**МИНОБРНАУКИ РОССИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМ. Р.Е. АЛЕКСЕЕВА»**

**(НГТУ)**

Кафедра «Вычислительные системы и технологии»

Дисциплина «Методы и средства обработки сигналов»

**Лабораторная работа №1**

Выполнил:

Студент группы 23–ВМз

Чернышев Антон Дмитриевич

Проверил:

доцент, кандидат технических наук

Авербух Михаил Леонидович

Нижний Новгород  
2025

**Задание**

Написать программу для расчета и визуализации функции, представляющей собой сумму трех синусов.

Имеем функцию y(x) = a1 \* sin(b1 \* x) + a2 \* sin(b2 \* x) + a3 \*sin(b3 \* x)

Пользователем задаются: a1, b1, a2, b2, a3, b3, x0 (начальное значение), xк (конечное значение), Δx (шаг). Расчет y(x) по заданным значениям a1, b1, a2, b2, a3, b3, x0 (начальное значение), xк (конечное значение), Δx (шаг). Отображение векторов x и y (в виде таблицы). Построение графика y(x) по

указанным векторам. GUI не обязателен (но приветствуется). Если студент решил не делать GUI, можно график просто сохранять в простом (!) графическом формате, таком как несжатый BMP (растровый формат, см. https://ru.wikipedia.org/wiki/BMP) или SVG (векторный формат, см. https://ru.wikipedia.org/wiki/SVG). Программу можно писать на любом языке программирования по выбору студента.

**Код программы**

#!/usr/bin/env python3

import numpy as np

import matplotlib.pyplot as plt

# Ввод параметров с проверками

def read\_params():

def ask(name: str) -> float:

while True:

try:

return float(input(f"{name}: "))

except ValueError:

print("Нужно число. Попробуйте ещё раз.")

a1 = ask("a1"); b1 = ask("b1")

a2 = ask("a2"); b2 = ask("b2")

a3 = ask("a3"); b3 = ask("b3")

x0 = ask("x0"); xk = ask("xk"); dx = ask("Δx")

if dx == 0:

raise ValueError("Δx не может быть 0.")

if (xk - x0) \* dx < 0:

raise ValueError("Знак Δx должен совпадать с направлением от x0 к xk.")

eps = abs(dx) \* 1e-9

x = np.arange(x0, xk + np.sign(dx) \* eps, dx, dtype=float)

return a1, b1, a2, b2, a3, b3, x

# Сама функция

def f(x: np.ndarray, a1, b1, a2, b2, a3, b3) -> np.ndarray:

return a1\*np.sin(b1\*x) + a2\*np.sin(b2\*x) + a3\*np.sin(b3\*x)

# Форматирование подписи вида: y(x) = a1·sin(b1·x) ± a2·sin(b2·x) ± a3·sin(b3·x)

def format\_function(a1, b1, a2, b2, a3, b3) -> str:

terms = []

for a, b in ((a1, b1), (a2, b2), (a3, b3)):

if a == 0:

continue

sign = "-" if a < 0 else "+"

term = f"{abs(a):g}·sin({b:g}·x)"

terms.append((sign, term))

if not terms:

return "y(x) = 0"

first\_sign, first\_term = terms[0]

expr = (("-" if first\_sign == "-" else "") + first\_term)

for sign, term in terms[1:]:

expr += f" {sign} {term}"

return "y(x) = " + expr

# Печать таблицы

def show\_table(x: np.ndarray, y: np.ndarray, limit: int = 2000):

print("Таблица значений y(x)")

for xi, yi in zip(x, y):

print(f"x: {xi:.3f}\ty: {yi:.3f}")

limit -= 1

if limit == 0:

print("... (обрезано)")

break

# График и сохранение

def plot\_xy(x: np.ndarray, y: np.ndarray, func\_str: str, out\_base: str = "plot"):

plt.figure(figsize=(12, 8), facecolor="white")

use\_markers = len(x) <= 500

plt.plot(

x, y,

marker='o' if use\_markers else None,

linestyle='dashed', linewidth=2, markersize=3

)

plt.title(f"График функции {func\_str}", fontsize=12, fontweight='bold')

plt.xlabel("X"); plt.ylabel("Y")

plt.grid(True, which='both', axis='both', linestyle='-', linewidth=0.75)

plt.axhline(0, color='black', linewidth=1)

plt.axvline(0, color='black', linewidth=1)

plt.tight\_layout()

plt.savefig(f"{out\_base}.svg")

plt.savefig(f"{out\_base}.png", dpi=200)

plt.show()

def main():

a1, b1, a2, b2, a3, b3, x = read\_params()

y = f(x, a1, b1, a2, b2, a3, b3)

func\_str = format\_function(a1, b1, a2, b2, a3, b3)

show\_table(x, y)

