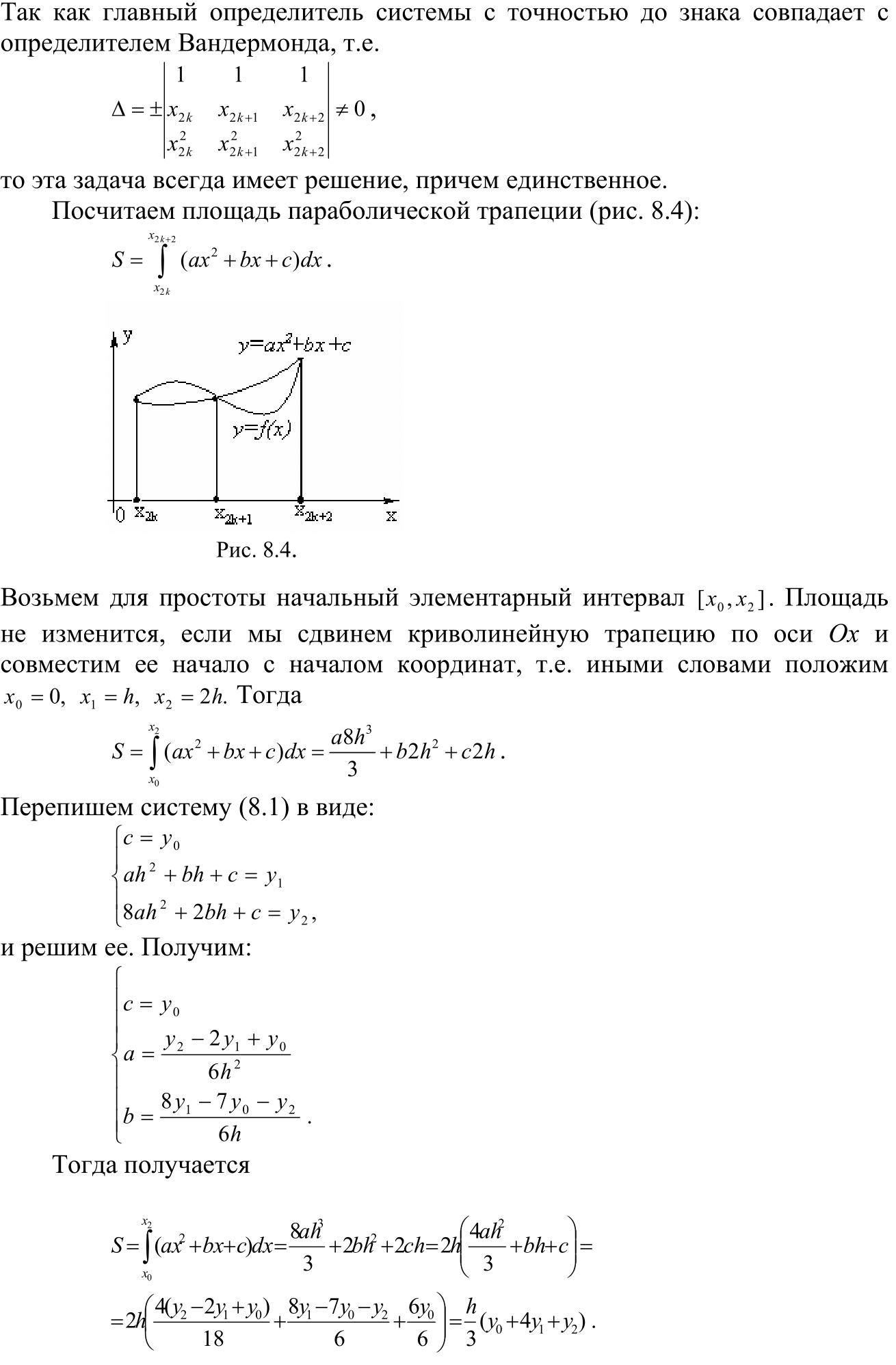
