федеральное государственное бюджетное образовательное
учреждение высшего образования
«Национальный исследовательский университет «МЭИ»

Институт информационных и вычислительных технологий
Кафедра управления и интеллектуальных технологий

Отчёт по лабораторной работе №2
По дисциплине «Управление в больших системах»
«Структурный анализ больших систем управления»

Выполнил студент: Михайловский М. Ю.
Группа: А-03-21
Вариант: 5
Проверили: Новиков В. Н, Обычайко Д. С.

Москва 2024

Содержание

1 Постановка задачи **3**

2 Подход к решению задачи **4**

2.1 Упорядочивание вершин 4

2.2 Нахождение наименьшего пути 5

3 Реализованная программа **5**

3.1 Техническое описание 5

3.2 Пример работы программы 6

4 Выводы **7**

А Блок-схемы алгоритмов **8**

Б Листинг программы **10**

1 Постановка задачи

Дана матрица смежности $A \in \mathbb{R}^{n \times n}$, определяющая ориентированный граф системы с заданными в ней весами дуг: $G = (V, E)$. Необходимо реализовать процедуру упорядочения вершин по уровням и нахождения минимальных путей в графе.

$$A = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 6 & 4 & 0 & 5 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
3 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix} \tag{1}$$

Граф G, который задан такой матрицей смежности A показан на рис. 1.1. Как видно, граф является довольно сложным, и визуальный анализ системы при работе с таким графом будет затруднён, поскольку его структура весьма запутанная.

Для более простого анализа такой системы далее в работе будет проведено упорядочивание вершин.

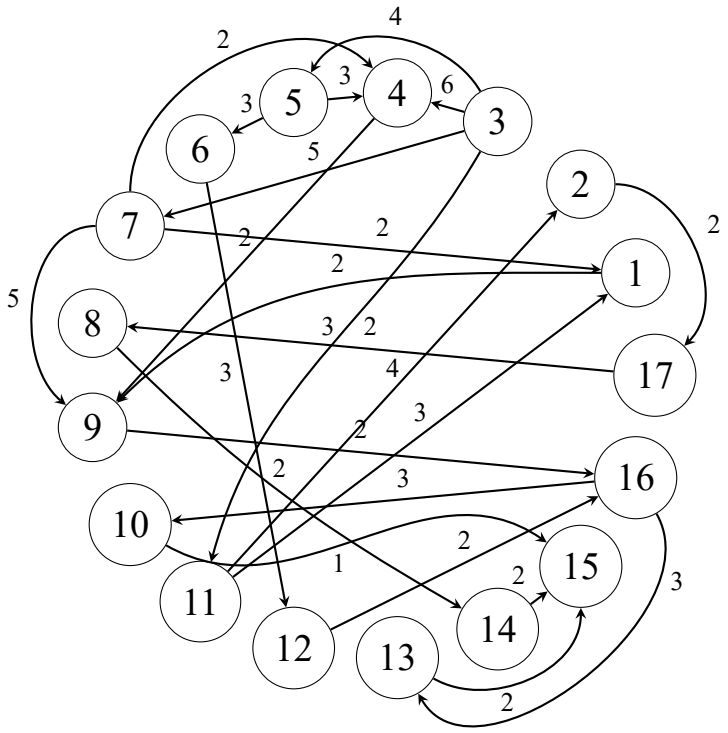

Рис. 1.1. Граф данной системы

2 Подход к решению задачи

2.1 Упорядочивание вершин

Упорядочивание на $L+1$ уровней $N = \{N_0,\ N_1,\ \dots\ N_L\}$ проводится, исходя из множеств левых инциденций i-ых узлов: $G^{-1}(i)$ $(i = \overline{1,\ n})$, по принципу (2):

$$\begin{aligned} N_0 &= \left\{i \mid G^{-1}(i)\varnothing\right\} \\ N_1 &= \left\{i \mid G^{-1}(i) \subset N_0\right\} \\ N_2 &= \left\{i \mid G^{-1}(i) \subset (N_0 \cup N_1)\right\} \\ &\vdots \\ N_L &= \left\{i \mid G^{-1}(i) \subset \bigcup_{k=0}^{L-1} N_k\right\} \end{aligned} \tag{2}$$

В результате упорядочивания легко построить граф этой же системы в виде, более наглядно представляющем структуру системы.

2.2 Нахождение наименьшего пути

В данной работе реализован алгоритм, похожий на алгоритм Дейкстры, но учитывающий упорядоченную структуру графа. Всем вершинам, кроме начальной задаётся метка равная ∞. Начальной вершине задаётся метка, равная 0. Алгоритм начинает работу с 0-ого уровня.

1. Рассматриваются все дуги инцидентные текущему уровню;
2. Если для пары (u, v) вершин соответствующих такой дуге $c(u) + c((u, v)) < c(v)$ ($c(u)$ - метка вершины u, $c((u, v))$ - цена дуги), то на вершину v ставится метка равная этой сумме;
3. Если текущий уровень $L - 1$, то завершить алгоритм. Иначе перейти на следующий уровень.

3 Реализованная программа

3.1 Техническое описание

Программа реализована на *Python 3.11.5* с модулями *PyQt6, numpy, networkx, matplotlib*. Программа состоит из следующих основных модулей (листинг которых приведён в приложении):

- **main.py** – Основной модуль, использующийся для запуска программы;
- **userInfo.py** – модуль для получения и проверки введённых пользователям данных;
- **tableHandlers.py** – модуль для работы с таблицей, в которой задаётся матрица смежности A;
- **GraphOptimizer.py** – модуль для упорядочивания вершин графа и поиска минимальных путей;
- **GraphDrawer.py** – модуль для построения визуализации упорядоченного графа;
- **my_networkx.py** – вспомогательный к **GraphDrawer.py** модуль, используемый для проставления весов на дугах, которые построены с кривизной;
- **main_window.ui** – разметка окна программы.

3.2 Пример работы программы

При открытии программы появляется окно, представленное на рис. 3.1. Здесь можно ввести исходную матрицу A.

