**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П.О.СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Дисциплина «высокоуровневые языки программирования»

ОТЧЕТ

Лабораторная работа № 3

Основные компоненты. Размещение компонентов в окнах

Выполнил студент

Группы ИТД-21

Чайдаков И.М.

Принял преподаватель

Ястребов А. А.

Гомель 2024

Основные компоненты. Размещение компонентов в окнах

**Цель работы:** изучить основные компоненты, определенные в модуле *QtWidgets*; изучить способы размещения компонентов в окнах.

**Задание:** Создать, которое при запуске выводит окно с надписью. Поменять значок в заголовке на свой. В заголовке окна указать свои имя, фамилию и группу.

Под надписью должно находиться 3 кнопки. При наведении мыши на кнопку должна выводиться подсказка.

Кнопки

1 — Открыть окно с анкетой. Использовать типы размещения и типы компонентов. После нажатия на кнопку отправить все ответы пользователя должны выводиться в сеть.

2 — Открыть окно, в котором по введенному значению аргумента, вычисляется значение функции, заданной в виде графика. В окне должен быть организован ввод пользователем аргумента *X*. И вывод значения аргумента *Y*. На форме вывести график в виде рисунка. Необходимо добавить кнопки: «решить», «очистить», «выход».

3 — Открыть окно, в котором по введенному значению аргумента, определяется попадает ли точка с заданными координатами в область, закрашенную серым цветом. В окне должен быть организован ввод пользователем аргументов *X*, *Y* и радиуса *R*. И вывод — попадает или не попадает точка в область. На форме исходные данные вывести в виде рисунка. Необходимо добавить кнопки: «определить», «очистить», «выход».

**Ход выполнения задания:**

**1.** Описываем класс для создания главного окна (рисунок 1).

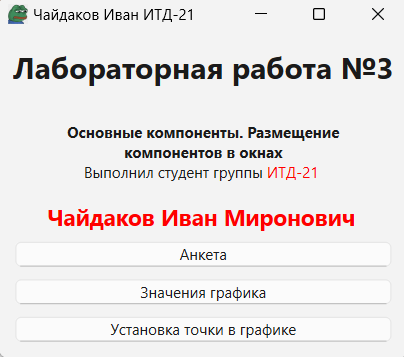


Рисунок 1 — Основное окно для вызова остальных окон.

**2.** Описываем класс для вывода анкеты и отображаем ее на экране нажатием первой кнопки (рисунок 2).

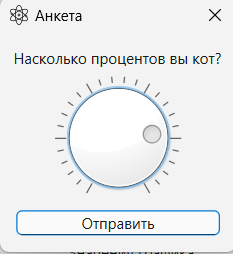


Рисунок 2 — Анкета с опросом пользователя.

**3.** Описываем класс определения значения аргумента на плоском графике и открываем его нажатием второй кнопки (рисунок 3).

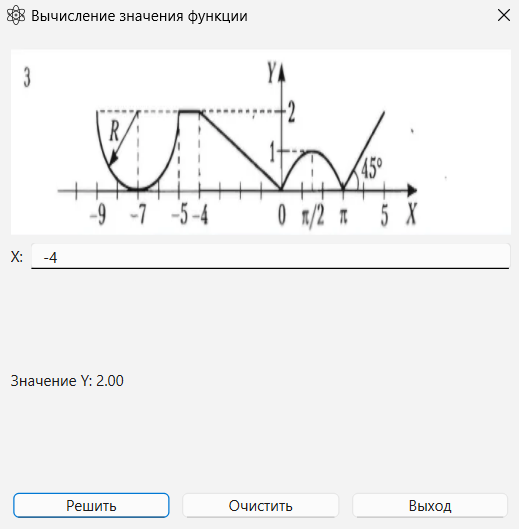


Рисунок 3 — Проверка значений функции.

На графике указано пять разных функций идущих в ряд и формула вычисления аргумента Y будет зависеть от указанного пользователем аргумента X.

**4.** Описываем класс для проверки попадания произвольной точки в область указанную на графике и открываем его нажатием третей кнопки (рисунок 4).

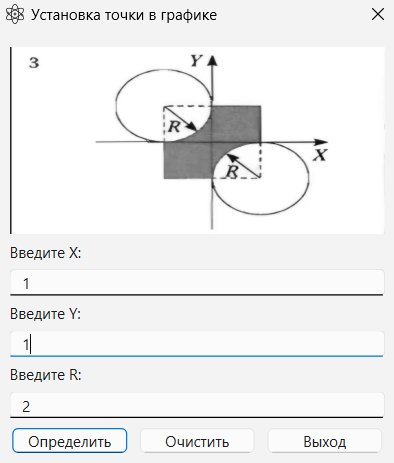


Рисунок 4 — Ввод полей аргументов для работы функции.

Далее при нажатии кнопки «Определить», мы можем наблюдать на всплывающем окне попала наша точка в серую область или нет (рисунок 5).

**Вывод:** Использование компонентов в окнах открывает доступ к более сложным структурам окон и возможности работы с большим количеством данных пользователя. Компоненты являются очень полезным решением для создания программ для математических расчетов на языке *Python*.