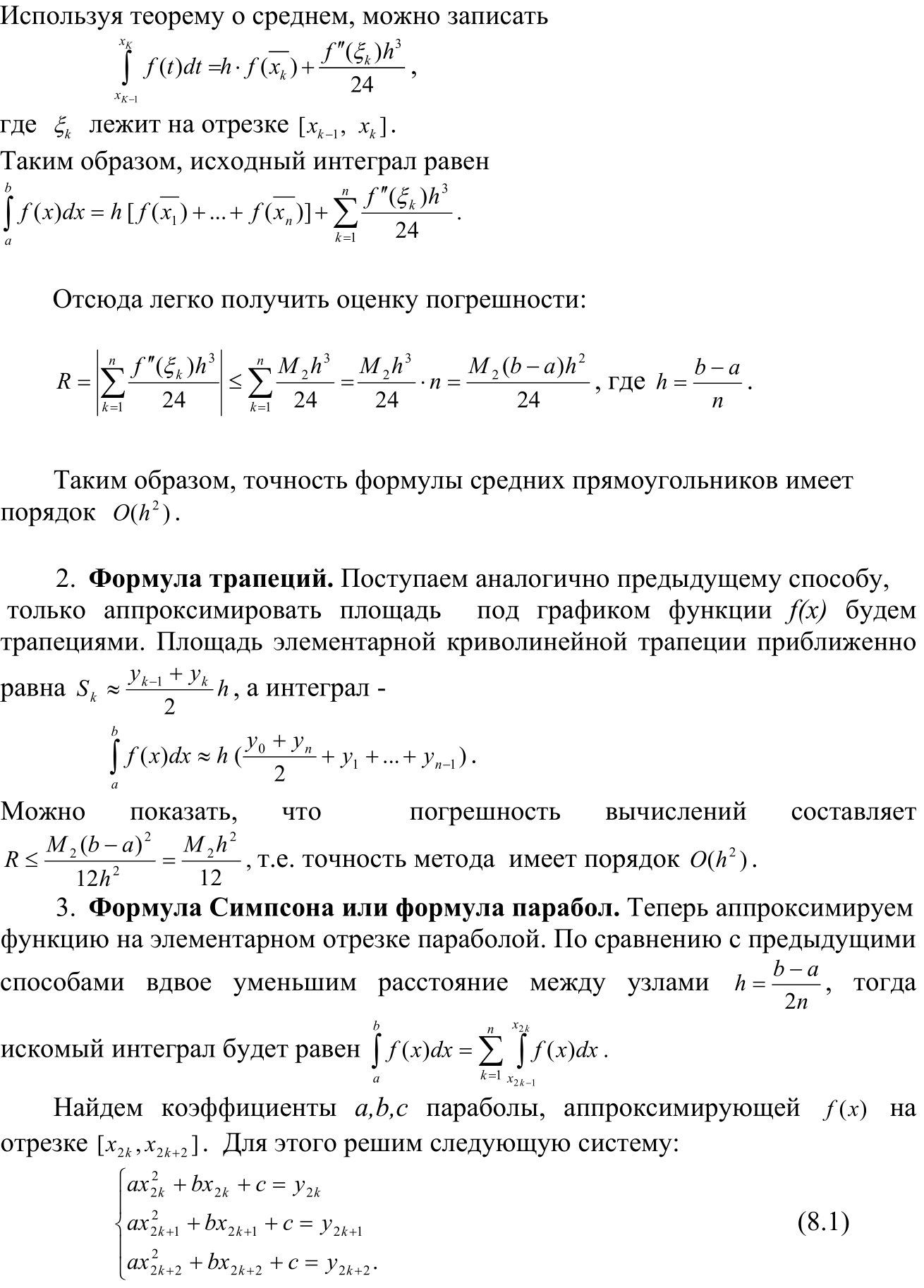
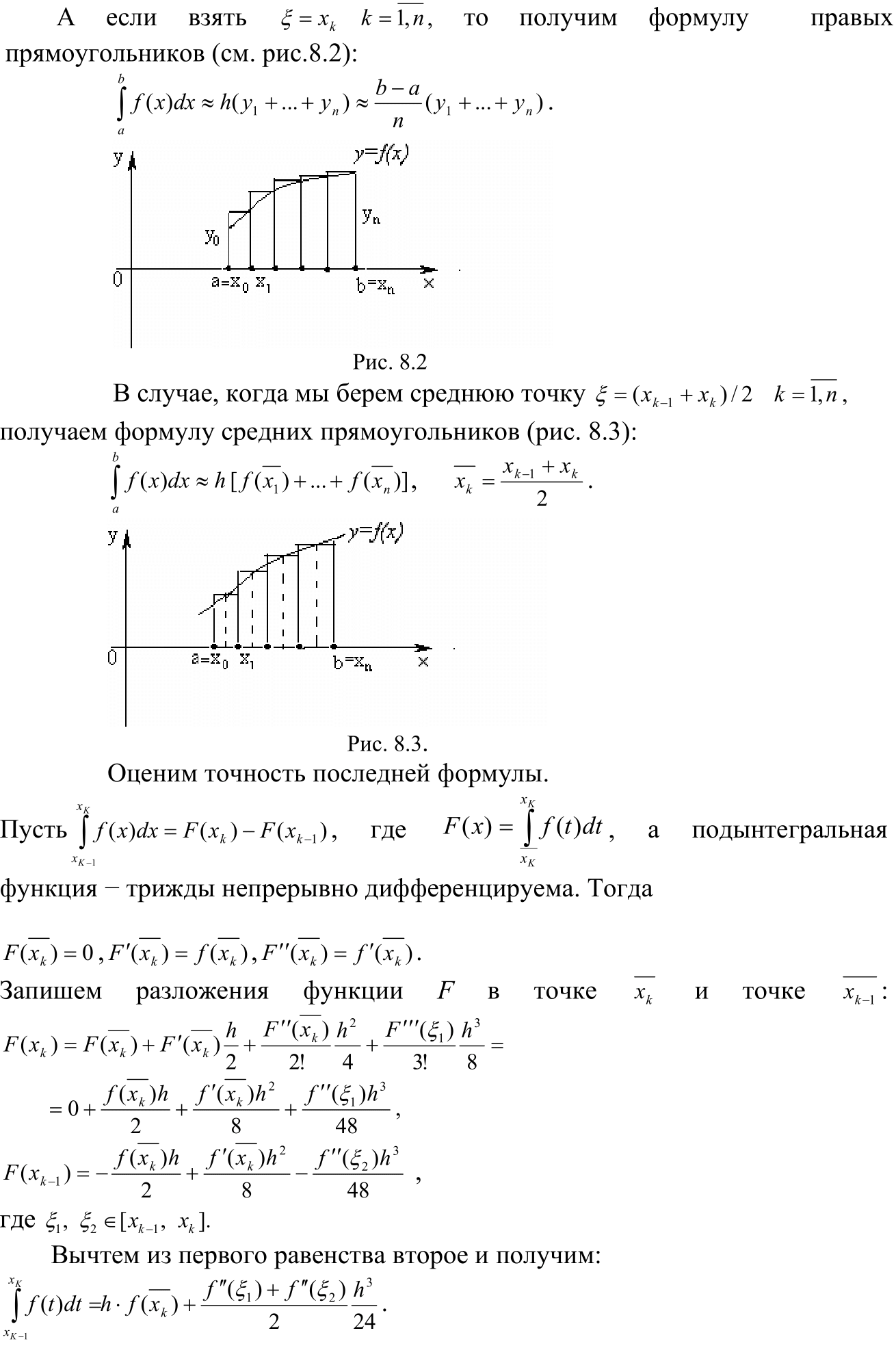
