Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

Отчет по лабораторной работе №8

Численное дифференцирование и интегрирование функций

Выполнил:

студент группы 153503

Вергасов В.М.

Руководитель:

доцент

Анисимов В.Я.

Минск 2022

**Содержание**

[**1. Цель работы** 3](#_Toc118822958)

[**2. Теоретические сведения** 3](#_Toc118822959)

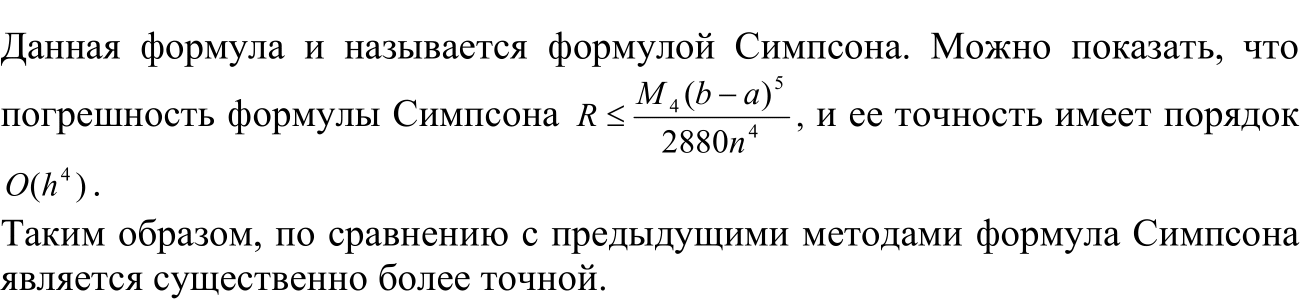
