# Лабораторные работы по программированию и основам алгоритмизации

Тутубалин П.И., Губайдуллин Ш.И.

ПМИ КНИТУ-КАИ

Материал подготовлен студентом группы 4112, Наумихиным Д.Д.

# Содержание

# 1 Собственные статические библиотеки

В лабораторных работах используются следующие библиотеки: MIO и MDialogs

## 1.1 MIO

Эта библиотека предназначена для обозначения собственных исключений (MExc) и обработки пользовательского ввода (MIO):

Листинг 1: Исключения, MExc.h

```h
#ifndef LABS_MEXC_H
#define LABS_MEXC_H

#include <exception>

namespace MExc {
    class NotThatType : public std::exception {
    public:
        [[nodiscard]] const char *what() const noexcept override;
    };

    class ThereIsNothing : public std::exception {
    public:
        [[nodiscard]] const char *what() const noexcept override;
    };
}

#endif //LABS_MEXC_H
```

Листинг 2: Исключения, MExc.cpp

```cpp
#include "MExc.h"
#include <vector>

const char *MExc::NotThatType::what() const noexcept {
    return "The type of input is not correct.";
}

const char *MExc::ThereIsNothing::what() const noexcept {
    return "There is nothing";
}
```

Листинг 3: Пользовательский ввод, MIO.h

```h
#ifndef LABS_MIO_H
#define LABS_MIO_H

#include <iostream>
#include <bitset>
```

```
6  #include "MExc.h"
7
8  namespace MIO {
9      uint8_t inputStreamLL(int64_t &x, std::istream &in = std::cin,
           std::ostream &eout = std::cerr, bool printErr = true);
10     uint8_t inputBool(bool &b, std::istream &in = std::cin,
           std::ostream &eout = std::cerr, bool printErr = true);
11     uint8_t inputULL(uint64_t &x, std::istream &in = std::cin,
           std::ostream &eout = std::cerr, bool printErr = true);
12     uint8_t inputNatural(uint64_t &x, std::istream &in = std::cin,
           std::ostream &eout = std::cerr, bool printErr = true);
13     uint8_t chooseVariant(uint64_t &x, uint64_t border,
           std::istream &in = std::cin, std::ostream &eout =
           std::cerr, bool printErr = true);
14
15     template<size_t T> uint8_t inputBS(std::bitset<T> &x,
           std::istream &in, std::ostream &eout, bool printErr) {
16         bool f = true;
17         std::bitset<T> y;
18         do {
19             try {
20                 if (in.eof()) return 1;
21                 if ((in >> y).fail()) throw MExc::NotThatType();
22                 x = y;
23                 f = false;
24             } catch (std::exception &err) {
25                 if (printErr) eout << err.what() << "\n" << "Write
                       bit sequence, please" << std::endl;
26                 in.clear();
27                 if (in.eof()) return 1;
28                 std::string tmp;
29                 in >> tmp;
30             }
31         } while (f);
32         return 0;
33     }
34 }
35 #endif //LABS_MIO_H
```