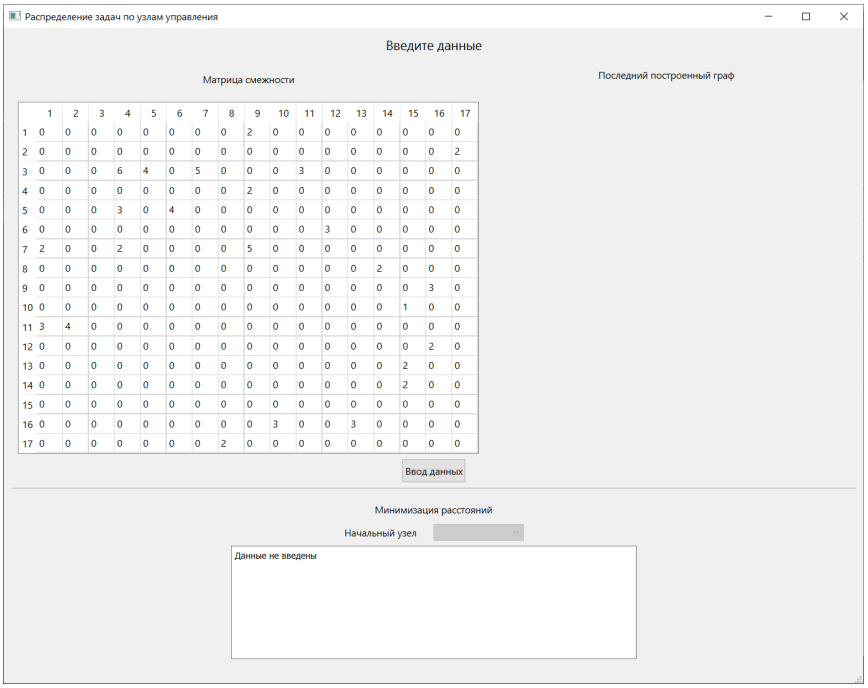

Рис. 3.1. Основное окно программы

После нажатия кнопки «Ввод данных» заполненная матрица A считывается и строится упорядоченный граф, как на рис. 3.2.

После этого в блоке снизу можно выбрать начальный узел для расчёта минимальных путей. Будут выведены списком только достижимые из выбранного начального узла вершины: рис. 3.3-3.4.

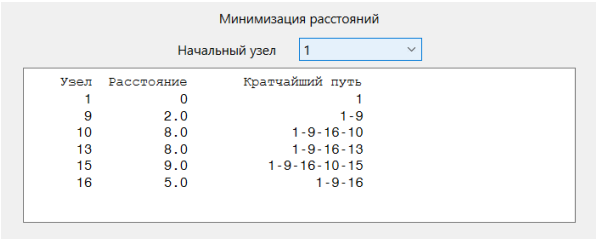

Рис. 3.3. Пример минимальных путей из узла 1

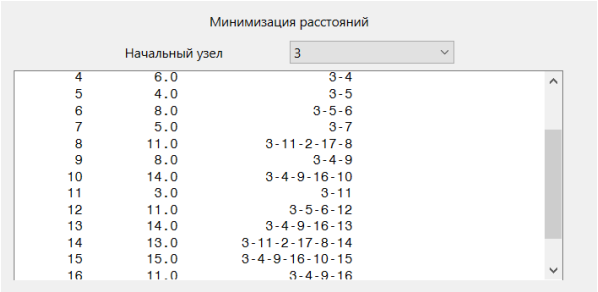

Рис. 3.4. Пример минимальных путей из узла 3

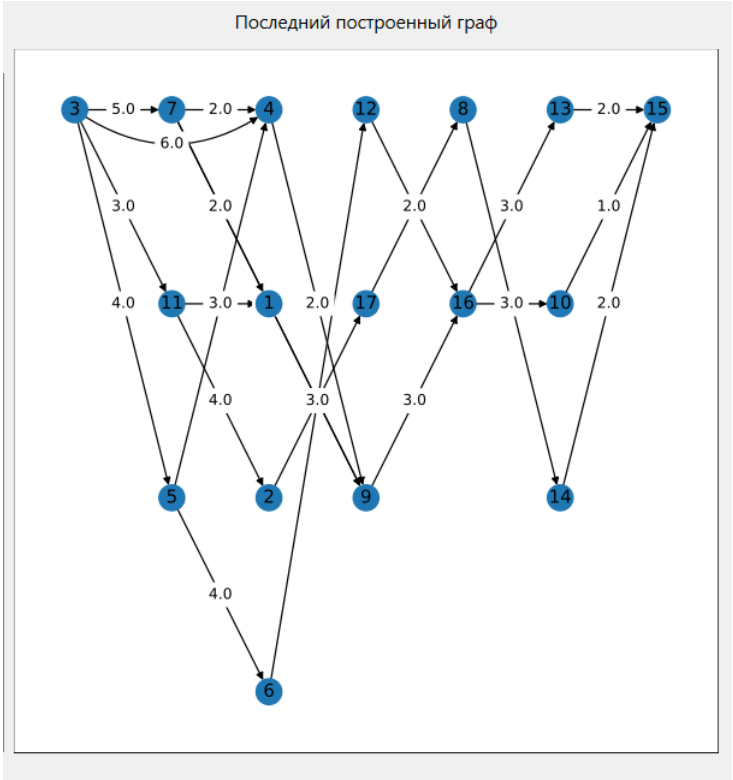

Рис. 3.2. Построенный упорядоченный граф

4 Выводы

В данной работе была реализована программа для работы с ориентированными графами с заданными ценами дуг. В ней было реализовано две основные процедуры:

- Упорядочивание вершин введённого графа по уровням – это позволяет представить граф в удобном для анализа структуры системы виде;
- Нахождение минимального пути между заданными вершинами.