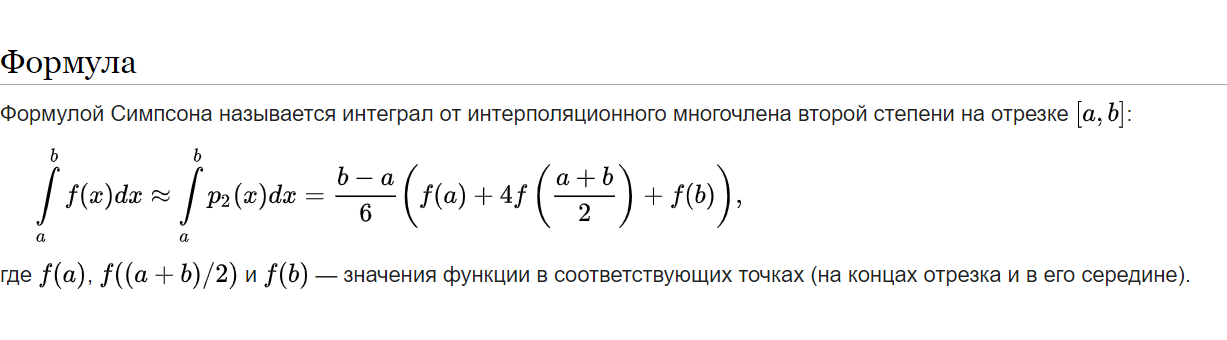
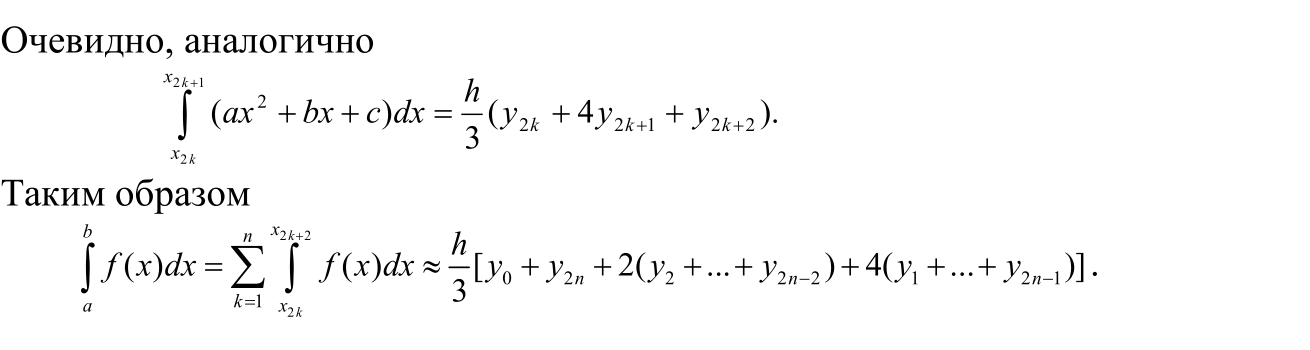
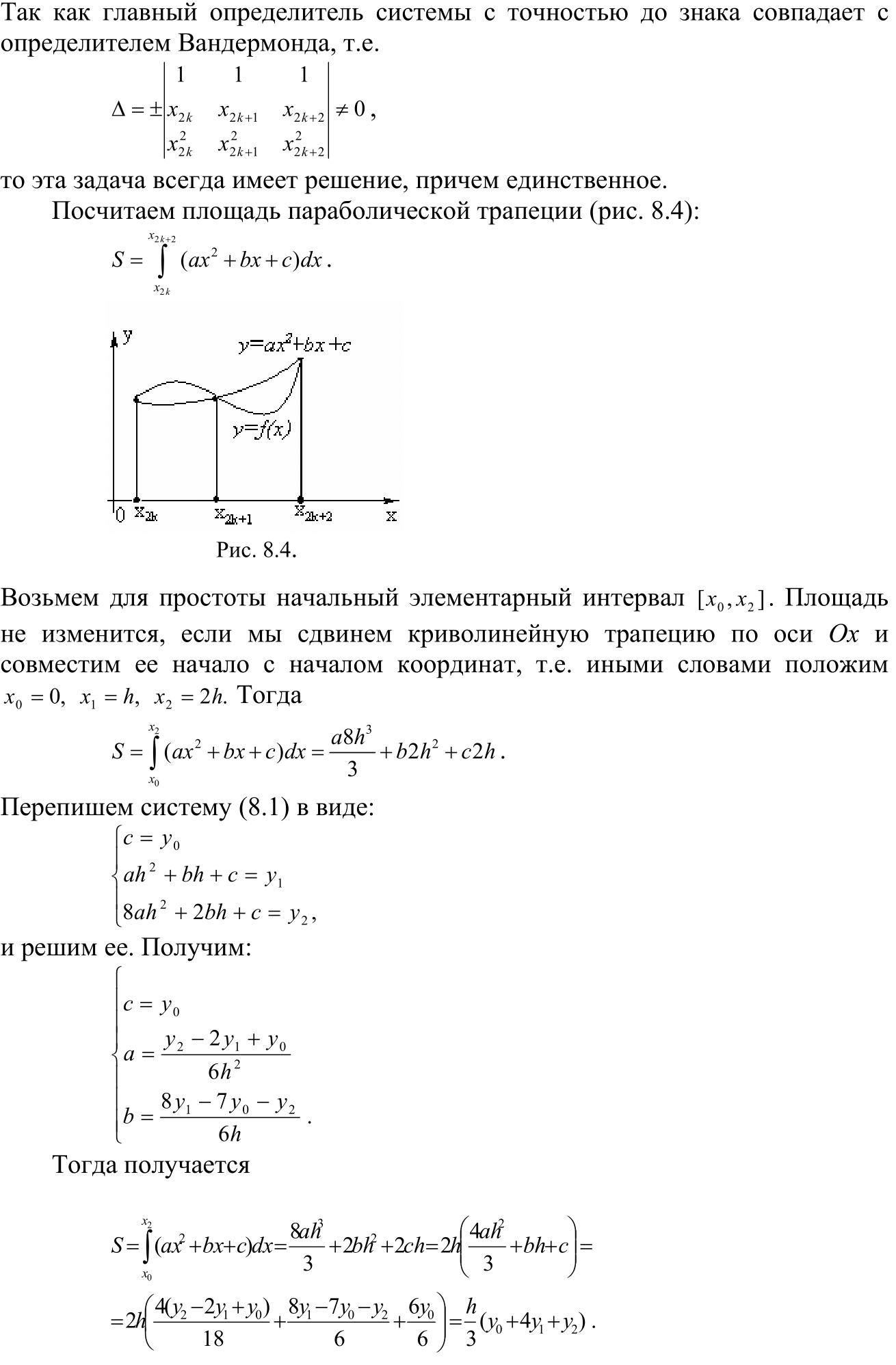