Листинг 4: Пользовательский ввод, MIO.cpp

```
1  #include "MIO.h"
2  #include <string>
3
4  uint8_t MIO::inputStreamLL(int64_t &x, std::istream &in,
       std::ostream &eout, bool printErr) {
5      bool f = true;
6      do {
```

```
7           try {
8               if (in.eof()) return 1;
9               int64_t tmp;
10              if ((in >> tmp).fail()) throw MExc::NotThatType();
11              x = tmp;
12              f = false;
13          } catch (std::exception &err) {
14              if (printErr) eout << err.what() << "\n" << "Write an
                    integer, please\n" << std::endl;
15              in.clear();
16              if (in.eof()) return 1;
17              std::string tmp;
18              in >> tmp;
19          }
20      } while (f);
21      return 0;
22  }
23
24  uint8_t MIO::inputULL(uint64_t &x, std::istream &in, std::ostream
        &eout, bool printErr) {
25      bool f = true;
26      std::string s;
27      do {
28          try {
29              if (in.eof()) return 1;
30              getline(in, s);
31              if (s.find('-') != std::string::npos) throw
                    MExc::NotThatType();
32              x = stoul(s);
33              f = false;
34          } catch (std::exception &err) {
35              if (printErr) eout << err.what() << "\n" << "Write an
                    unsigned integer, please" << std::endl;
36          }
37      } while (f);
38      return 0;
39  }
40
41  uint8_t MIO::inputNatural(uint64_t &x, std::istream &in,
        std::ostream &eout, bool printErr) {
42      bool f = true;
43      std::string s;
44      do {
45          try {
46              uint64_t tmp;
47              if (MIO::inputULL(tmp, in, eout, printErr)) return 1;
48              if (tmp == 0) throw MExc::NotThatType();
```

```
49              x = tmp;
50              f = false;
51          } catch (std::exception &err) {
52              if (printErr) eout << err.what() << "\n" << "Write a
                    natural number, please" << std::endl;
53          }
54      } while (f);
55      return 0;
56  }
57
58  uint8_t MIO::inputBool(bool &b, std::istream &in, std::ostream
        &eout, bool printErr) {
59      bool f = true;
60      std::string s;
61      do {
62          try {
63              if (in.eof()) return 1;
64              getline(in, s);
65              s.erase(remove_if(s.begin(), s.end(), ::isspace),
                    s.end());
66              if (!(s == "1"  s == "0")) throw MExc::NotThatType();
67              b = (s == "1");
68              f = false;
69          } catch (std::exception &err) {
70              if (printErr && !s.empty())
71                  eout << err.what() << "\n" << "Write a boolean
                        value (0, 1), please" << std::endl;
72          }
73      } while (f);
74      return 0;
75  }
76
77  uint8_t MIO::chooseVariant(uint64_t &x, uint64_t border,
        std::istream &in, std::ostream &eout, bool printErr) {
78      bool f = true;
79      std::string s;
80      do {
81          try {
82              uint64_t tmp;
83              if (MIO::inputNatural(tmp, in, eout, printErr)) return
                    1;
84              if (tmp > border) throw MExc::NotThatType();
85              x = tmp;
86              f = false;
87          } catch (std::exception &err) {
88              if (printErr)
89                  eout << err.what() << "\n" << "Write a natural
```

```
                    number not greater than " << border <<
                    std::endl;
90          }
91      } while (f);
92      return 0;
93  }
```

## 1.2 MDialogs

В этой библиотеке представлены средства для ведения диалога с пользователем. Эта библиотека публично зависит от MIO (это значит, что файлы из MIO нужны как для этой библиотеки, так и для пользователей этой библиотеки):

```
project(MDialogs)

set(SRCS
    dialog.cpp
)
add_library(MDialogs STATIC ${SRCS})
target_link_libraries(MDialogs PUBLIC MIO)
```

Листинг 5: dialog.h

```
1   #ifndef LABS_DIALOG_H
2   #define LABS_DIALOG_H
3
4   #include <iostream>
5   #include <fstream>
6   #include <functional>
7   #include "MIO.h"
8   #include "MExc.h"
9   #include <string>
10
11
12  struct Dialog {
13      struct Ioeqpp {
14          std::istream &in;
15          std::ostream &out, &eout, &qout;
16          bool printErr, printQst;
17      };
18
19      virtual void ioXY(Ioeqpp &iqp) = 0;
20
21      void consoleI(std::ostream &out);
22
```

```
23      void fileI(std::ostream &out);
24
25      int step();
26
27      template<typename T=std::ifstream>
28      static T fileD(Ioeqpp &iqp) {
29          bool f = true;
30          T fio;
31          std::string name = ((typeid(T) == typeid(std::ifstream)) ?
               "in.txt" : "out.txt");
32          if (typeid(T) != typeid(std::ifstream) && typeid(T) !=
               typeid(std::ofstream)) throw MExc::NotThatType();
33          if (iqp.printQst)
34              iqp.qout << "Write name of file " << "(" << name << "
                   as default" << ")" << ":" << std::endl;
35          do {
36              try {
37                  std::getline(iqp.in, name);
38                  if (name.empty())
39                      name = ((typeid(T) == typeid(std::ifstream)) ?
                           "in.txt" : "out.txt");
40                  fio.open(name);
41                  if (fio.fail()) throw MExc::ThereIsNothing();
42                  f = false;
43              } catch (std::exception &err) {
44                  if (iqp.printErr) iqp.eout << err.what() << "\n"
                       << "Write name of file, please" << std::endl;
45
46              }
47          } while (f);
48          return fio;
49      }
50
51      template<class T>
52      struct FuncWithDesc {
53          std::string desc;
54          std::function<T> func;
55      };
56
57      template<class T>
58      static uint64_t optionD(std::vector<FuncWithDesc<T>> &options,
           Ioeqpp &iqp) {
59          if (iqp.printQst) {
60              iqp.qout << "Choose an option:\n";
61              for (int i = 0; i < options.size(); ++i) {
62                  iqp.qout << i + 1 << " - " << options[i].desc <<
                       ";\n";
```

```
63              }
64          }
65          uint64_t option;
66          MIO::chooseVariant(option, options.size(), iqp.in,
                iqp.eout, iqp.printErr);
67          return option - 1;
68      }
69
70      virtual ~Dialog() = default;
71  };
72
73  #endif //LABS_DIALOG_H
```