Приложение А. Блок-схемы алгоритмов

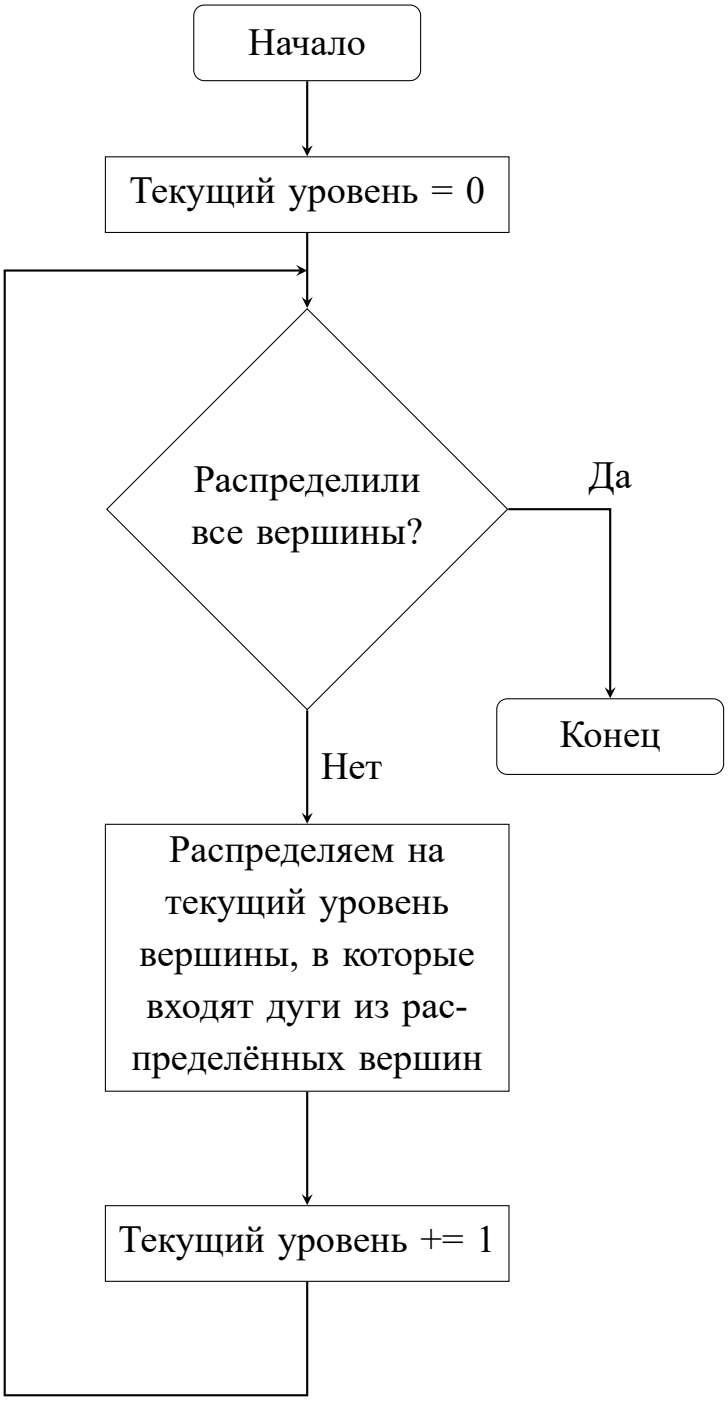

Рис. А.1. Распределение по уровням

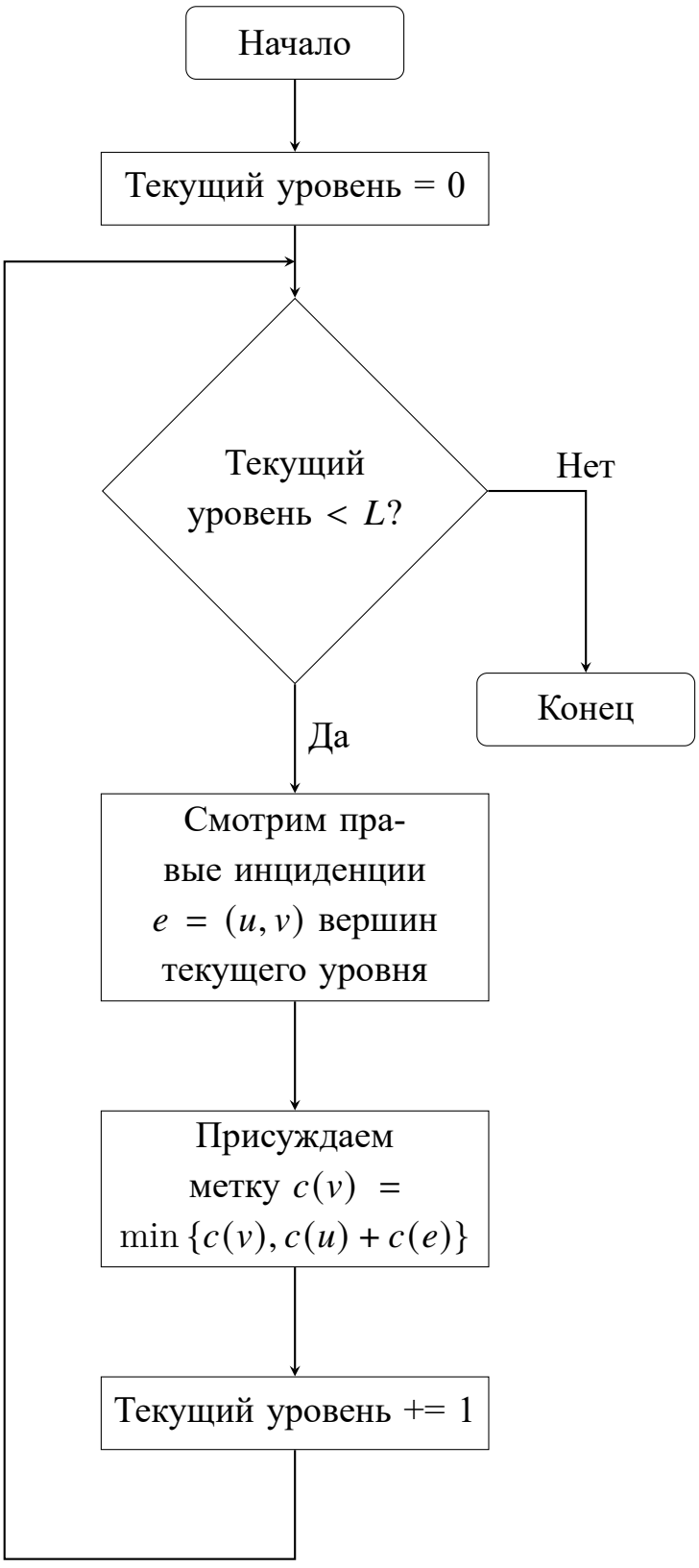

Рис. А.2. Нахождение минимальных путей

Приложение Б. Листинг программы

Листинг 1. main.py

```
from PyQt6 import uic
from PyQt6.QtWidgets import QApplication, QComboBox, QTextBrowser
from PyQt6.QtSvg import QSvgRenderer
from PyQt6.QtGui import QPainter
import numpy as np
from GraphDrawer import GraphDrawer
from GraphOptimizer import GraphOptimizer
from pathlib import Path

drawingAllowed = False

A_default = np.array([
    #  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17
    [  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2],
    [  0,  0,  0,  6,  4,  0,  5,  0,  0,  0,  3,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  3,  0,  4,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  3,  0,  0,  0,  0,  0],
    [  2,  0,  0,  2,  0,  0,  0,  0,  5,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  3,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  0,  0],
    [  3,  4,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  2,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  0,  0,  3,  0,  0,  3,  0,  0,  0,  0],
    [  0,  0,  0,  0,  0,  0,  0,  2,  0,  0,  0,  0,  0,  0,  0,  0,  0],
], dtype=np.int16)

def input_data(dataGetter, graphOptimizer, A_table):
    dataGetter.catch_input_errors()
    graphOptimizer.init_from_matrix(A_table.matrix)

def add_nodes_to_combo(node_names: list, combo: QComboBox):
    combo.clear()
    combo.addItems(sorted(node_names, key=float))
    combo.setEnabled(True)

def draw_graph(graphOptimizer, A_table):
    graphDrawer = GraphDrawer(graphOptimizer.levels, A_table.matrix)
    fig, ax = graphDrawer.draw()
    Path('./tmp').mkdir(exist_ok=True)
    fig.savefig('./tmp/graph.svg', bbox_inches='tight', pad_inches=0)
    global drawingAllowed
    drawingAllowed = True
```

```
49
50 def calc_min_tracks(start_node_name: str, graphOptimizer: GraphOptimizer,
   ↪ tracksInfo: QTextBrowser):
51     graphOptimizer.min_tracks_from(start_node_name)
52     tracksInfo.setText(graphOptimizer.min_tracks_info_str())
53
54 def main():
55     Form, Window = uic.loadUiType("main_window.ui")
56     app = QApplication([])
57     window = Window()
58     form = Form()
59     form.setupUi(window)
60
61     from tableHandlers import TableHandler
62     A_table = TableHandler(form.AMatrix, A_default)
63     A_table.table.itemChanged.connect(TableHandler.floatValidateAndMessage)
64
65     from userInfo import DataGetter
66     dataGetter = DataGetter({'A': A_table})
67
68     graphOptimizer = GraphOptimizer(A_table.matrix)
69
70     form.dataInput.clicked.connect(lambda: (
71         input_data(dataGetter, graphOptimizer, A_table),
72         add_nodes_to_combo(graphOptimizer.levels.keys(), form.nodesCombo),
73         calc_min_tracks(form.nodesCombo.currentText(), graphOptimizer,
          ↪ form.tracksInfo),
74         draw_graph(graphOptimizer, A_table)
75     ))
76     form.nodesCombo.activated.connect(lambda:
      ↪ calc_min_tracks(form.nodesCombo.currentText(), graphOptimizer,
      ↪ form.tracksInfo))
77
78     def showSVG(event):
79         global drawingAllowed
80         if drawingAllowed:
81             svgRenderer = QSvgRenderer('./tmp/graph.svg')
82             painter = QPainter(form.svgView)
83             svgRenderer.render(painter)
84
85     form.svgView.paintEvent = showSVG
86     window.show()
87     app.exec()
88
89 if __name__ == '__main__':
90     main()
```