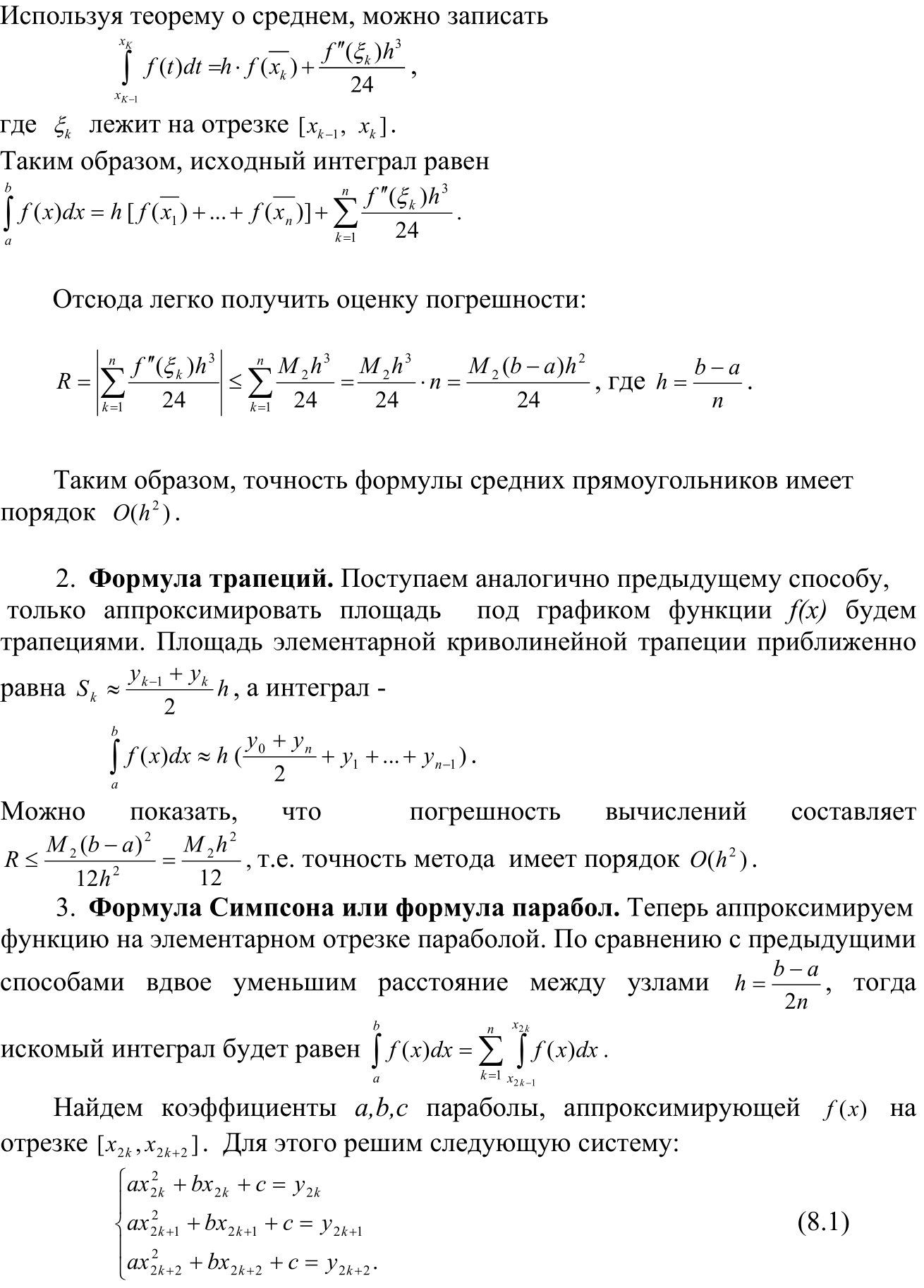
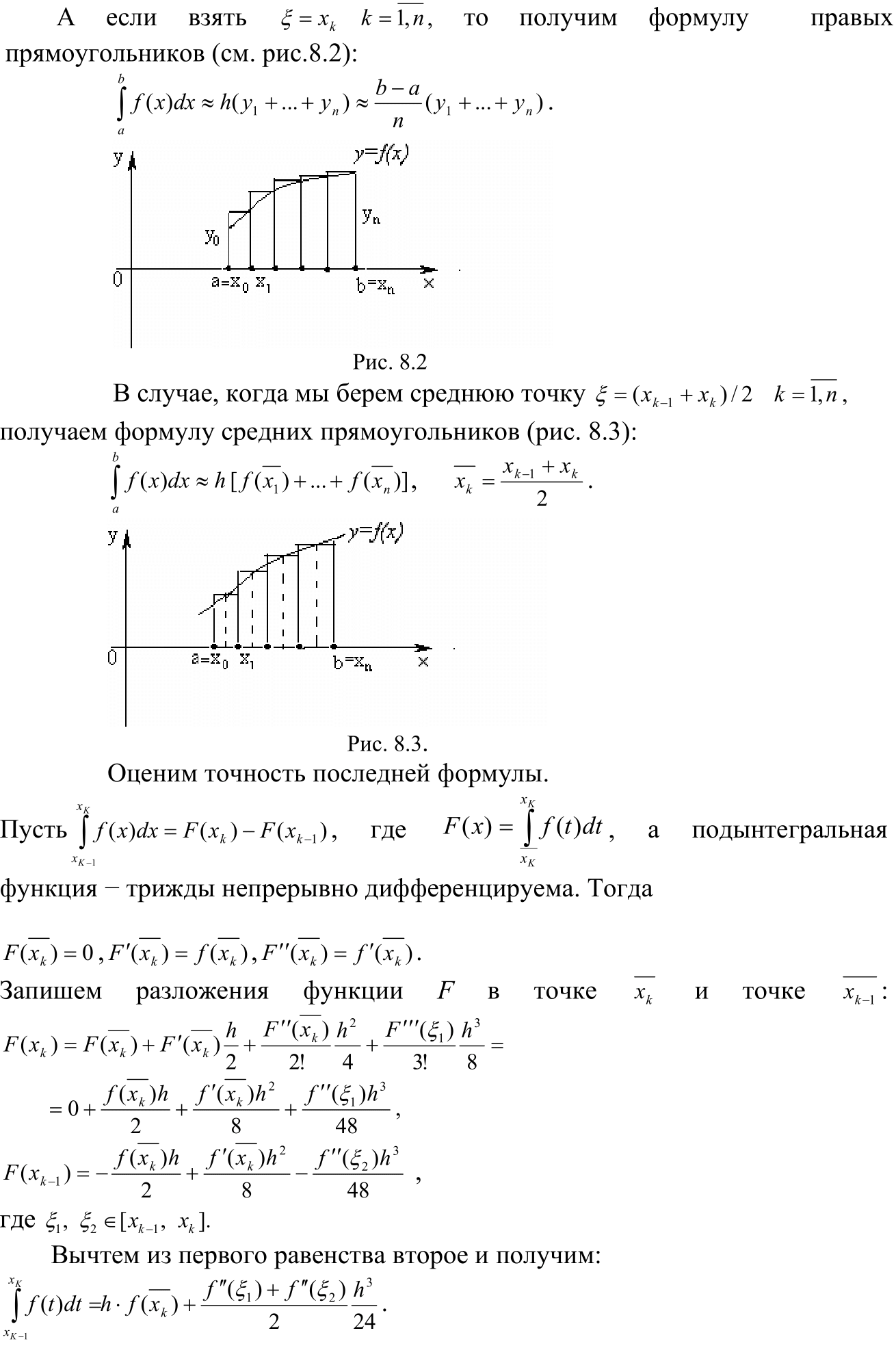
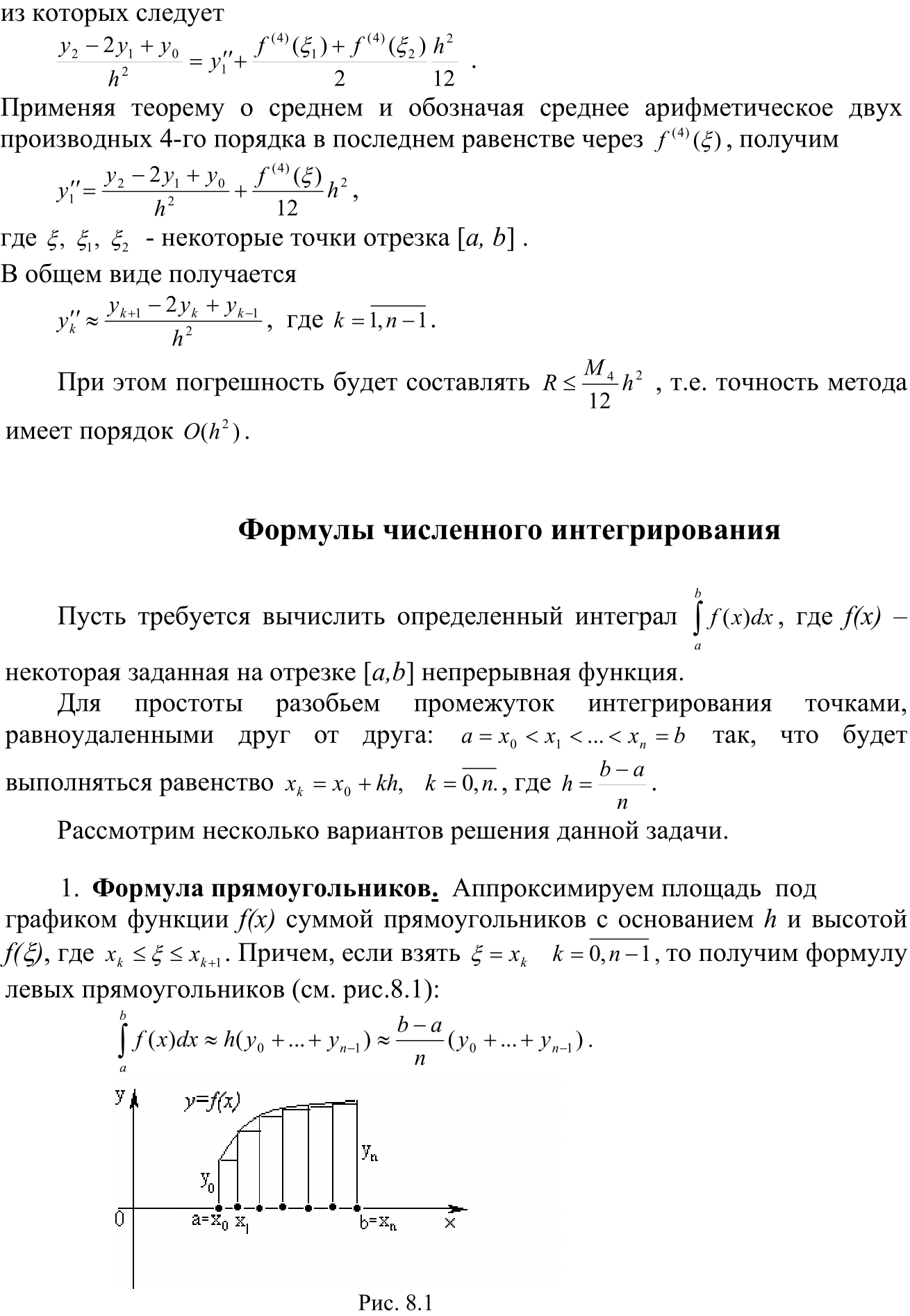