Листинг 6: dialog.cpp

```
1   #include "dialog.h"
2
3   void Dialog::consoleI(std::ostream &out) {
4       Ioeqpp iqp = {std::cin, out, std::cerr, std::cout, true, true};
5       ioXY(iqp);
6   }
7
8
9   void Dialog::fileI(std::ostream &out) {
10      Ioeqpp iqp1 = {std::cin, out, std::cerr, std::cout, true,
            true};
11      std::ifstream fin = fileD(iqp1);
12      Ioeqpp iqp2 = {fin, out, std::cerr, out, false, false};
13      while (!fin.eof())
14          ioXY(iqp2);
15  }
16
17
18  int Dialog::step() {
19      Ioeqpp iqp1 = {std::cin, std::cout, std::cerr, std::cout,
            true, true};
20      bool cI, cO;
21      iqp1.qout << "What would you use for input? 1 - Console, 0 -
            file: " << std::endl;
22      MIO::inputBool(cI, iqp1.in, iqp1.eout, true);
23      iqp1.qout << "What would you use for output? 1 - Console, 0 -
            file: " << std::endl;
24      MIO::inputBool(cO, iqp1.in, iqp1.eout, true);
25
26      std::ofstream file;
27      if (!cO) file = fileD<std::ofstream>(iqp1);
28      std::ostream &out = (cO ? std::cout : file);
29
```

```
30      if (cI) {
31          consoleI(out);
32          return 1;
33      } else fileI(out);
34      return 0;
35  }
```

# 2 version.h

В файле version.h после сборки находится информация насчёт версии сборки и хеша коммита. В дереве можно найти шаблон для этого файла «version.h.in», который будет заполняться при помощи cmake.

Листинг 7: version.h.in

```
#ifndef VERSION_H
#define VERSION_H

#define GIT_COMMIT_HASH "@GIT_COMMIT_HASH@"
#define LABS_VERSION "@CMAKE_PROJECT_VERSION@"
#define LABS_VERSION_MAJOR @CMAKE_PROJECT_VERSION_MAJOR@
#define LABS_VERSION_MINOR @CMAKE_PROJECT_VERSION_MINOR@

#endif
```

Вот пример готового хедера:

Листинг 8: version.h

```
#ifndef VERSION_H
#define VERSION_H

#define GIT_COMMIT_HASH "8197905"
#define LABS_VERSION "1.0"
#define LABS_VERSION_MAJOR 1
#define LABS_VERSION_MINOR 0

#endif
```

# 3 Лабораторная работа 1

## 3.1 Задание

Дано натуральное число $n$. Вычислить значения функции

$$y = \frac{x^2 - 3x + 2}{\sqrt{2x^3 - 1}}$$

для $x = 1, 1.1, 1.2, \ldots, 1 + 0.1n$.

## 3.2 Требования

1. Дружественный интерфейс.
2. Возможность многократного ввода исходных данных необходимых для решения поставленной задачи. Программа должна спрашивать: «Хотите повторить ввод исходных данных? Да — 1, Нет — 0.» Также в некотором виде должен формироваться запрос(ы), определяющие откуда поступят исходные данные и куда будет осуществлён вывод результата.
3. Использование минимум одной функции помимо функции main.
4. Ввод исходных данных из консоли.
5. Вывод результатов в файл (путь к файлу задаётся в коде).
6. Вывод результатов в консоль.
7. В случае ввода данных из файла программа должна завершаться (не предлагать повторно ввести исходные данные).
8. Использование uniform инициализации c++.
9. Защиту от некорректного пользовательского ввода. При этом следует продумать возможные случаи такого некорректного ввода. Перечислить примеры возможных вариантов некорректного пользовательского ввода.
10. Размещение пользовательских констант в отдельном файле, например, ”constants.h”с «header guards».
11. Размещение прототипов пользовательских функций в отдельном файла, например, ”myfuncs.h”с «header guards». Допускается несколько таких файлов.
12. Размещение описаний пользовательских функций в отдельном файла, например, ”myfuncs.cpp”. Допускается несколько таких файлов.
13. Передачу статических массивов в функции(ю) в качестве параметров.