Листинг 2. userInfo.py

```
1 from PyQt6.QtWidgets import QMessageBox
2
3 def showMessageBox(title: str, description: str, icon: QMessageBox.Icon):
```

```
4       msgBox = QMessageBox()
5       msgBox.setText(title)
6       msgBox.setInformativeText(description)
7       msgBox.setIcon(icon)
8       msgBox.exec()
9
10  class TemplateMessageBox(QMessageBox):
11      def __init__(self, title: str, description: str, icon: QMessageBox.Icon):
12          super().__init__()
13          self.setText(title)
14          self.setInformativeText(description)
15          self.setIcon(icon)
16
17  class DataGetter:
18      def __init__(self, tables: dict, lineEdits: dict = {}, lineEditsLinkedTables:
        ↪ dict = {}):
19          '''
20          tables -> dict(str: TableHandler)
21          lineEdits -> dict(str: QLineEdit)
22          '''
23          self.tables = tables
24          self.lineEdits = lineEdits
25          self.data_good = False
26          self.lineEditsTexts = dict()
27          self.lineEditsLinkedTables = lineEditsLinkedTables
28          self.inputBtnMode = 'input'
29
30      def _get_and_check(self):
31          '''
32          Считывает данные из полей для ввода и выполняет проверки на корректность
            ↪ введённых данных
33          '''
34          for name, table in self.tables.items():
35              table.toNumpy()
36              if not table.data_good:
37                  raise ValueError(f"{name}. Неверно введённые данные")
38
39          for name, lineEdit in self.lineEdits.items():
40              text = lineEdit.text()
41              if len(text) == 0:
42                  raise ValueError(f"{name}. Должно быть введено значение")
43              self.lineEditsTexts[name] = text
44
45
46              if name not in self.lineEditsLinkedTables:
47                  return
48
49              table_name = self.lineEditsLinkedTables[name]
50              theorMin = self.tables[table_name].theorMin
51              if float(text) < theorMin:
52                  raise ValueError(f"{name}. Значение не может быть меньше
                    ↪ теоретического минимума равного {theorMin}")
53
54      def catch_input_errors(self) -> bool:
```

```
55          try:
56              self._get_and_check()
57              msg = TemplateMessageBox("Данные успешно введены", "",
                    ↪ QMessageBox.Icon.Information)
58              msg.exec()
59              return False
60          except ValueError as e:
61              return True
```

Листинг 3. tableHandlers.py

```
1  from PyQt6.QtWidgets import QTableWidget, QTableWidgetItem, QMessageBox
2  from PyQt6.QtGui import QColor
3  from userInfo import TemplateMessageBox
4  import numpy as np
5
6  class TableHandler:
7      valueErrorTitle: str = 'Введены неверные данные!'
8      valueErrorMessages: dict = {
9          'empty_cell': TemplateMessageBox(valueErrorTitle, 'Заполните все ячейки
               ↪ матрицы.', QMessageBox.Icon.Warning),
10         'non_float': TemplateMessageBox(valueErrorTitle, 'Все значения должны быть
               ↪ числовыми.', QMessageBox.Icon.Warning),
11         'negative_value': TemplateMessageBox(valueErrorTitle, 'Значение не может
               ↪ быть отрицательным.', QMessageBox.Icon.Warning)
12     }
13
14     def __init__(self, table: QTableWidget, defaultValues: np.array = None):
15         self.table: QTableWidget = table
16         self.rows = self.table.rowCount()
17         self.columns = self.table.columnCount()
18         self.matrix: np.ndarray = defaultValues
19         self.data_good = True
20         self.theorMin = None
21
22         if self.matrix is None:
23             self.data_good = False
24             return
25
26         if (self.rows, self.columns) != self.matrix.shape:
27             raise ValueError("Table shape does not match passed defaultValues:
                   ↪ np.array shape")
28
29         self.toTable(self.matrix)
30
31     @staticmethod
32     def floatValidate(item: QTableWidgetItem):
33         '''
34         Валидирует значение в соответствии со следующими критериями: значение
               ↪ непустое, вещественное типа float, неотрицательное. Возвращает код
               ↪ результата проверки и само проверенное значение, если оно прошло
               ↪ валидацию.
```

```
35          '''
36          if not item.text():
37              item.setText('0')
38              return ('good', 0)
39
40          try:
41              item_f = float(item.text())
42          except ValueError:
43              return ('non_float', None)
44
45          if item_f < 0:
46              return ('negative_value', None)
47
48          return ('good', item_f)
49
50      @staticmethod
51      def floatValidateAndMessage(item: QTableWidgetItem):
52          '''
53          Валидирует введённое значение и выводит сообщение, если значение не прошло
            ↪ валидацию. Проверяемые критерии определяются floatValidate()
54          '''
55          status, value = TableHandler.floatValidate(item)
56          if status != 'good':
57              TableHandler.valueErrorMessages[status].exec()
58              return ('err', None)
59
60          return ('good', value)
61
62      def toNumpy(self):
63          '''
64          Запись значений из таблицы QTableWidget в матрицу np.array
65          '''
66          matrix = []
67          for i in range(self.table.rowCount()):
68              row = []
69              for j in range(self.table.columnCount()):
70                  item = self.table.item(i, j)
71
72                  status, value = TableHandler.floatValidateAndMessage(item)
73                  if status == 'err':
74                      self.table.setCurrentCell(i, j)
75                      self.data_good = False
76                      return None
77
78                  row += [value,]
79              matrix += [row,]
80
81          np_matrix = np.array(matrix, dtype=np.float16)
82          self.matrix = np_matrix
83          self.data_good = True
84          return np_matrix
85
86      def toTable(self, matrix: np.array):
87          '''
```

```
88          Запись значений из матрицы np.array в таблицу QTableWidget
89          '''
90          if (self.rows, self.columns) != matrix.shape:
91              raise ValueError("Table shape does not match passed np.array shape")
92
93          for i in range(self.rows):
94              for j in range(self.columns):
95                  value = QTableWidgetItem(str(matrix[i,j]))
96                  self.table.setItem(i, j, value)
97
98
```