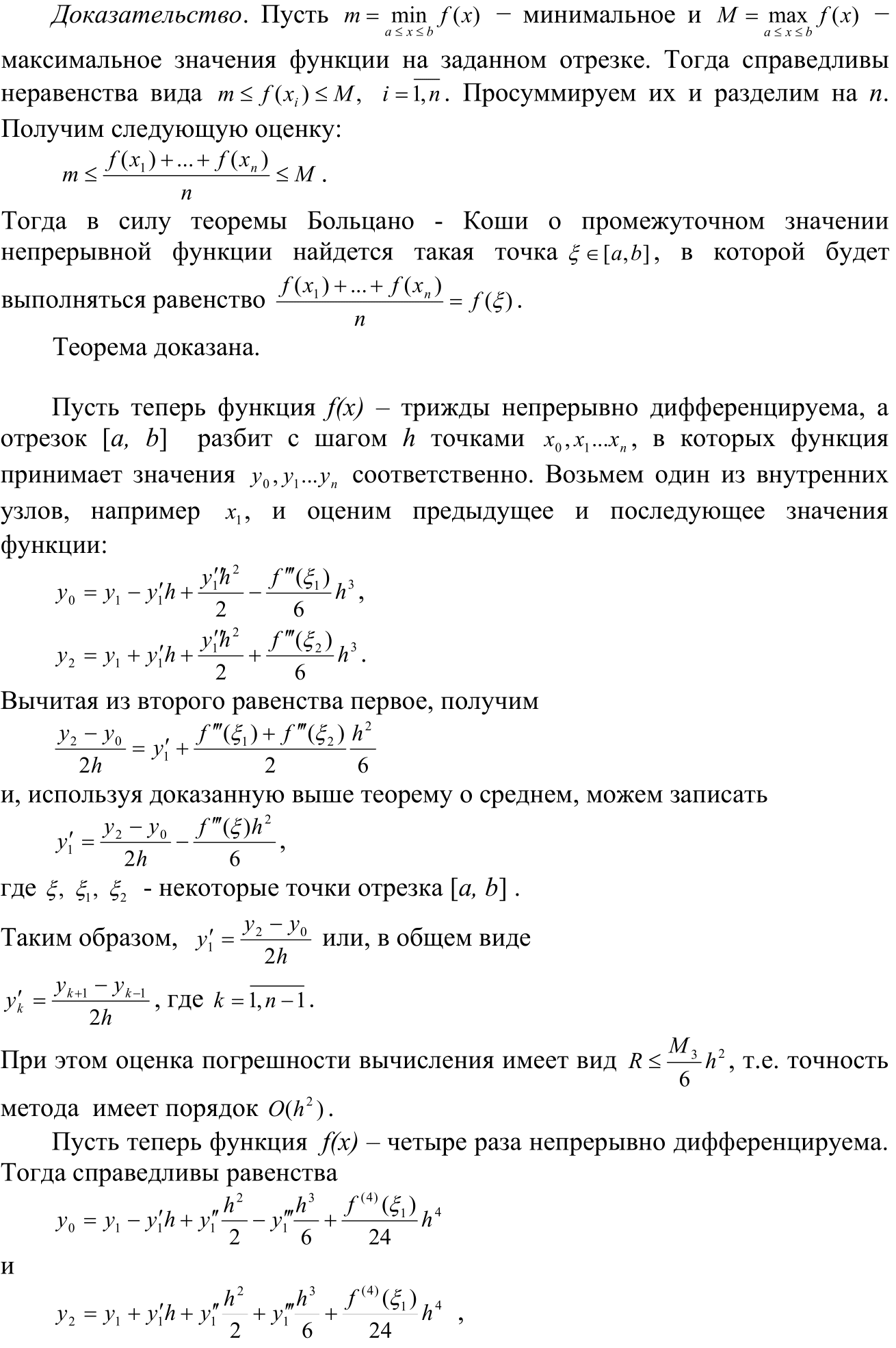
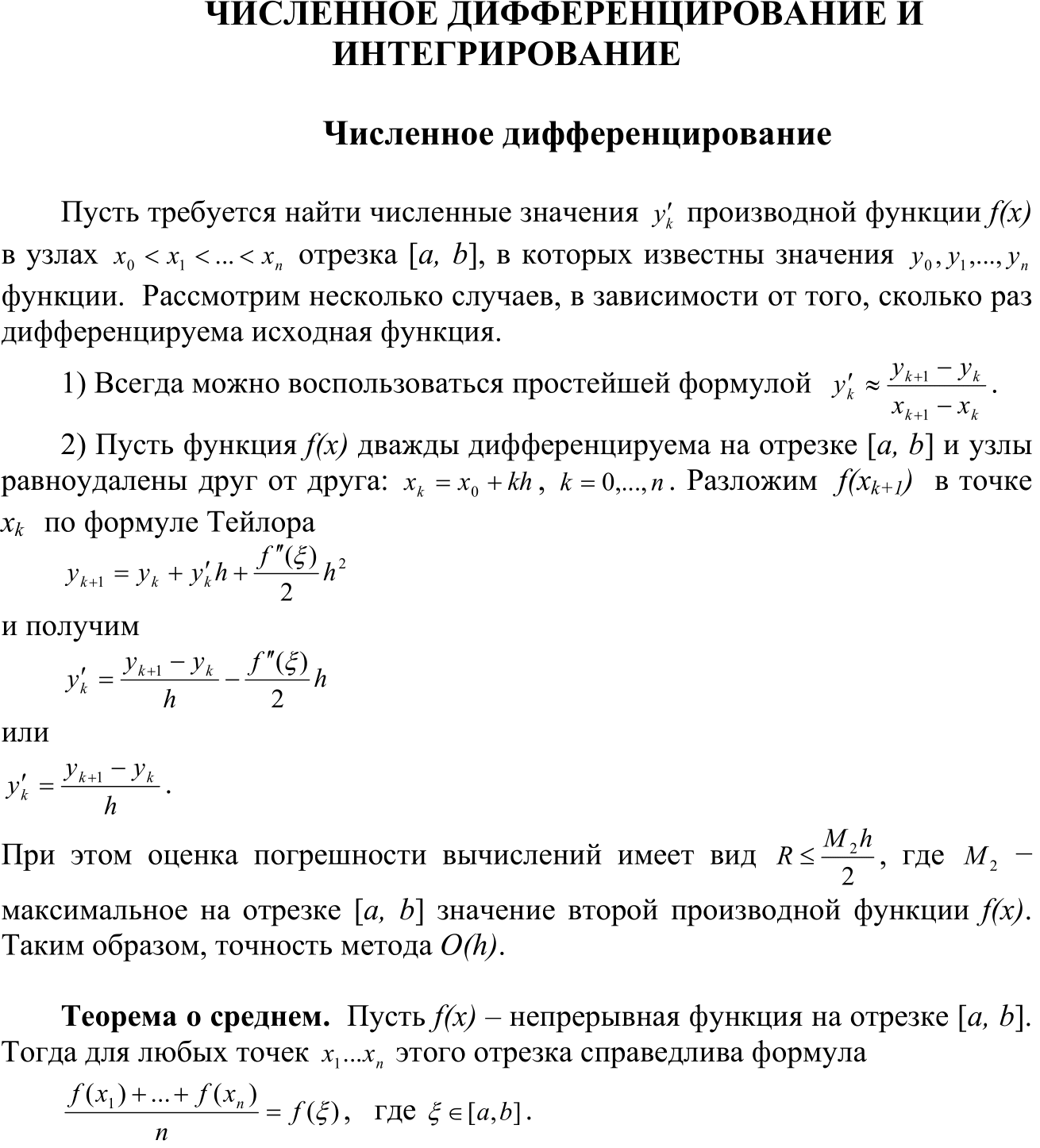
[**3. Программная реализация** 10](#_Toc118822960)