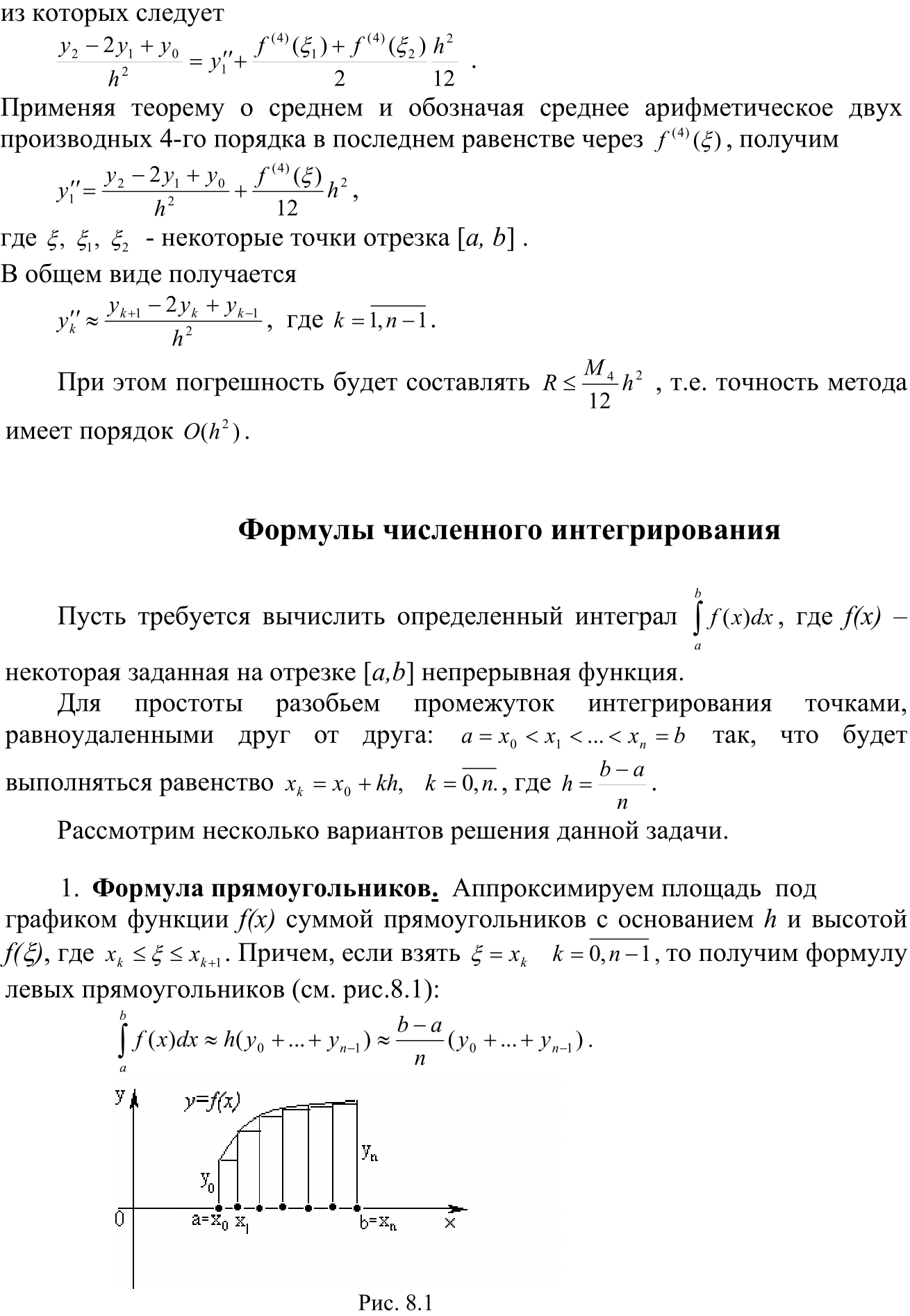