## 3.3 Листинги исходных файлов программы

Листинг 9: main.cpp

```
#include "myfuncs.h"
#include "version.h"

int main() {
    std::cout << "Currently you use the " << LABS_VERSION << "
       version of the Labs.\n" << "Git commit hash: " <<
       GIT_COMMIT_HASH << "." << std::endl;
    MDialog dialog{};
    bool DS = true;
    while (DS) {
        if (!dialog.step()) break;
        std::cout << "Repeat? 1 - Yes, 0 - No: " << std::endl;
        MIO::inputBool(DS, std::cin, std::cerr);
    }
    return 0;
}
```

Листинг 10: myfuncs.h

```
#ifndef LAB1_MYFUNCS_H
#define LAB1_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
#include "dialog.h"

struct MDialog : public Dialog {
    void ioXY(Ioeqpp &iqp) override;
};

#endif//LAB1_MYFUNCS_H
```

Листинг 11: myfuncs.cpp

```
#include "myfuncs.h"
#include "constants.h"
#include "MExc.h"
#include "MIO.h"
#include <iostream>
#include <string>
#include "dialog.h"

namespace {
    double y(double x) {
```

```
11          return (x * x - 3 * x + 2) / (sqrt(2 * x * x * x - 1));
12      }
13
14      std::pair<double *, size_t> iXY(Dialog::Ioeqpp &iqp) {
15          size_t n = 0;
16          static double arr[arrsize];
17          if (iqp.printQst) iqp.qout << "n" << std::endl;
18          try {
19              if (MIO::inputNatural(n, iqp.in, iqp.eout,
                    iqp.printErr) != 0) throw MExc::ThereIsNothing();
20              for (int i = 0; i <= n; ++i) {
21                  arr[i] = y(1 + 0.1 * i);
22              }
23          } catch (std::exception &err) {
24              if (iqp.printErr) iqp.eout << err.what() << std::endl;
25          }
26          return {arr, n};
27      }
28
29      void oXY(double inp[], size_t n, std::ostream &out) {
30          out << "x\ty" << std::endl;
31          for (uint64_t i = 0; i <= n; ++i) {
32              out << 1 + 0.1 * (double)i << "\t" << inp[i] <<
                    std::endl;
33          }
34      }
35  }
36
37  void MDialog::ioXY(Ioeqpp &iqp) {
38      auto [arr, n] = iXY(iqp);
39      if (n != 0) oXY(arr, n, iqp.out);
40  }
```

Листинг 12: constants.h

```
1  #ifndef LAB1_CONSTANTS_H
2  #define LAB1_CONSTANTS_H
3
4  #include <cstdint>
5
6  constexpr size_t arrsize = 100500;
7
8  #endif//LAB1_CONSTANTS_H
```

## 3.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль

```
What would you use for input? 1 - Console, 0 - file:
1
What would you use for output? 1 - Console, 0 - file:
1
n
8
x        y
1        0
1.1      -0.0698115
1.2      -0.102095
1.3      -0.113989
1.4      -0.113288
1.5      -0.104257
1.6      -0.0894925
1.7      -0.0706866
1.8      -0.0489959
Repeat? 1 - Yes, 0 - No:
1
What would you use for input? 1 - Console, 0 - file:
1
What would you use for output? 1 - Console, 0 - file:
1
n
2
x        y
1        0
1.1      -0.0698115
1.2      -0.102095
Repeat? 1 - Yes, 0 - No:
0
```

2. Консоль $\longrightarrow$ файл

консоль

```
What would you use for input? 1 - Console, 0 - file:
1
What would you use for output? 1 - Console, 0 - file:
0
n
10
Repeat? 1 - Yes, 0 - No:
0
```

out.txt

```
x   y
1   0
```

```
3  1.1  -0.0698115
4  1.2  -0.102095
5  1.3  -0.113989
6  1.4  -0.113288
7  1.5  -0.104257
8  1.6  -0.0894925
9  1.7  -0.0706866
10 1.8  -0.0489959
11 1.9  -0.0252367
12 2    0
```

3. Файл ⟶ консоль

in.txt

```
1 8
2 5
3 1
```

консоль

```
1  What would you use for input? 1 - Console, 0 - file:
2  0
3  What would you use for output? 1 - Console, 0 - file:
4  1
5  x        y
6  1        0
7  1.1      -0.0698115
8  1.2      -0.102095
9  1.3      -0.113989
10 1.4      -0.113288
11 1.5      -0.104257
12 1.6      -0.0894925
13 1.7      -0.0706866
14 1.8      -0.0489959
15 x        y
16 1        0
17 1.1      -0.0698115
18 1.2      -0.102095
19 1.3      -0.113989
20 1.4      -0.113288
21 1.5      -0.104257
22 x        y
23 1        0
24 1.1      -0.0698115
```