[**4. Выводы** 12](#_Toc118822961)

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**



# **Программная реализация**

Найти численное значение первой и второй производной в точке. Найти численное значение интеграла по формулам прямоугольников, трапеций, Симпсона

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Тестовый пример 1   |  |  |  | | --- | --- | --- | | *Функция* | *Точка* | *Интервал* | |  |  |  | | *Значение* | *Округл.* | *Приближ.* | |  | 0.333 | 0.334 | |  | -0.111 | -0.123 | | Метод оценки погрешностей | | | | *Вид* | *Округл.* |  | | *точный* | 4.0471896 | – | | *ср. прям.* | 4.0471896 | 0.0000001 | | *трапец.* | 4.0471895 | 0.0000001 | | *Симпсон* | 4.0471896 | 0.0000001 | | Тестовый пример 2   |  |  |  | | --- | --- | --- | | *Функция* | *Точка* | *Интервал* | |  |  |  | | *Значение* | *Округл.* | *Приближ.* | |  | 0.313 | 0.313 | |  | 2.500 | 2.510 | | Метод оценки погрешностей | | | | *Вид* | *Округл.* |  | | *точный* | 0.0000000 | – | | *ср. прям.* | 0.0000002 | 0.0000002 | | *трапец.* | 0.0000000 | 0.0000001 | | *Симпсон* | 0.0000000 | 0.0000001 | |
| Тестовый пример 3   |  |  |  | | --- | --- | --- | | *Функция* | *Точка* | *Интервал* | |  |  |  | | *Значение* | *Округл.* | *Приближ.* | |  | 2.301 | 2.301 | |  | 2.509 | 2.519 | | Метод оценки погрешностей | | | | *Вид* | *Округл.* |  | | *точный* | 11.5487394 | – | | *ср. прям.* | 11.5487392 | 0.0000002 | | *трапец.* | 11.5487395 | 0.0000002 | | *Симпсон* | 11.5487394 | 0.0000001 | |

ЗАДАНИЕ

Вариант 2

В каждом варианте найти численное значение первой и второй производной в точке, являющейся серединой заданного интервала, с точностью до 0,01. Вычислить с точностью 0,000001 интегралы по формулам прямоугольников, трапеций, Симпсона. Сравнить методы по точности.

Ответ:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Функция | | Интервал | | Производная в точке | |
|  | |  | |  | |
| Значение производных | | | | | |
| Первой | | | Второй | | |
| Округлённое | Приближённое | | Округлённое | | Приближённое |
| ≈ 0.500 | ≈ 0.500 | | ≈ – 0.500 | | ≈ – 0.500 |
| Значение интеграла | | | | | |
| Точное в точке x | | | Округлённое на интервале [0, 2] | | |
|  | | | ≈ 1.4095785 | | |
| Приближённое по методу остаточного отрезка через метод | | | | | |
| Средних | | Трапеций | | Симпсона | |
| ≈ 1.4095786 | | ≈ 1.4095784 | | ≈ 1.4095785 | |
| |Δ| < 0.0000002 | | |Δ| < 0.0000002 | | |Δ| < 0.0000001 | |
| N = 1078 | | N = 1599 | | N = 23 | |

**Код программы представлен в Приложении А**

# **Выводы**

Таким образом, в ходе выполнения лабораторной работы были изучены и сравнены по трудоёмкости и точности методы численного вычисления производных и методы численного интегрирования. Составлена компьютерная программа, на тестовых примерах проверена правильность её работы. Для функции заданного варианта найдено численное значение первой и второй производной в точке, вычислены с заданной точностью интегралы по формулам прямоугольников, трапеций, Симпсона. Самый точный результат для вычисления интеграла дал метод Симпсона.

**Приложение А**

print("Численное дифференцирование и интегрирование\n")

import numpy as np

import math

np.random.seed(12062003)

L, R, DerXdot = 0, 2, 1

def f(x):

return np.arctan(x)

def F(x):

return x \* np.arctan(x) - (1 / 2) \* np.log(x\*\*2 + 1)

def fd1(x):

return 1 / (x\*\*2 + 1)

def fd2(x):

return -(2 \* x) / (x\*\*2 + 1) \*\* 2

def fd3(x):

return (6 \* x\*\*2 - 2) / (x\*\*2 + 1) \*\* 3

def fd4(x):

return -(24 \* x \* (x\*\*2 - 1)) / (x\*\*2 + 1) \*\* 4

M2deLR, M4deLR = 0, 0

IntEps = 0.000001

IntFormatString = "{:.7f}"

DerEps = 0.01

DerFormatString = "{:.3f}"

def DerivativeFirst(f, x, d):

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

def DerivativeFirstViaEstimation(f, x):

M2 = abs(fd2(x))

df = 2 \* DerEps / M2

M3 = abs(fd3(x))

ds = (6 \* DerEps / M3) \*\* (1 / 2)

return DerivativeFirst(f, x, min(df, ds))

def DerivativeFirstViaTenInMinus5(f, x):

d = 10.0\*\*-5

return DerivativeFirst(f, x, d)

def DerivativeSecond(f, x, d):

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

def DerivativeSecondViaEstimation(f, x):