Листинг 4. GraphOptimizer.py

```
1  import numpy as np
2
3  class GraphOptimizer:
4      def __init__(self, adjacency_matrix: np.array):
5          self.init_from_matrix(adjacency_matrix)
6
7      def init_from_matrix(self, adjacency_matrix: np.array):
8          self.A = adjacency_matrix
9          self.nodes_count = self.A.shape[0]
10
11         if self.A.shape[0] != self.A.shape[1]:
12             raise ValueError('Wrong shape of adjacency_matrix. Matrix should be
               ↪ square')
13
14         self.levels = self.sort_nodes_by_levels()
15
16     def sort_nodes_by_levels(self) -> dict:
17         '''
18         returns dict levels = {
19             <node_name: str>: (<level: int>, <row_in_level: int>)
20         }
21         '''
22
23         levels = dict()
24
25         self.first_nodes = np.where((self.A==0).all(axis=0))[0] + 1
26         levels.update( { str(node): (0, i) for i, node in
           ↪ enumerate(self.first_nodes)} )
27         current_level = 1
28         while len(levels) < self.nodes_count:
29             added_nodes = set()
30             for j in range(self.nodes_count):
31                 if str(j+1) in levels.keys():
32                     continue
33
34                 adj_nodes = np.where(self.A[:,j]!=0)[0] + 1
35                 #Если adj_nodes подмножество levels.keys()
36                 if set(levels.keys()) >= set( map(str, list(adj_nodes)) ):
```

```
                    added_nodes.add(str(j+1))

            levels.update( { str(node): (current_level, i) for i, node in
            ↪  enumerate(added_nodes)} )
            current_level += 1

        self.max_level = current_level - 1
        return levels

    def min_tracks_from(self, node_from_name: str):
        #tracks_info = { <узел>: (<расстояние>, <путь>) }
        self.tracks_info: dict = { node_name: (np.inf, '') for node_name in
        ↪  self.levels.keys() }
        self.tracks_info[node_from_name] = (0, node_from_name)

        for level in range(self.max_level):
            for node_name, placement in self.levels.items():
                if placement[0] != level:
                    continue

                node_ind = int(node_name) - 1
                adj_node_inds = np.where(self.A[node_ind,:] != 0)[0]
                for adj_node_ind in adj_node_inds:
                    price_of_way = self.tracks_info[str(node_ind+1)][0] +
                    ↪  self.A[node_ind, adj_node_ind]
                    way = self.tracks_info[str(node_ind+1)][1] +
                    ↪  f'-{adj_node_ind+1}'

                    if price_of_way < self.tracks_info[str(adj_node_ind+1)][0]:
                        self.tracks_info[str(adj_node_ind+1)] = (price_of_way,
                        ↪  way)

    def min_tracks_info_str(self):
        column1 = 8
        column2 = 10
        column3 = 20
        res = f"{'Узел':>{column1}}  {'Расстояние':>{column2}}  {'Кратчайший
        ↪  путь':>{column3}}\n"
        for node_name, info in sorted(self.tracks_info.items(), key=lambda x:
        ↪  int(x[0])):
            price = info[0]
            min_track = info[1]
            if price < np.inf:
                res += f'{node_name:>{column1}}  {price:>{column2}}
                ↪  {min_track:>{column3}}\n'

        return res

```

Листинг 5. GraphDrawer.py

```
import networkx as nx
import matplotlib.pyplot as plt
```

```
from my_networkx import draw_networkx_curved_edge_labels
import numpy as np

class GraphDrawer:
    def __init__(self, levels: dict, adj_matrix: np.array=None, edges_list: list=None):
        if adj_matrix is None and edges_list is None:
            raise ValueError('Graph should me defined by edges_list or adj_matrix')

        self.G = nx.DiGraph()
        self.levels = levels
        self.picture_drawn = False

        if edges_list is not None:
            self.edges_list = edges_list
        else:
            self.edges_list = self.edges_from_adj_matrix(adj_matrix)

        self.G.add_edges_from(self.edges_list)

        self.pos = {n: self.node_pos(n) for n in self.G.nodes()}
        self.straight_edges, self.curved_edges = self.sort_straight_curved()
        self.edge_weights = nx.get_edge_attributes(self.G, 'weight')

        self.curved_edge_labels = {edge: self.edge_weights[edge] for edge in self.curved_edges}
        self.straight_edge_labels = {edge: self.edge_weights[edge] for edge in self.straight_edges}

    def edges_from_adj_matrix(self, adj_matrix):
        edges_list = []
        nodes = adj_matrix.shape[0]
        for i in range(nodes):
            for j in range(nodes):
                weight = adj_matrix[i,j]
                if weight == 0:
                    continue

                edges_list.append( (str(i+1), str(j+1), {'weight': weight}) )

        return edges_list

    # Функция для определения позиции узла на основе уровня
    def node_pos(self, node):
        x = self.levels[node][0]
        y = -self.levels[node][1]
        return (x, y)

    def sort_straight_curved(self):
        straight_edges = []
        curved_edges = []
        for u, v, _ in self.edges_list:
            #Одинаковый y и располагаются дальше чем на 1 уровень
```

```
53              if self.pos[u][1] == self.pos[v][1] and self.pos[v][0] -
                ↪  self.pos[u][0] > 1:
54                  curved_edges.append((u,v))
55              else:
56                  straight_edges.append((u,v))
57
58          return (straight_edges, curved_edges)
59
60      def draw(self):
61          fig, ax = plt.subplots()
62          fig.set_figwidth(8.5)
63          fig.set_figheight(8.5)
64
65          #Узлы
66          nx.draw_networkx_nodes(self.G, self.pos, ax=ax)
67          nx.draw_networkx_labels(self.G, self.pos, ax=ax)
68
69          #Дуги
70          nx.draw_networkx_edges(self.G, self.pos, ax=ax,
                ↪  edgelist=self.straight_edges)
71          arc_rad = 0.35
72          nx.draw_networkx_edges(self.G, self.pos, ax=ax,
                ↪  edgelist=self.curved_edges,
73                                 connectionstyle=f'arc3, rad = {arc_rad}')
74
75          #Метки
76          draw_networkx_curved_edge_labels(self.G, self.pos, ax=ax,
77                                           edge_labels=self.curved_edge_labels,
                                                 ↪  rotate=False,
78                                           rad=arc_rad)
79          nx.draw_networkx_edge_labels(self.G, self.pos, ax=ax,
                ↪  edge_labels=self.straight_edge_labels,
80                                       rotate=False)
81
82          self.picture_drawn = True
83          return fig, ax
84
85
86  def main():
87      '''
88      Демо работы модуля
89      '''
90      # edges_list = [
91      #     ('A', 'B', {'weight': 10}),
92      #     ('A', 'C', {'weight': 20}),
93      #     ('B', 'D', {'weight': 30}),
94      #     ('A', 'D', {'weight': 20}),
95      #     ('A', 'F', {'weight': 15}),
96      #     ('C', 'E', {'weight': 40}),
97      #     ('D', 'F', {'weight': 50}),
98      #     ('E', 'F', {'weight': 60})
99      # ]
100     A = np.array([
101         [0, 2, 5, 0, 3, 7],
```

```
        [0, 0, 0, 7, 9, 0],
        [0, 0, 0, 2, 1, 0],
        [0, 0, 0, 0, 0, 4],
        [0, 0, 0, 0, 0, 2],
        [0, 0, 0, 0, 0, 0]
    ])

    # Определяем уровни узлов
    levels = {
        '1': (0, 0),
        '2': (1, 0), '3': (1, 1),
        '4': (2, 0), '5': (2, 1),
        '6': (3, 0)
    }

    #graphDrawer = GraphDrawer(edges_list=edges_list, levels=levels)
    graphDrawer = GraphDrawer(levels, adj_matrix=A)
    graphDrawer.draw()

    plt.show()


if __name__ == '__main__':
    main()
```