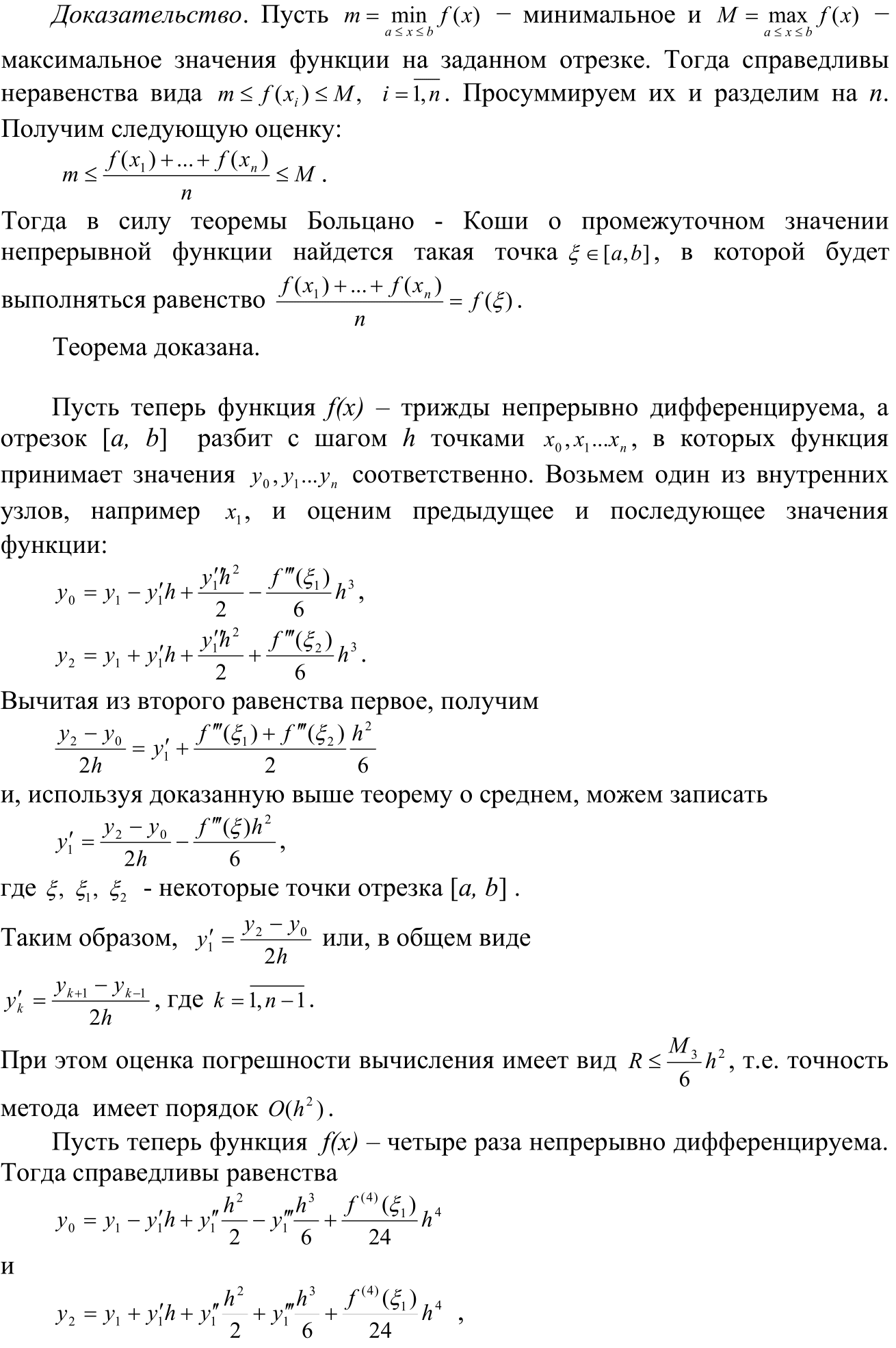