M4 = abs(fd4(x))

if M4 == 0:

d = DerEps

else:

d = (12 \* DerEps / M4) \*\* (1 / 2)

return DerivativeSecond(f, x, d)

def DerivativeSecondViaTenInMinus4(f, x):

d = 10.0\*\*-4

return DerivativeSecond(f, x, d)

print()

def delta1(derappr):

return np.ceil(abs(derappr - fd1(DerXdot)) \* (1 / (DerEps / 10))) \* (DerEps / 10)

print(f"Первая производная = {fd1(DerXdot):.3f}")

print(

f"С помощью приближений = {DerivativeFirstViaEstimation(f, DerXdot):.3f} | delta = {delta1(DerivativeFirstViaEstimation(f, DerXdot)):.3f}"

)

print(

f"Округленное = {DerivativeFirstViaTenInMinus5(f, DerXdot):.3f} | delta = {delta1(DerivativeFirstViaTenInMinus5(f, DerXdot)):.3f}"

)

print()

def delta2(derappr):

return np.ceil(abs(derappr - fd2(DerXdot)) \* (1 / (DerEps / 10))) \* (DerEps / 10)

print(f"Вторая производная = {fd2(DerXdot):.3f}")

print(

f"С помощью приближений = {DerivativeSecondViaEstimation(f, DerXdot):.3f} | delta = {delta2(DerivativeSecondViaEstimation(f, DerXdot)):.3f}"

)

print(

f"Округленное = {DerivativeSecondViaTenInMinus4(f, DerXdot):.3f} | delta = {delta2(DerivativeSecondViaTenInMinus4(f, DerXdot)):.3f}"

)

print()

def IntegralViaMiddleRectangles(f, L, R, N):

h = (R - L) / N

x = L + h / 2

s = 0.0

while x < R:

s += f(x) \* h

x += h

return s

def IntegralViaTrapezoids(f, L, R, N):

h = (R - L) / N

x = L + h / 2

s = 0.0

while x < R:

s += ((f(x - h / 2) + f(x + h / 2)) / 2) \* h

x += h

return s

def IntegralViaSimpson(f, L, R, N):

h = (R - L) / N

x = L + h / 2

s = 0.0

while x < R:

fa = f(x - h / 2)

fm = f(x)

fb = f(x + h / 2)

s += (fa + 4 \* fm + fb) \* h / 6

x += h

return s

def IntegralViaRandomSegments(method, f, L, R):

LeftCoeff, RightCoeff = 1 / 3, 1 / 2

h\_prev = R - L

ans\_prev = method(f, L, R, 1)

while True:

h\_new = h\_prev \* (LeftCoeff + (RightCoeff - LeftCoeff) \* np.random.rand())

N = math.floor((R - L) / h\_new)

M = L + h\_new \* N

ans\_new = method(f, L, M, N) + method(f, M, R, 1)

if abs(ans\_new - ans\_prev) < IntEps:

print("Случайные отрезки: N =", N)

return ans\_new

ans\_prev = ans\_new

h\_prev = h\_new

def IntegralViaMiddleRectanglesViaEstimation(f, L, R):

if M2deLR > 0.0:

M2 = M2deLR

h = (24 \* IntEps / (R - L) / M2) \*\* (1 / 2)

N = np.ceil((R - L) / h)

return IntegralViaMiddleRectangles(f, L, R, N)

else:

return IntegralViaRandomSegments(IntegralViaMiddleRectangles, f, L, R)

def IntegralViaTrapezoidsViaEstimation(f, L, R):

if M2deLR > 0.0:

M2 = M2deLR

h = (12 \* IntEps / (R - L) / M2) \*\* (1 / 2)

N = np.ceil((R - L) / h)

return IntegralViaTrapezoids(f, L, R, N)

else:

return IntegralViaRandomSegments(IntegralViaTrapezoids, f, L, R)

def IntegralViaSimpsonViaEstimation(f, L, R):

if M4deLR > 0.0:

M4 = M4deLR

h = (180 \* IntEps / (R - L) / M4) \*\* (1 / 4)

N = np.ceil((R - L) / h)

return IntegralViaSimpson(f, L, R, N)

else:

return IntegralViaRandomSegments(IntegralViaSimpson, f, L, R)

print()

intprecised = F(R) - F(L)

print(f"Интеграл = {intprecised:.7f}")

def delta(intappr):

return np.ceil(abs(intappr - intprecised) \* (1 / (IntEps / 10))) \* (IntEps / 10)

intappr = IntegralViaMiddleRectanglesViaEstimation(f, L, R)

print(f"Прямоугольники = {intappr:.7f} | delta = {delta(intappr):.7f}")

intappr = IntegralViaTrapezoidsViaEstimation(f, L, R)

print(f"Трапеции = {intappr:.7f} | delta = {delta(intappr):.7f}")

intappr = IntegralViaSimpsonViaEstimation(f, L, R)

print(f"Симпсон = {intappr:.7f} | delta = {delta(intappr):.7f}")

print()