Листинг 6. my_networkx.py

```
def draw_networkx_curved_edge_labels(
    G,
    pos,
    edge_labels=None,
    label_pos=0.5,
    font_size=10,
    font_color="k",
    font_family="sans-serif",
    font_weight="normal",
    alpha=None,
    bbox=None,
    horizontalalignment="center",
    verticalalignment="center",
    ax=None,
    rotate=True,
    clip_on=True,
    rad=0
):
    """Draw edge labels.

    Parameters
    ----------
    G : graph
        A networkx graph

```

```
    pos : dictionary
        A dictionary with nodes as keys and positions as values.
        Positions should be sequences of length 2.

    edge_labels : dictionary (default={})
        Edge labels in a dictionary of labels keyed by edge two-tuple.
        Only labels for the keys in the dictionary are drawn.

    label_pos : float (default=0.5)
        Position of edge label along edge (0=head, 0.5=center, 1=tail)

    font_size : int (default=10)
        Font size for text labels

    font_color : string (default='k' black)
        Font color string

    font_weight : string (default='normal')
        Font weight

    font_family : string (default='sans-serif')
        Font family

    alpha : float or None (default=None)
        The text transparency

    bbox : Matplotlib bbox, optional
        Specify text box properties (e.g. shape, color etc.) for edge labels.
        Default is {boxstyle='round', ec=(1.0, 1.0, 1.0), fc=(1.0, 1.0, 1.0)}.

    horizontalalignment : string (default='center')
        Horizontal alignment {'center', 'right', 'left'}

    verticalalignment : string (default='center')
        Vertical alignment {'center', 'top', 'bottom', 'baseline',
        ↪  'center_baseline'}

    ax : Matplotlib Axes object, optional
        Draw the graph in the specified Matplotlib axes.

    rotate : bool (deafult=True)
        Rotate edge labels to lie parallel to edges

    clip_on : bool (default=True)
        Turn on clipping of edge labels at axis boundaries

    Returns
    -------
    dict
        'dict' of labels keyed by edge

    Examples
    --------
    >>> G = nx.dodecahedral_graph()
```

```
    >>> edge_labels = nx.draw_networkx_edge_labels(G, pos=nx.spring_layout(G))

    Also see the NetworkX drawing examples at
    https://networkx.org/documentation/latest/auto_examples/index.html

    See Also
    --------
    draw
    draw_networkx
    draw_networkx_nodes
    draw_networkx_edges
    draw_networkx_labels
    """
    import matplotlib.pyplot as plt
    import numpy as np

    if ax is None:
        ax = plt.gca()
    if edge_labels is None:
        labels = {(u, v): d for u, v, d in G.edges(data=True)}
    else:
        labels = edge_labels
    text_items = {}
    for (n1, n2), label in labels.items():
        (x1, y1) = pos[n1]
        (x2, y2) = pos[n2]
        (x, y) = (
            x1 * label_pos + x2 * (1.0 - label_pos),
            y1 * label_pos + y2 * (1.0 - label_pos),
        )
        pos_1 = ax.transData.transform(np.array(pos[n1]))
        pos_2 = ax.transData.transform(np.array(pos[n2]))
        linear_mid = 0.5*pos_1 + 0.5*pos_2
        d_pos = pos_2 - pos_1
        rotation_matrix = np.array([(0,1), (-1,0)])
        ctrl_1 = linear_mid + rad*rotation_matrix@d_pos
        ctrl_mid_1 = 0.5*pos_1 + 0.5*ctrl_1
        ctrl_mid_2 = 0.5*pos_2 + 0.5*ctrl_1
        bezier_mid = 0.5*ctrl_mid_1 + 0.5*ctrl_mid_2
        (x, y) = ax.transData.inverted().transform(bezier_mid)

        if rotate:
            # in degrees
            angle = np.arctan2(y2 - y1, x2 - x1) / (2.0 * np.pi) * 360
            # make label orientation "right-side-up"
            if angle > 90:
                angle -= 180
            if angle < -90:
                angle += 180
            # transform data coordinate angle to screen coordinate angle
            xy = np.array((x, y))
            trans_angle = ax.transData.transform_angles(
                np.array((angle,)), xy.reshape((1, 2))
            )[0]
```

```
        else:
            trans_angle = 0.0
        # use default box of white with white border
        if bbox is None:
            bbox = dict(boxstyle="round", ec=(1.0, 1.0, 1.0), fc=(1.0, 1.0, 1.0))
        if not isinstance(label, str):
            label = str(label)  # this makes "1" and 1 labeled the same

        t = ax.text(
            x,
            y,
            label,
            size=font_size,
            color=font_color,
            family=font_family,
            weight=font_weight,
            alpha=alpha,
            horizontalalignment=horizontalalignment,
            verticalalignment=verticalalignment,
            rotation=trans_angle,
            transform=ax.transData,
            bbox=bbox,
            zorder=1,
            clip_on=clip_on,
        )
        text_items[(n1, n2)] = t

    ax.tick_params(
        axis="both",
        which="both",
        bottom=False,
        left=False,
        labelbottom=False,
        labelleft=False,
    )

    return text_items
```

Листинг 7. main_window.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1055</width>
    <height>808</height>
   </rect>
  </property>
```