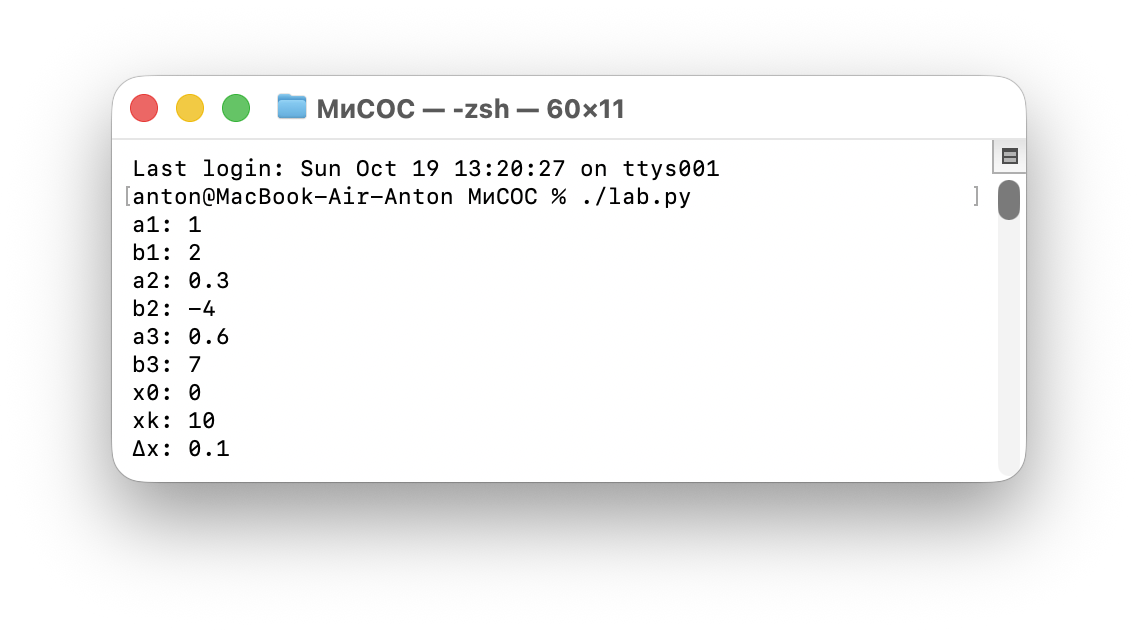
print("Закройте окно графика, чтобы завершить.")

plot\_xy(x, y, func\_str)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Демонстрация работы**

**Входные данные:**  
  
  
  
**Выходные данные:**  
  
Таблица значений y(x)