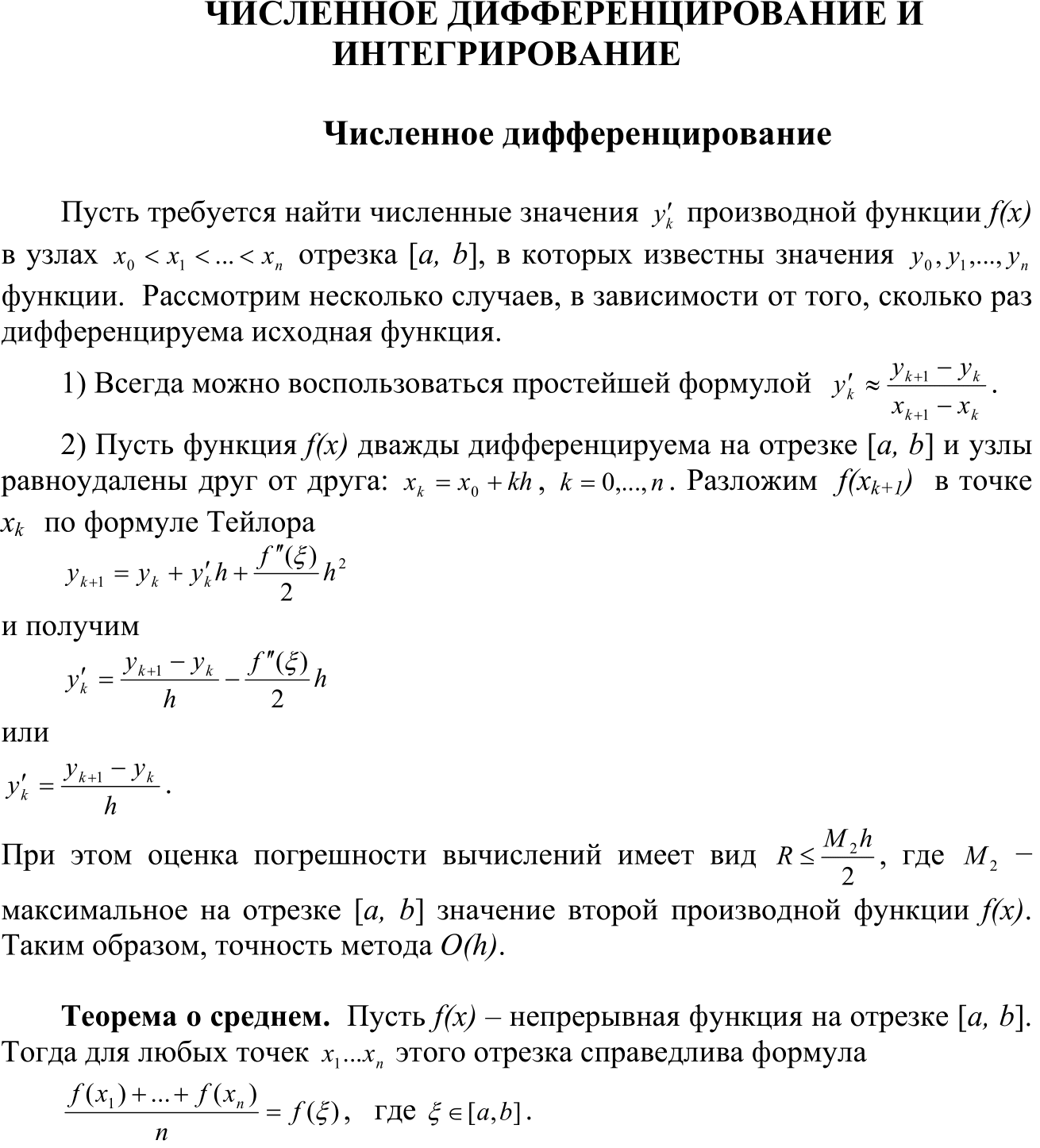
[6. ЗАДАНИЕ 15](#_Toc120059955)

