Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский  
политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчёт по практической работе №2  
по дисциплине «Информатика»

Выполнил: ПМ-23-2б Воронин М.А.

Проверил: ст. пр. каф. ВММБ Ильиных Г.В.

Пермь, 2023

# Задание 1

## Постановка задачи

1. Добавить возможность выбора количества точек на графике.

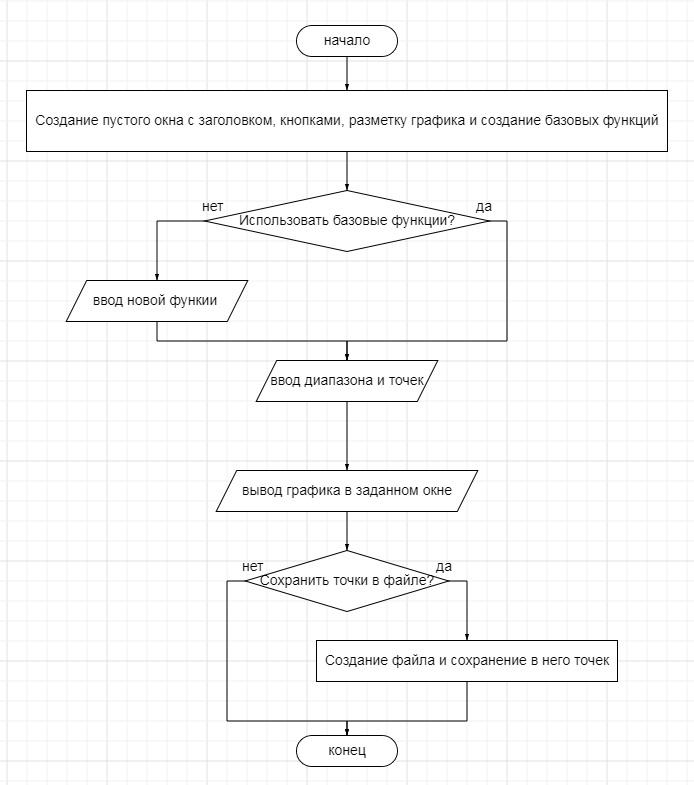
2. Добавить обработку ошибки ввода неверной функции.

3. Добавить кнопку очистки графика.

4. Добавить возможность сохранения точек графика в файл.

5. Добавить возможность добавления функций в виджет списка и выбора из него функций к отрисовке.

## 2 Алгоритм решения



## 3 Решение задачи, код программы

1. from PyQt5.QtWidgets import (QApplication, QLabel, QLineEdit, QMainWindow, QPushButton, QVBoxLayout, QWidget, QComboBox)  
   import numpy as np  
   import matplotlib.pyplot as plt  
   from matplotlib.backends.backend\_qt5agg import FigureCanvasQTAgg as FigureCanvas  
     
     
   class MainWindow(QMainWindow):  
    def \_\_init\_\_(self, parent=None):  
    super(MainWindow, self).\_\_init\_\_(parent)  
     
    self.setWindowTitle('График')  
    self.fig = plt.figure()  
    self.canvas = FigureCanvas(self.fig)  
     
    cental\_widget = QWidget()  
    layout = QVBoxLayout()  
    cental\_widget.setLayout(layout)  
     
    layout.addWidget(self.canvas)  
     
    self.setCentralWidget(cental\_widget)  
     
    self.plot\_button = QPushButton('Нарисовать график')  
    self.plot\_button.clicked.connect(self.plot\_data)  
     
    self.range\_label = QLabel('Диапазон:')  
    self.range\_start\_input = QLineEdit('0')  
    self.range\_end\_input = QLineEdit('1')  
     
    self.add\_function\_button = QPushButton('Добавить функцию в список')  
    self.function\_input = QLineEdit('\*Введите функцию для добавление в список\*')  
    self.function\_widget = QComboBox()  
    self.function\_widget.addItems(['x', '2\*x', 'x\*\*2', 'x\*\*3'])  
    self.add\_function\_button.clicked.connect(self.add\_function)  
     
    self.point\_amount = QLabel('Количество точек на графике:')  
    self.point\_input = QLineEdit('50')  
     
    self.clear\_button = QPushButton('Очистить график')  
    self.clear\_button.clicked.connect(self.clear\_plot)  
     
    self.file\_button = QPushButton('Сохранить точки в файл')  
    self.file\_button.clicked.connect(self.file\_save)  
     
    layout.addWidget(self.function\_widget)  
    layout.addWidget(self.range\_label)  
    layout.addWidget(self.range\_start\_input)  
    layout.addWidget(self.range\_end\_input)  
    layout.addWidget(self.point\_amount)  
    layout.addWidget(self.point\_input)  
    layout.addWidget(self.plot\_button)  
    layout.addWidget(self.clear\_button)  
    layout.addWidget(self.file\_button)  
    layout.addWidget(self.add\_function\_button)  
    layout.addWidget(self.function\_input)  
     
    def vectors(self):  
    try:  
    expression = self.function\_widget.currentText()  
    except NameError:  
    expression = 'x'  
     
    try:  
    range\_start = float(self.range\_start\_input.text())  
    range\_end = float(self.range\_end\_input.text())  
    points = int(self.point\_input.text())  
    except ValueError:  
    range\_start = 0  
    range\_end = 1  
    points = 50  
     
    functions = {}  
    exec(f'def f(x): return {expression}', functions)  
     
    x = np.linspace(range\_start, range\_end, points)  
    function = functions['f']  
    try:  
    y = [function(value) for value in x]  
    except NameError:  
    y = [value for value in x]  
     
    return x, y  
     
    def plot\_data(self):  
     
    x, y = self.vectors()  
    axes = plt.subplot()  
    axes.plot(x, y)  
    plt.grid(True)  
    plt.xlabel('x')  
    plt.ylabel('y')  
     
    self.centralWidget().layout().itemAt(0).widget().draw()  
     
    def clear\_plot(self):  
    for ax in self.fig.axes:  
    ax.clear()  
    plt.grid(True)  
    self.canvas.draw()  
     
    def file\_save(self):  
    x, y = self.vectors()  
    file = open('qwerty.txt', 'w')  
    file.write('x ' + ' ' + 'y\n')  
    for i in range(len(x)):  
    a, b = map(str, (x[i], y[i]))  
    a, b = a[0:6], b[0:6]  
    if len(a) < 6:  
    a += '0' \* (6 - len(a))  
    if len(b) < 6:  
    b += '0' \* (6 - len(b))  
    file.write(a + ' ' + b + '\n')  
     
    def add\_function(self):  
    text\_x = self.function\_input.text()  
    self.function\_widget.addItems([text\_x])  
     
     
   app = QApplication([])  
   main\_window = MainWindow()  
   main\_window.show()  
   app.exec()
2. ***Тестирование работы программы с проверкой***

|  |  |  |
| --- | --- | --- |
|  | Python | Wolfram alpha |
| 1  Функция 2x  Диапазон от -10 до 10 |  |  |
| 2  Функция x  Диапазон от -10 до 10 |  |  |
| 3  Функция x2  Диапазон от -10 до 10 |  |  |
| 4  Функция x3  Диапазон от -10 до 10 |  |  |