**ПРИЛОЖЕНИЕ А**

**Главное окно**

import sys

from PyQt6.QtWidgets import QApplication, QWidget, QLabel, QPushButton, QVBoxLayout

from PyQt6.QtGui import QIcon, QGuiApplication

from PyQt6.QtCore import Qt

from Question import SurveyWindow # type: ignore

from Graph import GraphWindow # type: ignore

from DotGraph import InputWindow # type: ignore

class MainWindow(QWidget):

    def \_\_init\_\_(self, parent=None):

        QWidget.\_\_init\_\_(self, parent)

        self.setWindowTitle("Чайдаков Иван ИТД-21")

        self.setWindowIcon(QIcon("Images/Pepe.jpg"))

        screen = QGuiApplication.primaryScreen()

        siz = screen.size()

        window\_width, window\_height = 300, 200

        self.setGeometry(

            (siz.width() - window\_width) // 2,

            (siz.height() - window\_height) // 2,

            window\_width,

            window\_height

        )

        label = QLabel('''

            <center>

                <h1>Лабораторная работа №3</h1><br>

                <b>Основные компоненты. Размещение</b><br>

                <b>компонентов в окнах</b><br>

                Выполнил студент группы <span style="color: red;">ИТД-21

                <h2>Чайдаков Иван Миронович</h2></span>

            </center>

        ''')

        label.setAlignment(Qt.AlignmentFlag.AlignCenter)

        button1 = QPushButton("Анкета", self)

        button1.setToolTip("Анкета с опросом насколько вы кот")

        button1.clicked.connect(self.openSurvey)

        button2 = QPushButton("Значения графика", self)

        button2.setToolTip("Проверка значения Y на плоском графике")

        button2.clicked.connect(self.openGraph)

        button3 = QPushButton("Установка точки в графике", self)

        button3.setToolTip("Проверка попадания точки в закрашенную область на графике")

        button3.clicked.connect(self.openDotGraph)

        layout = QVBoxLayout()

        layout.addWidget(label)

        layout.addWidget(button1)

        layout.addWidget(button2)

        layout.addWidget(button3)

        self.setLayout(layout)

    def openSurvey(self):

        self.surveyWindow = SurveyWindow(self)

        self.surveyWindow.show()

    def openGraph(self):

        self.graphWindow = GraphWindow(self)

        self.graphWindow.show()

    def openDotGraph(self):

        self.inputWindow = InputWindow(self)

        self.inputWindow.show()

if \_\_name\_\_ == '\_\_main\_\_':

    app = QApplication(sys.argv)

    window = MainWindow()

    window.show()

    sys.exit(app.exec())

**ПРИЛОЖЕНИЕ B**

**Классы побочных окон**

**Анкета:**

import sys

from PyQt6.QtWidgets import QApplication, QLabel, QPushButton, QGridLayout, QDialog, QDial

from PyQt6.QtGui import QIcon

class SurveyWindow(QDialog):

    def \_\_init\_\_(self, parent=None):

        QDialog.\_\_init\_\_(self, parent)

        self.setWindowTitle("РђРЅРєРµС‚Р°")

        layout = QGridLayout()

        self.setWindowIcon(QIcon("Images/logo.png"))

        question\_label = QLabel("РќР°СЃРєРѕР»СЊРєРѕ РїСЂРѕС†РµРЅС‚РѕРІ РІС‹ РєРѕС‚?")

        self.dial = QDial()

        self.dial.setMinimum(0)

        self.dial.setMaximum(100)

        self.dial.setValue(50)

        self.dial.setNotchesVisible(True)

        submitButton = QPushButton("РћС‚РїСЂР°РІРёС‚СЊ")

        submitButton.clicked.connect(self.submitResponse)

        layout.addWidget(question\_label, 0, 0, 1, 2)

        layout.addWidget(self.dial, 1, 0, 1, 2)

        layout.addWidget(submitButton, 2, 0, 1, 2)

        self.setLayout(layout)

    def submitResponse(self):

        response\_value = self.dial.value()

        print("РћС‚РІРµС‚ РїРѕР»СЊР·РѕРІР°С‚РµР»СЏ:", response\_value)

        self.accept()

if \_\_name\_\_ == '\_\_main\_\_':

    app = QApplication(sys.argv)

    window = SurveyWindow()

    window.show()

    sys.exit(app.exec())

**Расчет аргумента *Y* на графике:**

import sys

import numpy as np

from PyQt6.QtWidgets import QApplication, QLabel, QPushButton, QVBoxLayout, QHBoxLayout, QLineEdit, QDialog

from PyQt6.QtGui import QPixmap, QIcon

class GraphWindow(QDialog):

    def \_\_init\_\_(self, parent=None):