```
13   <property name="sizePolicy">
14    <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
15     <horstretch>0</horstretch>
16     <verstretch>0</verstretch>
17    </sizepolicy>
18   </property>
19   <property name="windowTitle">
20    <string>Распределение задач по узлам управления</string>
21   </property>
22   <widget class="QWidget" name="centralwidget">
23    <layout class="QVBoxLayout" name="verticalLayout">
24     <item>
25      <layout class="QHBoxLayout" name="horizontalLayout_4">
26       <item>
27        <spacer name="horizontalSpacer_4">
28         <property name="orientation">
29          <enum>Qt::Horizontal</enum>
30         </property>
31         <property name="sizeHint" stdset="0">
32          <size>
33           <width>40</width>
34           <height>20</height>
35          </size>
36         </property>
37        </spacer>
38       </item>
39       <item>
40        <widget class="QLabel" name="instructionLabel">
41         <property name="font">
42          <font>
43           <pointsize>12</pointsize>
44          </font>
45         </property>
46         <property name="text">
47          <string>Введите данные</string>
48         </property>
49         <property name="scaledContents">
50          <bool>false</bool>
51         </property>
52         <property name="alignment">
53          <set>Qt::AlignCenter</set>
54         </property>
55        </widget>
56       </item>
57       <item>
58        <spacer name="horizontalSpacer_3">
59         <property name="orientation">
60          <enum>Qt::Horizontal</enum>
61         </property>
62         <property name="sizeHint" stdset="0">
63          <size>
64           <width>40</width>
65           <height>20</height>
66          </size>
```

```
67          </property>
68         </spacer>
69        </item>
70       </layout>
71      </item>
72      <item>
73       <widget class="QFrame" name="frame">
74        <property name="frameShape">
75         <enum>QFrame::NoFrame</enum>
76        </property>
77        <property name="frameShadow">
78         <enum>QFrame::Plain</enum>
79        </property>
80        <layout class="QHBoxLayout" name="horizontalLayout">
81         <property name="topMargin">
82          <number>3</number>
83         </property>
84         <property name="bottomMargin">
85          <number>0</number>
86         </property>
87         <item>
88          <spacer name="horizontalSpacer">
89           <property name="orientation">
90            <enum>Qt::Horizontal</enum>
91           </property>
92           <property name="sizeHint" stdset="0">
93            <size>
94             <width>40</width>
95             <height>20</height>
96            </size>
97           </property>
98          </spacer>
99         </item>
100        <item>
101         <layout class="QVBoxLayout" name="verticalLayout_2">
102          <item>
103           <widget class="QLabel" name="cLabel">
104            <property name="sizePolicy">
105             <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
106              <horstretch>0</horstretch>
107              <verstretch>0</verstretch>
108             </sizepolicy>
109            </property>
110            <property name="minimumSize">
111             <size>
112              <width>0</width>
113              <height>35</height>
114             </size>
115            </property>
116            <property name="text">
117             <string>Матрица смежности</string>
118            </property>
119            <property name="alignment">
120             <set>Qt::AlignCenter</set>
```

```
121             </property>
122            </widget>
123           </item>
124           <item>
125            <widget class="QTableWidget" name="AMatrix">
126             <property name="sizePolicy">
127              <sizepolicy hsizetype="Minimum" vsizetype="Minimum">
128               <horstretch>0</horstretch>
129               <verstretch>0</verstretch>
130              </sizepolicy>
131             </property>
132             <property name="minimumSize">
133              <size>
134               <width>0</width>
135               <height>0</height>
136              </size>
137             </property>
138             <property name="verticalScrollBarPolicy">
139              <enum>Qt::ScrollBarAlwaysOff</enum>
140             </property>
141             <property name="horizontalScrollBarPolicy">
142              <enum>Qt::ScrollBarAlwaysOff</enum>
143             </property>
144             <property name="sizeAdjustPolicy">
145              <enum>QAbstractScrollArea::AdjustToContents</enum>
146             </property>
147             <property name="selectionMode">
148              <enum>QAbstractItemView::SingleSelection</enum>
149             </property>
150             <property name="textElideMode">
151              <enum>Qt::ElideLeft</enum>
152             </property>
153             <property name="showGrid">
154              <bool>true</bool>
155             </property>
156             <property name="wordWrap">
157              <bool>false</bool>
158             </property>
159             <property name="cornerButtonEnabled">
160              <bool>true</bool>
161             </property>
162             <property name="rowCount">
163              <number>17</number>
164             </property>
165             <property name="columnCount">
166              <number>17</number>
167             </property>
168             <attribute name="horizontalHeaderVisible">
169              <bool>true</bool>
170             </attribute>
171             <attribute name="horizontalHeaderCascadingSectionResizes">
172              <bool>false</bool>
173             </attribute>
174             <attribute name="horizontalHeaderDefaultSectionSize">
```

```
175              <number>25</number>
176             </attribute>
177             <attribute name="horizontalHeaderMinimumSectionSize">
178              <number>25</number>
179             </attribute>
180             <attribute name="horizontalHeaderStretchLastSection">
181              <bool>false</bool>
182             </attribute>
183             <attribute name="verticalHeaderVisible">
184              <bool>true</bool>
185             </attribute>
186             <attribute name="verticalHeaderDefaultSectionSize">
187              <number>20</number>
188             </attribute>
189             <attribute name="verticalHeaderMinimumSectionSize">
190              <number>20</number>
191             </attribute>
192             <row/>
193             <row/>
194             <row/>
195             <row/>
196             <row/>
197             <row/>
198             <row/>
199             <row/>
200             <row/>
201             <row/>
202             <row/>
203             <row/>
204             <row/>
205             <row/>
206             <row/>
207             <row/>
208             <row/>
209             <column/>
210             <column/>
211             <column/>
212             <column/>
213             <column/>
214             <column/>
215             <column/>
216             <column/>
217             <column/>
218             <column/>
219             <column/>
220             <column/>
221             <column/>
222             <column/>
223             <column/>
224             <column/>
225             <column/>
226             <item row="0" column="0">
227              <property name="text">
228               <string/>
```

```
229             </property>
230             <property name="textAlignment">
231              <set>AlignTrailing|AlignVCenter</set>
232             </property>
233            </item>
234           </widget>
235          </item>
236         </layout>
237        </item>
238        <item>
239         <layout class="QVBoxLayout" name="verticalLayout_3">
240          <property name="spacing">
241           <number>0</number>
242          </property>
243          <item>
244           <widget class="QLabel" name="cLabel_2">
245            <property name="sizePolicy">
246             <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
247              <horstretch>0</horstretch>
248              <verstretch>0</verstretch>
249             </sizepolicy>
250            </property>
251            <property name="minimumSize">
252             <size>
253              <width>0</width>
254              <height>35</height>
255             </size>
256            </property>
257            <property name="text">
258             <string>Последний построенный граф</string>
259            </property>
260            <property name="alignment">
261             <set>Qt::AlignCenter</set>
262            </property>
263           </widget>
264          </item>
265          <item>
266           <widget class="QWidget" name="svgView" native="true">
267            <property name="enabled">
268             <bool>false</bool>
269            </property>
270            <property name="sizePolicy">
271             <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
272              <horstretch>0</horstretch>
273              <verstretch>0</verstretch>
274             </sizepolicy>
275            </property>
276            <property name="minimumSize">
277             <size>
278              <width>450</width>
279              <height>450</height>
280             </size>
281            </property>
282            <property name="maximumSize">
```