4. Файл ⟶ файл

in.txt

```
8
5
1
```

консоль

```
What would you use for input? 1 - Console, 0 - file:
0
What would you use for output? 1 - Console, 0 - file:
0
```

out.txt

```
x   y
1   0
1.1 -0.0698115
1.2 -0.102095
1.3 -0.113989
1.4 -0.113288
1.5 -0.104257
1.6 -0.0894925
1.7 -0.0706866
1.8 -0.0489959
x   y
1   0
1.1 -0.0698115
1.2 -0.102095
1.3 -0.113989
1.4 -0.113288
1.5 -0.104257
x   y
1   0
1.1 -0.0698115
```

## 3.5 Неккорктный ввод

1. Здесь приведены примеры обработки неправильного ввода данных со стороны пользователя (консоль):

консоль

```
What would you use for input? 1 - Console, 0 - file:
NotABoolean
The type of input is not correct.
Write a boolean value (0, 1), please
1.0
The type of input is not correct.
Write a boolean value (0, 1), please
```

```
-1
The type of input is not correct.
Write a boolean value (0, 1), please
1
What would you use for output? 1 - Console, 0 - file:
1
n
NotAnUInt
stoul
Write an unsigned integer, please
-1
The type of input is not correct.
Write an unsigned integer, please
0
The type of input is not correct.
Write a natural number, please
1.0
x       y
1       0
1.1     -0.0698115
Repeat? 1 - Yes, 0 - No:
0
```

2. Неправильный ввод (файл):

in.txt

```
NotAnUInt
-1
0
1.0
```

консоль

```
What would you use for input? 1 - Console, 0 - file:
0
What would you use for output? 1 - Console, 0 - file:
1
x       y
1       0
1.1     -0.0698115
```

3. Неправильный ввод (пустой файл):

in.txt

консоль

```
What would you use for input? 1 - Console, 0 - file:
```

```
2 0
3 What would you use for output? 1 - Console, 0 - file:
4 1
5 There is nothing
```

# 4 Лабораторная работа 2

## 4.1 Задание

Дано натуральное число n. Получить все его натуральные делители.

## 4.2 Требования

1. Требования 1-12 из лабораторной работы 1.
2. Передачу статических массивов в функции(ю) в качестве параметров.

## 4.3 Листинги исходных файлов программы

Листинг 13: main.cpp

```
#include "myfuncs.h"
#include "version.h"

int main() {
    std::cout << "Currently you use the " << LABS_VERSION << "
       version of the Labs.\n" << "Git commit hash: " <<
       GIT_COMMIT_HASH << "." << std::endl;
    MDialog dialog{};
    bool DS = true;
    while (DS) {
        if (!dialog.step()) break;
        std::cout << "Repeat? 1 - Yes, 0 - No: " << std::endl;
        MIO::inputBool(DS, std::cin, std::cerr);
    }
    return 0;
}
```

Листинг 14: myfuncs.h

```
#ifndef LAB2_MYFUNCS_H
#define LAB2_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
#include "dialog.h"

struct MDialog : public Dialog {
    void ioXY(Ioeqpp &iqp) override;
};

#endif//LAB2_MYFUNCS_H
```

Листинг 15: myfuncs.cpp

```
#include "myfuncs.h"
#include "constants.h"
#include "MExc.h"
#include "MIO.h"
#include <string>
#include "dialog.h"


namespace {
    std::pair<uint64_t *, uint64_t> iXY(Dialog::Ioeqpp &iqp) {
        uint64_t n = -1, kd = 0;
        static uint64_t arr[arrsize];
        if (iqp.printQst) iqp.qout << "n" << std::endl;
        if (MIO::inputNatural(n, iqp.in, iqp.eout, iqp.printErr)
           == 0) {
            for (int i = 1; i <= n; ++i) {
                if (n % i == 0) arr[kd++] = i;
            }
        } else {
            iqp.eout << "There is nothing" << std::endl;
        }
        return {arr, kd};
    }

    void oXY(uint64_t *inp, uint64_t n, std::ostream &out) {
        out << "Natural divisors of a number " << inp[n - 1] << "
           are:\n";
        for (int i = 0; i < n; ++i) {
            out << inp[i] << " ";
        }
        out << std::endl;
    }
}

void MDialog::ioXY(Ioeqpp &iqp) {
    auto [arr, n] = iXY(iqp);
    if (n != 0) oXY(arr, n, iqp.out);
}
```

Листинг 16: constants.h

```
#ifndef LAB2_CONSTANTS_H
#define LAB2_CONSTANTS_H

#include <cstdint>

constexpr size_t arrsize = 100500;
```

```
7
8 #endif//LAB2_CONSTANTS_H
```

## 4.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль

```
1  What would you use for input? 1 - Console, 0 - file:
2  1
3  What would you use for output? 1 - Console, 0 - file:
4  1
5  n
6  2596
7  Natural divisors of a number 2596 are:
8  1 2 4 11 22 44 59 118 236 649 1298 2596
9  Repeat? 1 - Yes, 0 - No:
10 0
```

# 5 Лабораторная работа 3

## 5.1 Задание

Дана целочисленная квадратная матрица порядка $n$. Найти номера строк:

a) все элементы которых - нули;

b) элементы в каждой из которых одинаковы;

c) все элементы которых четны;

d) элементы каждой из которых образуют монотонную последовательность (монотонно убывающую или монотонно возрастающую);

e) элементы которых образуют симметричные последовательности (палиндромы).