        QDialog.\_\_init\_\_(self, parent)

        self.setWindowTitle("Вычисление значения функции")

        self.setGeometry(100, 100, 400, 400)

        self.setWindowIcon(QIcon("Images/logo.png"))

        self.inputX = QLineEdit(self)

        self.inputX.setPlaceholderText("Введите значение X")

        self.outputY = QLabel("Значение Y:", self)

        solveButton = QPushButton("Решить", self)

        solveButton.clicked.connect(self.solveFunction)

        clearButton = QPushButton("Очистить", self)

        clearButton.clicked.connect(self.clearFields)

        exitButton = QPushButton("Выход", self)

        exitButton.clicked.connect(self.close)

        self.graph = QLabel(self)

        pixmap = QPixmap("Images/graph.webp")

        self.graph.setPixmap(pixmap)

        self.graph.setScaledContents(True)

        self.graph.setFixedSize(400, 150)

        layout = QVBoxLayout()

        layout.addWidget(self.graph)

        inputLayout = QHBoxLayout()

        inputLayout.addWidget(QLabel("X:"))

        inputLayout.addWidget(self.inputX)

        layout.addLayout(inputLayout)

        layout.addWidget(self.outputY)

        button\_layout = QHBoxLayout()

        button\_layout.addWidget(solveButton)

        button\_layout.addWidget(clearButton)

        button\_layout.addWidget(exitButton)

        layout.addLayout(button\_layout)

        self.setLayout(layout)

    def solveFunction(self):

        try:

            x = float(self.inputX.text())

            y = self.calculate\_y(x)

            self.outputY.setText(f"Значение Y: {y:.2f}")

        except ValueError:

            self.outputY.setText("Ошибка: введите число для X")

    def clearFields(self):

        self.inputX.clear()

        self.outputY.setText("Значение Y:")

    def calculate\_y(self, x):

        if -9 <= x <= -5:

            y = np.cos(45 \* (x+9))\*2

        elif -5 < x <= -4:

            y = 2

        elif -4 < x <= 0:

            y = 0 + x \* -0.5

        elif 0 < x <= np.pi:

            y = np.cos(x) + 1

        elif np.pi < x <= 5:

            y = np.tan(np.radians(45)) \* (x - np.pi)

        else:

            y = 0

        return y

if \_\_name\_\_ == '\_\_main\_\_':

    app = QApplication(sys.argv)

    window = GraphWindow()

    window.show()

    sys.exit(app.exec())

**Проверка попадания точки на график:**

import sys

from PyQt6.QtGui import QPixmap, QIcon

from PyQt6.QtWidgets import (

    QApplication, QPushButton, QLabel, QLineEdit,

    QVBoxLayout, QDialog, QMessageBox, QHBoxLayout,

)

class InputWindow(QDialog):

    def \_\_init\_\_(self, parent=None):

        QDialog.\_\_init\_\_(self, parent)

        self.setWindowTitle("Установка точки в графике")

        self.setGeometry(200, 200, 300, 200)

        self.setWindowIcon(QIcon("Images/logo.png"))

        self.graph = QLabel(self)

        pixmap = QPixmap("Images/graph2.webp")

        self.graph.setPixmap(pixmap)

        self.graph.setScaledContents(True)

        self.graph.setFixedSize(300, 150)

        self.labelX = QLabel("Введите X:")

        self.inputX = QLineEdit()

        self.labelY = QLabel("Введите Y:")

        self.inputY = QLineEdit()

        self.labelR = QLabel("Введите R:")

        self.inputR = QLineEdit()

        self.detButton = QPushButton("Определить")

        self.detButton.clicked.connect(self.detPoint)

        self.clearButton = QPushButton("Очистить")

        self.clearButton.clicked.connect(self.clearFields)

        self.exitButton = QPushButton("Выход")

        self.exitButton.clicked.connect(self.close)

        layout = QVBoxLayout()

        layout.addWidget(self.graph)

        layout.addWidget(self.labelX)

        layout.addWidget(self.inputX)

        layout.addWidget(self.labelY)

        layout.addWidget(self.inputY)

        layout.addWidget(self.labelR)

        layout.addWidget(self.inputR)

        buttonLayout = QHBoxLayout()

        buttonLayout.addWidget(self.detButton)

        buttonLayout.addWidget(self.clearButton)

        buttonLayout.addWidget(self.exitButton)

        layout.addLayout(buttonLayout)

        self.setLayout(layout)

    def detPoint(self):

        try:

            x = float(self.inputX.text())

            y = float(self.inputY.text())

            r = float(self.inputR.text())

            if (x >= 0 and y >= 0 and x\*\*2 + y\*\*2 <= r\*\*2) or \

                (x <= 0 and y <= 0 and x\*\*2 + y\*\*2 <= r\*\*2):

                QMessageBox.information(self, "Результат", "Точка находится в серой области.")

            else:

                QMessageBox.information(self, "Результат", "Точка не находится в серой области.")

        except ValueError:

            QMessageBox.warning(self, "Ошибка", "Пожалуйста, введите корректные значения для X, Y и R.")

    def clearFields(self):

        self.inputX.clear()

        self.inputY.clear()

        self.inputR.clear()

if \_\_name\_\_ == '\_\_main\_\_':

    app = QApplication(sys.argv)

    mainWindow = InputWindow()

    mainWindow.show()

    sys.exit(app.exec())