[7. ВЫВОД 16](#_Toc120059956)

1. **ЦЕЛЬ РАБОТЫ**

Изучить методы численного вычисления производных и методы численного интегрирования. Сравнить методы по трудоёмкости, точности. Выполнить тестовое задание по численному дифференцированию и интегрированию.

1. **Теоретические сведения**



# **АЛГОРИТМ РЕШЕНИЯ**



# **ПРОГРАММНАЯ РЕАЛИЗАЦИЯ**

import sympy

import math

import numpy

xSymbol = sympy.symbols('x')

func = sympy.log(xSymbol, 10)

L, R = 1, 3

xDot = (L + R) / 2

derFormat = "{:.7f}" # Формат вывода чисел в производных

intFormat = "{:.7f}" # Формат вывода чисел в интегралах

derTextWidth = 40 # Ширина вывода текста в производных

intTextWidth = 45 # Ширина вывода текста в интегралах

derEps = 0.000001 # Точность в производных

intEps = 0.000001 # Точность в интегралах

def f(x):

return func.subs(xSymbol, x)

def fd(x, n):

return (sympy.diff(func, xSymbol, n)).subs(xSymbol, x)

def DerFirst1(x):

fd2 = sympy.diff(func, xSymbol, 2); # Производная второго порядка

M2 = abs(sympy.calculus.util.maximum(fd2, xSymbol, sympy.Interval(L, R))) # M2

h = abs(2 \* derEps / M2) # Шаг

return (f(x + h) - f(x)) / h

def DerFirst2(x):

fd3 = sympy.diff(func, xSymbol, 3); # Производная третьего порядка

M3 = abs(sympy.calculus.util.maximum(fd3, xSymbol, sympy.Interval(L, R))) # M3

h = math.sqrt((6 \* derEps) / M3)

return (f(x + h) - f(x - h)) / (2 \* h)

def DerSecond(x):

fd4 = sympy.diff(func, xSymbol, 4); # Производная четвертого порядка

M4 = abs(sympy.calculus.util.maximum(fd4, xSymbol, sympy.Interval(L, R))) # M4

h = math.sqrt((12 \* derEps) / M4)

return (f(x + h) - 2 \* f(x) + f(x - h)) / (h \*\* 2)

def IntCalculator():

return sympy.integrate(func, (xSymbol, L, R))

def IntMiddleRectangles():

fd2 = sympy.diff(func, xSymbol, 2); # Производная второго порядка

M2 = abs(sympy.calculus.util.maximum(fd2, xSymbol, sympy.Interval(L, R))) # M2

h = ((24\*intEps)/(M2)) \*\* (1. / 3.)

n = numpy.ceil((R - L) / h)

S = 0.0

for k in range(1, int(n)):

S += f((2 \* L + 2 \* k \* h - h) / 2.0)

S \*= h

return S

def IntLeftRectangles():

fd1 = sympy.diff(func, xSymbol, 1); # Производная первого порядка

M1 = abs(sympy.calculus.util.maximum(fd1, xSymbol, sympy.Interval(L, R))) # M1

h = math.sqrt((2\*intEps)/(M1))

n = numpy.ceil((R - L) / h)

S = 0.0

for k in range(0, int(n - 1)):

S += f(L + k \* h)

S \*= h

return S

def IntRightRectangles():

fd1= sympy.diff(func, xSymbol, 1); # Производная первого порядка

M1 = abs(sympy.calculus.util.maximum(fd1, xSymbol, sympy.Interval(L, R))) # M1

h = math.sqrt((2\*intEps)/(M1))

n = numpy.ceil((R - L) / h)

S = 0.0

for k in range(1, int(n)):

S += f(L + k \* h)

S \*= h

return S

def IntTrapezoid():

fd2 = sympy.diff(func, xSymbol, 2); # Производная второго порядка

M2 = abs(sympy.calculus.util.maximum(fd2, xSymbol, sympy.Interval(L, R))) # M2

h = math.sqrt((24 \* intEps) / (M2 \* (R - L)))

n = numpy.ceil((R - L) / h)