```
283             <size>
284              <width>450</width>
285              <height>450</height>
286             </size>
287            </property>
288           </widget>
289          </item>
290         </layout>
291        </item>
292        <item>
293         <spacer name="horizontalSpacer_2">
294          <property name="orientation">
295           <enum>Qt::Horizontal</enum>
296          </property>
297          <property name="sizeHint" stdset="0">
298           <size>
299            <width>40</width>
300            <height>20</height>
301           </size>
302          </property>
303         </spacer>
304        </item>
305       </layout>
306      </widget>
307     </item>
308     <item>
309      <layout class="QHBoxLayout" name="horizontalLayout_5">
310       <item>
311        <spacer name="horizontalSpacer_7">
312         <property name="orientation">
313          <enum>Qt::Horizontal</enum>
314         </property>
315         <property name="sizeHint" stdset="0">
316          <size>
317           <width>40</width>
318           <height>20</height>
319          </size>
320         </property>
321        </spacer>
322       </item>
323       <item>
324        <widget class="QPushButton" name="dataInput">
325         <property name="minimumSize">
326          <size>
327           <width>30</width>
328           <height>30</height>
329          </size>
330         </property>
331         <property name="maximumSize">
332          <size>
333           <width>999999</width>
334           <height>16777215</height>
335          </size>
336         </property>
```

```
337          <property name="layoutDirection">
338           <enum>Qt::LeftToRight</enum>
339          </property>
340          <property name="text">
341           <string>Ввод данных</string>
342          </property>
343         </widget>
344        </item>
345        <item>
346         <spacer name="horizontalSpacer_8">
347          <property name="orientation">
348           <enum>Qt::Horizontal</enum>
349          </property>
350          <property name="sizeHint" stdset="0">
351           <size>
352            <width>40</width>
353            <height>20</height>
354           </size>
355          </property>
356         </spacer>
357        </item>
358       </layout>
359      </item>
360      <item>
361       <widget class="Line" name="line">
362        <property name="orientation">
363         <enum>Qt::Horizontal</enum>
364        </property>
365       </widget>
366      </item>
367      <item>
368       <layout class="QVBoxLayout" name="verticalLayout_4">
369        <property name="spacing">
370         <number>0</number>
371        </property>
372        <property name="leftMargin">
373         <number>300</number>
374        </property>
375        <property name="rightMargin">
376         <number>300</number>
377        </property>
378        <item>
379         <widget class="QLabel" name="cLabel_3">
380          <property name="sizePolicy">
381           <sizepolicy hsizetype="Preferred" vsizetype="Minimum">
382            <horstretch>0</horstretch>
383            <verstretch>0</verstretch>
384           </sizepolicy>
385          </property>
386          <property name="minimumSize">
387           <size>
388            <width>0</width>
389            <height>35</height>
390           </size>
```

```
391          </property>
392          <property name="text">
393           <string>Минимизация расстояний</string>
394          </property>
395          <property name="alignment">
396           <set>Qt::AlignCenter</set>
397          </property>
398         </widget>
399        </item>
400        <item>
401         <layout class="QHBoxLayout" name="horizontalLayout_2">
402          <item>
403           <spacer name="horizontalSpacer_5">
404            <property name="orientation">
405             <enum>Qt::Horizontal</enum>
406            </property>
407            <property name="sizeHint" stdset="0">
408             <size>
409              <width>40</width>
410              <height>20</height>
411             </size>
412            </property>
413           </spacer>
414          </item>
415          <item>
416           <widget class="QLabel" name="label">
417            <property name="sizePolicy">
418             <sizepolicy hsizetype="Expanding" vsizetype="Preferred">
419              <horstretch>0</horstretch>
420              <verstretch>0</verstretch>
421             </sizepolicy>
422            </property>
423            <property name="maximumSize">
424             <size>
425              <width>150</width>
426              <height>16777215</height>
427             </size>
428            </property>
429            <property name="text">
430             <string>Начальный узел</string>
431            </property>
432           </widget>
433          </item>
434          <item>
435           <widget class="QComboBox" name="nodesCombo">
436            <property name="enabled">
437             <bool>false</bool>
438            </property>
439            <property name="sizePolicy">
440             <sizepolicy hsizetype="Expanding" vsizetype="Fixed">
441              <horstretch>0</horstretch>
442              <verstretch>0</verstretch>
443             </sizepolicy>
444            </property>
```

```
445             <property name="maximumSize">
446              <size>
447               <width>150</width>
448               <height>16777215</height>
449              </size>
450             </property>
451            </widget>
452           </item>
453           <item>
454            <spacer name="horizontalSpacer_6">
455             <property name="orientation">
456              <enum>Qt::Horizontal</enum>
457             </property>
458             <property name="sizeHint" stdset="0">
459              <size>
460               <width>40</width>
461               <height>20</height>
462              </size>
463             </property>
464            </spacer>
465           </item>
466          </layout>
467         </item>
468        </layout>
469       </item>
470       <item>
471        <layout class="QHBoxLayout" name="horizontalLayout_3">
472         <item>
473          <widget class="QTextBrowser" name="tracksInfo">
474           <property name="sizePolicy">
475            <sizepolicy hsizetype="Expanding" vsizetype="Minimum">
476             <horstretch>0</horstretch>
477             <verstretch>0</verstretch>
478            </sizepolicy>
479           </property>
480           <property name="minimumSize">
481            <size>
482             <width>0</width>
483             <height>100</height>
484            </size>
485           </property>
486           <property name="maximumSize">
487            <size>
488             <width>500</width>
489             <height>200</height>
490            </size>
491           </property>
492           <property name="font">
493            <font>
494             <family>Monospac821 BT</family>
495            </font>
496           </property>
497           <property name="html">
```

```
498          <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
              ↪  &quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
499 &lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot; content=&quot;1&quot;
    ↪  /&gt;&lt;style type=&quot;text/css&quot;&gt;
500 p, li { white-space: pre-wrap; }
501 &lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'Monospac821 BT';
    ↪  font-size:8.064pt; font-weight:400; font-style:normal;&quot;&gt;
502 &lt;p style=&quot; margin-top:0px; margin-bottom:0px; margin-left:0px;
    ↪  margin-right:0px; -qt-block-indent:0; text-indent:0px;&quot;&gt;&lt;span
    ↪  style=&quot; font-family:'MS Shell Dlg 2';&quot;&gt;Данные не
    ↪  введены&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
503          </property>
504         </widget>
505        </item>
506       </layout>
507      </item>
508     </layout>
509    </widget>
510    <widget class="QMenuBar" name="menubar">
511     <property name="geometry">
512      <rect>
513       <x>0</x>
514       <y>0</y>
515       <width>1055</width>
516       <height>27</height>
517      </rect>
518     </property>
519    </widget>
520    <widget class="QStatusBar" name="statusbar"/>
521   </widget>
522   <resources/>
523   <connections/>
524 </ui>
```