## 5.2 Требования

1. Требования 1-12 из лабораторной работы 1.
2. Динамическое выделение памяти.
3. Передачу динамических массивов в качестве параметров функции.

## 5.3 Листинги исходных файлов программы

Листинг 17: main.cpp

```
#include "myfuncs.h"
#include "version.h"

int t();

int main() {
    std::cout << "Currently you use the " << LABS_VERSION << "
       version of the Labs.\n" << "Git commit hash: " <<
       GIT_COMMIT_HASH << "." << std::endl;
    MDialog dialog{};
    bool DS = true;
    while (DS) {
        if (!dialog.step()) break;
        std::cout << "Repeat? 1 - Yes, 0 - No: " << std::endl;
        MIO::inputBool(DS, std::cin, std::cerr);
    }
    return 0;
}
```

Листинг 18: myfuncs.h

```
#ifndef LAB3_MYFUNCS_H
#define LAB3_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
#include "dialog.h"

struct MDialog : public Dialog {
    void ioXY(Ioeqpp &iqp) override;
};

#endif//LAB3_MYFUNCS_H
```

Листинг 19: myfuncs.cpp, общая часть

```
#include "myfuncs.h"
#include "constants.h"
#include "MExc.h"
#include "MIO.h"
#include <stdint.h>
#include <string>
#include <vcruntime.h>
#include "dialog.h"

#if defined USE_VECTOR
#include "Matrix.h"
```

Различные реализации:

Листинг 20: USE VECTOR, myfuncs.cpp

```
namespace {
    Matrix iXY(Dialog::Ioeqpp &iqp) {
        try {
            if (iqp.printQst) iqp.qout << "n" << std::endl;
            size_t n;
            if (MIO::inputULL(n, iqp.in, iqp.eout, iqp.printErr)
               != 0) throw MExc::ThereIsNothing();
            Matrix mrx{n, n};
            if (iqp.printQst) iqp.qout << "Matrix nxn" <<
               std::endl;
            for (size_t i = 0; i < n; ++i) {
                for (size_t j = 0; j < n; ++j) {
                    if (MIO::inputStreamLL(mrx[i][j], iqp.in,
                       iqp.eout, iqp.printErr) != 0)
                        throw MExc::ThereIsNothing();
                }
```

```
14              }
15              return mrx;
16          } catch (std::exception &err) {
17              iqp.eout << err.what() << std::endl;
18          }
19          return {0, 0};
20      }
21
22      void oXY(const std::vector<int64_t> &vr, const std::string
           &exc, std::ostream &out) {
23          out << exc << ") ";
24          for (size_t i = 1, endi = vr.size(); i <= endi; ++i) {
25              bool f = false;
26              for (size_t j = 0, endj = vr.size(); j < endj; ++j) {
27                  if (vr[j] == i) {
28                      out << j + 1 << " ";
29                      f = true;
30                  }
31              }
32              if (f) out << "\t";
33          }
34          out << std::endl;
35      }
36  }
37  void MDialog::ioXY(Dialog::Ioeqpp &iqp) {
38      Matrix mrx = iXY(iqp);
39
40      auto ans = std::vector<int64_t>(mrx.cols());
41      for (size_t i = 0, n = ans.size(); i < n; ++i) {
42          ans[i] = 1;
43          for (size_t j = 0; j < n; ++j) {
44              ans[i] = ans[i] && (mrx[i][j] == 0);
45          }
46      }
47      oXY(ans, "a", iqp.out);
48
49      ans = std::vector<int64_t>(mrx.cols());
50      for (size_t i = 0, n = ans.size(); i < n; ++i) {
51          ans[i] = 0;
52          for (size_t j = 0; j < n; ++j) {
53              bool f = true;
54              for (size_t k = 0; k < n; ++k) {
55                  f = f && (mrx[i][k] == mrx[j][k]);
56              }
57              if (f) ans[j] = i + 1;
58          }
59      }
```

```
60      oXY(ans, "b", iqp.out);
61
62      ans = std::vector<int64_t>(mrx.cols());
63      for (size_t i = 0, n = ans.size(); i < n; ++i) {
64          ans[i] = 1;
65          for (size_t j = 0; j < n; ++j) {
66              ans[i] = ans[i] && (mrx[i][j] % 2 == 0);
67          }
68      }
69      oXY(ans, "c", iqp.out);
70
71      ans = std::vector<int64_t>(mrx.cols());
72      for (size_t i = 0, n = ans.size(); i < n; ++i) {
73          ans[i] = 1;
74          for (size_t j = 1; j < n; ++j) {
75              ans[i] = ans[i] && ((mrx[i][0] < mrx[i][n - 1] &&
                    mrx[i][j - 1] < mrx[i][j])
76                                  (mrx[i][0] > mrx[i][n - 1] &&
                                       mrx[i][j - 1] > mrx[i][j]));
77          }
78      }
79      oXY(ans, "d", iqp.out);
80
81      ans = std::vector<int64_t>(mrx.cols());
82      for (size_t i = 0, n = ans.size(); i < n; ++i) {
83          ans[i] = 1;
84          for (size_t j = 0; j < n; ++j) {
85              ans[i] = ans[i] && (mrx[i][j] == mrx[i][n - j - 1]);
86          }
87      }
88      oXY(ans, "e", iqp.out);
89  }
```