S = 0.0

for k in range(1, int(n - 1)):

S += f(L + k \* h)

S += (f(L) + f(L + n \* h)) / 2.0

S \*= h

return S

def IntSimpson():

fd4 = sympy.diff(func, xSymbol, 4); # Производная четвертого порядка

M4 = abs(sympy.calculus.util.maximum(fd4, xSymbol, sympy.Interval(L, R))) # M4

n = (M4 \* (R - L)\*\*5 / 2880 / intEps) \*\* (1. / 4.)

h = (R - L) / (2 \* n)

n = numpy.ceil(n)

S = f(L) + f(L + (2\*n) \* h)

add = 0

for k in range(2, 2\*n-2):

add += f(L + k \* h)

S += 2 \* add

add = 0

for k in range(1, 2\*n-1):

add += f(L + k \* h)

S += 4 \* add

S = S \* h / 3

return ((R - L) / 6) \* (f(L) + 4 \* f(((L + R) / 2)) + f(R))

return S

print(f"Функция: y = {func}")

print(f"Интервал: [{L}, {R}]")

print(f"xDot = {xDot}")

print("\n")

print("Первая производная (калькулятор):".ljust(derTextWidth), derFormat.format(fd(xDot, 1)))

print("Первая производная (метод #1: O(h)): ".ljust(derTextWidth), derFormat.format(DerFirst1(xDot)))

print("delta =", intFormat.format(abs(fd(xDot, 1) - DerFirst1(xDot))))

print("Первая производная (метод #2: O(h^2)): ".ljust(derTextWidth), derFormat.format(DerFirst2(xDot)))

print("delta =", intFormat.format(abs(fd(xDot, 1) - DerFirst2(xDot))))

print("\n")

print("Вторая производная (калькулятор):".ljust(derTextWidth), derFormat.format(fd(xDot, 2)))

print("Вторая производная (O(h^2)): ".ljust(derTextWidth), derFormat.format(DerSecond(xDot)))

print("delta =", intFormat.format(abs(fd(xDot, 2) - DerSecond(xDot))))

print("\n")

print("Интеграл (калькулятор): ".ljust(intTextWidth), intFormat.format(IntCalculator()))

print("Интеграл (средних прямоугольников: O(h^2)): ".ljust(intTextWidth), intFormat.format(IntMiddleRectangles()))

print("delta =", intFormat.format(abs(IntCalculator() - IntMiddleRectangles())))

print("Интеграл (левых прямоугольников: O(h^2)): ".ljust(intTextWidth), intFormat.format(IntLeftRectangles()))

print("delta =", intFormat.format(abs(IntCalculator() - IntLeftRectangles())))

print("Интеграл (правых прямоугольников: O(h^2)): ".ljust(intTextWidth), intFormat.format(IntRightRectangles()))

print("delta =", intFormat.format(abs(IntCalculator() - IntRightRectangles())))

print("Интеграл (трапеций: O(h^2)): ".ljust(intTextWidth), intFormat.format(IntTrapezoid()))

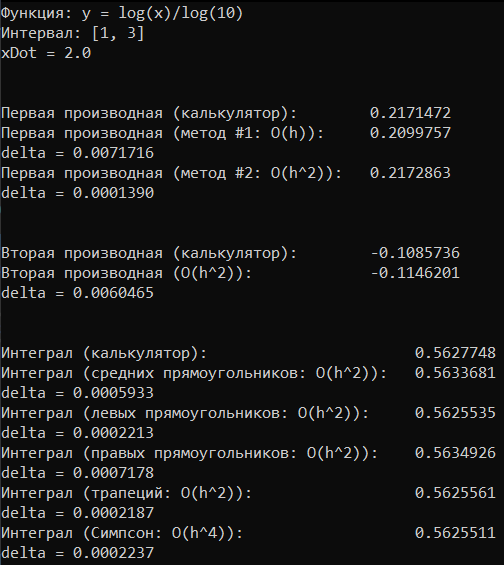
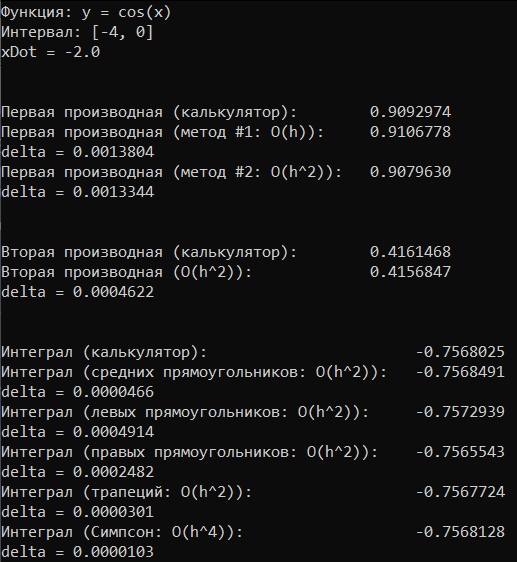
print("delta =", intFormat.format(abs(IntCalculator() - IntTrapezoid())))