x: 0.000 y: 0.000

x: 0.100 y: 0.468

x: 0.200 y: 0.765

x: 0.300 y: 0.803

x: 0.400 y: 0.618

x: 0.500 y: 0.358

x: 0.600 y: 0.206

x: 0.700 y: 0.295

x: 0.800 y: 0.638

x: 0.900 y: 1.117

x: 1.000 y: 1.531

x: 1.100 y: 1.687

x: 1.200 y: 1.487

x: 1.300 y: 0.972

x: 1.400 y: 0.304

x: 1.500 y: -0.303

x: 1.600 y: -0.681

x: 1.700 y: -0.775

x: 1.800 y: -0.660

x: 1.900 y: -0.500

x: 2.000 y: -0.459

x: 2.100 y: -0.621

x: 2.200 y: -0.945

x: 2.300 y: -1.290

x: 2.400 y: -1.476

x: 2.500 y: -1.381

x: 2.600 y: -0.998

x: 2.700 y: -0.448

x: 2.800 y: 0.072

x: 2.900 y: 0.378

x: 3.000 y: 0.384

x: 3.100 y: 0.139

x: 3.200 y: -0.191

x: 3.300 y: -0.403

x: 3.400 y: -0.347

x: 3.500 y: 0.005

x: 3.600 y: 0.544

x: 3.700 y: 1.079

x: 3.800 y: 1.419

x: 3.900 y: 1.463

x: 4.000 y: 1.238

x: 4.100 y: 0.884

x: 4.200 y: 0.579

x: 4.300 y: 0.453

x: 4.400 y: 0.523

x: 4.500 y: 0.688

x: 4.600 y: 0.777

x: 4.700 y: 0.637

x: 4.800 y: 0.213

x: 4.900 y: -0.418

x: 5.000 y: -1.075

x: 5.100 y: -1.546

x: 5.200 y: -1.684

x: 5.300 y: -1.474

x: 5.400 y: -1.035

x: 5.500 y: -0.567

x: 5.600 y: -0.261

x: 5.700 y: -0.217

x: 5.800 y: -0.399

x: 5.900 y: -0.660

x: 6.000 y: -0.815

x: 6.100 y: -0.733

x: 6.200 y: -0.398

x: 6.300 y: 0.084

x: 6.400 y: 0.534

x: 6.500 y: 0.791

x: 6.600 y: 0.785

x: 6.700 y: 0.575

x: 6.800 y: 0.320

x: 6.900 y: 0.203

x: 7.000 y: 0.337

x: 7.100 y: 0.714

x: 7.200 y: 1.197

x: 7.300 y: 1.579

x: 7.400 y: 1.679

x: 7.500 y: 1.419

x: 7.600 y: 0.865

x: 7.700 y: 0.193

x: 7.800 y: -0.386

x: 7.900 y: -0.716

x: 8.000 y: -0.766

x: 8.100 y: -0.632

x: 8.200 y: -0.482

x: 8.300 y: -0.472

x: 8.400 y: -0.667

x: 8.500 y: -1.007

x: 8.600 y: -1.337

x: 8.700 y: -1.482

x: 8.800 y: -1.335

x: 8.900 y: -0.912

x: 9.000 y: -0.353

x: 9.100 y: 0.142

x: 9.200 y: 0.400

x: 9.300 y: 0.357

x: 9.400 y: 0.084

x: 9.500 y: -0.241

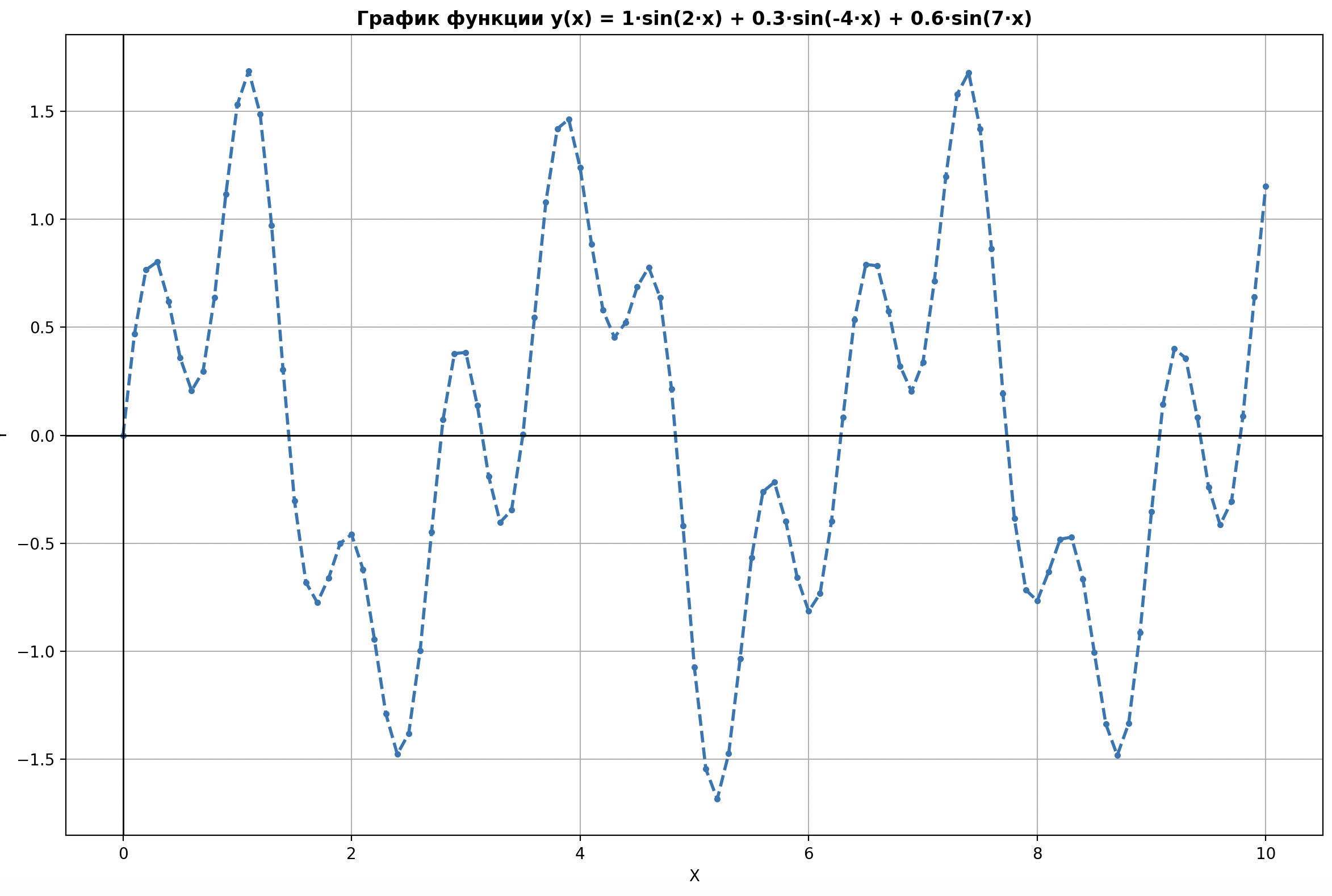
x: 9.600 y: -0.415

x: 9.700 y: -0.307

x: 9.800 y: 0.087

x: 9.900 y: 0.640

x: 10.000 y: 1.154



**Вывод**

В ходе выполнения лабораторной работы была успешно решена поставленная задача: разработана программа на языке Python для вычисления и визуализации функции, заданной суммой трёх синусоидальных составляющих. Программа корректно формирует векторы значений аргумента X и функции Y на заданном пользователем интервале с указанным шагом, выводит результаты в виде таблицы и строит график функции, сохраняя его в графический файл.