Листинг 21: USE ONLY MATRIX, myfuncs.cpp

```
1   #include "Matrix.h"
2   namespace {
3       Matrix iXY(Dialog::Ioeqpp &iqp) {
4           try {
5               if (iqp.printQst) iqp.qout << "n" << std::endl;
6               size_t n;
7               if (MIO::inputULL(n, iqp.in, iqp.eout, iqp.printErr)
                    != 0) throw MExc::ThereIsNothing();
8               Matrix mrx{n, n};
9               if (iqp.printQst) iqp.qout << "Matrix nxn" <<
                    std::endl;
10              for (size_t i = 0; i < n; ++i) {
11                  for (size_t j = 0; j < n; ++j) {
```

```
12                      if (MIO::inputStreamLL(mrx[i][j], iqp.in,
                           iqp.eout, iqp.printErr) != 0)
13                          throw MExc::ThereIsNothing();
14                  }
15              }
16              return mrx;
17          } catch (std::exception &err) {
18              iqp.eout << err.what() << std::endl;
19          }
20          return {0, 0};
21      }
22
23      void oXY(const Matrix &mrx, const std::string &exc,
           std::ostream &out) {
24          out << exc << ") ";
25          for (size_t i = 1, endi = mrx.cols(); i <= endi; ++i) {
26              bool f = false;
27              for (size_t j = 0, endj = mrx.cols(); j < endj; ++j) {
28                  if (mrx[0][j] == i) {
29                      out << j + 1 << " ";
30                      f = true;
31                  }
32              }
33              if (f) out << "\t";
34          }
35          out << std::endl;
36      }
37  }
38  void MDialog::ioXY(Dialog::Ioeqpp &iqp) {
39      Matrix mrx = iXY(iqp);
40      auto ans = Matrix(1, mrx.cols());
41      for (size_t i = 0, n = mrx.cols(); i < n; ++i) {
42          ans[0][i] = 1;
43          for (size_t j = 0; j < n; ++j) {
44              ans[0][i] = ans[0][i] && (mrx[i][j] == 0);
45          }
46      }
47      oXY(ans, "a", iqp.out);
48
49      ans = {1, mrx.cols()};
50      for (size_t i = 0, n = ans.cols(); i < n; ++i) {
51          ans[0][i] = 0;
52          for (size_t j = 0; j < n; ++j) {
53              bool f = true;
54              for (size_t k = 0; k < n; ++k) {
55                  f = f && (mrx[i][k] == mrx[j][k]);
56              }
```

```
57              if (f) ans[0][j] = i + 1;
58          }
59      }
60      oXY(ans, "b", iqp.out);
61
62      ans = {1, mrx.cols()};
63      for (size_t i = 0, n = ans.cols(); i < n; ++i) {
64          ans[0][i] = 1;
65          for (size_t j = 0; j < n; ++j) {
66              ans[0][i] = ans[0][i] && (mrx[i][j] % 2 == 0);
67          }
68      }
69      oXY(ans, "c", iqp.out);
70
71      ans = {1, mrx.cols()};
72      for (size_t i = 0, n = ans.cols(); i < n; ++i) {
73          ans[0][i] = 1;
74          for (size_t j = 1; j < n; ++j) {
75              ans[0][i] = ans[0][i] && ((mrx[i][0] < mrx[i][n - 1]
                    && mrx[i][j - 1] < mrx[i][j])
76                                        (mrx[i][0] > mrx[i][n - 1] &&
                                            mrx[i][j - 1] > mrx[i][j]));
77          }
78      }
79      oXY(ans, "d", iqp.out);
80
81      ans = {1, mrx.cols()};
82      for (size_t i = 0, n = ans.cols(); i < n; ++i) {
83          ans[0][i] = 1;
84          for (size_t j = 0; j < n; ++j) {
85              ans[0][i] = ans[0][i] && (mrx[i][j] == mrx[i][n - j -
                    1]);
86          }
87      }
88      oXY(ans, "e", iqp.out);
```