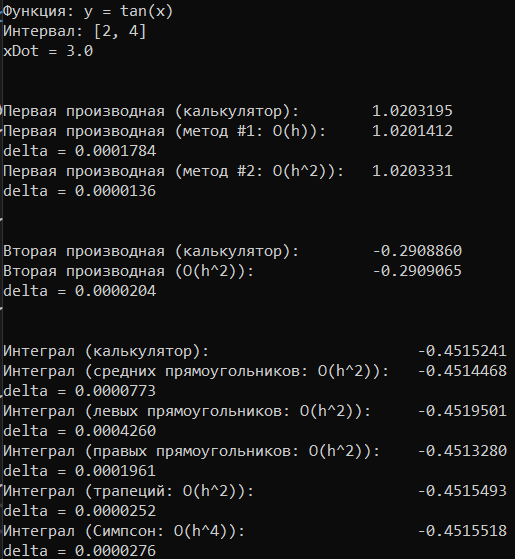
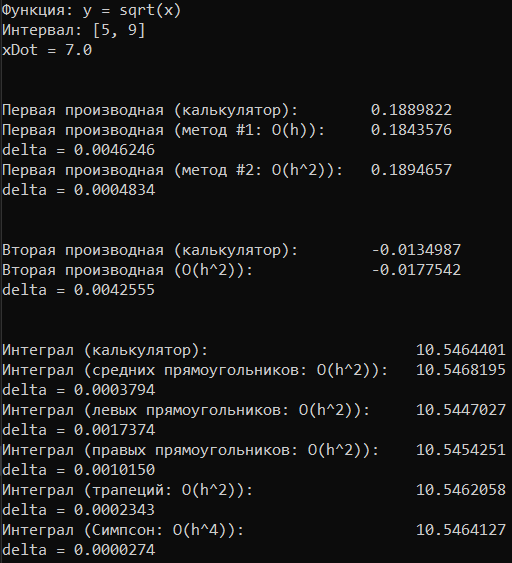
print("Интеграл (Симпсон: O(h^4)): ".ljust(intTextWidth), intFormat.format(IntSimpson()))

print("delta =", intFormat.format(abs(IntCalculator() - IntSimpson())))

**5. ТЕСТОВЫЕ ПРИМЕРЫ**

Найти численное значение первой и второй производной в точке, являющейся серединой заданного интервала. Вычислить с точностью интегралы по формулам прямоугольников, трапеций, Симпсона.

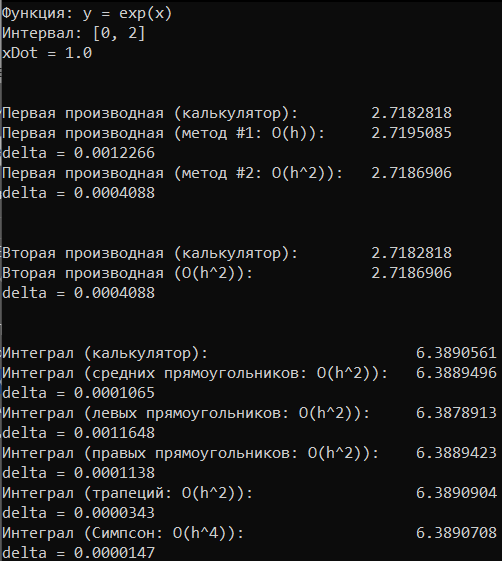
Тестовый пример 1 Тестовый пример 2

Тестовый пример 3 Тестовый пример 4

6. ЗАДАНИЕ

Вариант 1

Найти численное значение первой и второй производной в точке, являющейся серединой заданного интервала, с точностью до 0.01. Вычислить с точностью 0.001 интегралы по формулам прямоугольников, трапеций, Симпсона. Сравнить методы по точности.



# **7. ВЫВОД**

Таким образом, в ходе выполнения лабораторной работы были изучены методы численного интегрирования и дифференцирования функций. Составлен алгоритм и программа интегрирования и дифференцирования, проверена правильность работы на тестовых примерах. Для функции заданного варианта найдено численное значение первой и второй производной в точке, вычислены с заданной точностью интегралы по формулам прямоугольников, трапеций, Симпсона.