Листинг 22: Первая реализация, myfuncs.cpp

```
1  #else
2  namespace {
3      std::pair<int64_t *, size_t> iXY(Dialog::Ioeqpp &iqp) {
4          size_t n = 0;
5          int64_t *mrx = nullptr;
6
7          try {
8              if (iqp.printQst) iqp.qout << "n" << std::endl;
9              if (MIO::inputULL(n, iqp.in, iqp.eout, iqp.printErr)
                    != 0) throw MExc::ThereIsNothing();
```

```
10              mrx = new int64_t[n * n];
11              if (iqp.printQst) iqp.qout << "Matrix nxn:" <<
                    std::endl;
12              for (size_t i = 0; i < n * n; ++i)
13                  if (MIO::inputStreamLL(mrx[i], iqp.in, iqp.eout,
                        iqp.printErr) != 0)
14                      throw MExc::ThereIsNothing();
15          } catch (std::exception &err) {
16              iqp.eout << err.what() << std::endl;
17              delete[] mrx;
18              return {nullptr, 0};
19          }
20          return {mrx, n};
21      }
22
23      void oXY(const uint32_t *inp, size_t n, const std::string
            &exs, std::ostream &out) {
24          out << exs << ") ";
25          for (int i = 1; i <= n; ++i) {
26              bool f = false;
27              for (int j = 0; j < n; ++j) {
28                  if (inp[j] == i) {
29                      out << j + 1 << " ";
30                      f = true;
31                  }
32              }
33              if (f) out << "\t";
34          }
35          out << std::endl;
36      }
37  }
38
39  void MDialog::ioXY(Ioeqpp &iqp) {
40      auto [mrx, n] = iXY(iqp);
41      if (mrx == nullptr) return;
42      auto *ans = new uint32_t[n];
43      for (size_t i = 0; i < n; ++i) {
44          ans[i] = 1;
45          for (size_t j = 0; j < n; ++j) {
46              ans[i] = ans[i] && (mrx[i * n + j] == 0);
47          }
48      }
49      oXY(ans, n, "a", iqp.out);
50      delete[] ans;
51
52      ans = new uint32_t[n];
53      for (size_t i = 0; i < n; ++i) {
```

```
54          ans[i] = 0;
55          for (size_t j = 0; j < n; ++j) {
56              bool f = true;
57              for (size_t k = 0; k < n; ++k) {
58                  f = f && (mrx[i * n + k] == mrx[j * n + k]);
59              }
60              if (f) ans[j] = i + 1;
61          }
62      }
63      oXY(ans, n, "b", iqp.out);
64      delete[] ans;
65
66      ans = new uint32_t[n];
67      for (size_t i = 0; i < n; ++i) {
68          ans[i] = 1;
69          for (size_t j = 0; j < n; ++j) {
70              ans[i] = ans[i] && (mrx[i * n + j] % 2 == 0);
71          }
72      }
73      oXY(ans, n, "c", iqp.out);
74      delete[] ans;
75
76      ans = new uint32_t[n];
77      for (size_t i = 0; i < n; ++i) {
78          ans[i] = 1;
79          for (size_t j = 1; j < n; ++j) {
80              ans[i] = ans[i] && ((mrx[i * n] < mrx[(i + 1) * n - 1]
                    && mrx[i * n + j - 1] < mrx[i * n + j])
81                                  (mrx[i * n] > mrx[(i + 1) * n - 1]
                                        && mrx[i * n + j - 1] > mrx[i *
                                        n + j]));
82          }
83      }
84      oXY(ans, n, "d", iqp.out);
85      delete[] ans;
86
87      ans = new uint32_t[n];
88      for (size_t i = 0; i < n; ++i) {
89          ans[i] = 1;
90          for (size_t j = 0; j < n; ++j) {
91              ans[i] = ans[i] && (mrx[i * n + j] == mrx[(i + 1) * n
                    - 1 - j]);
92          }
93      }
94      oXY(ans, n, "e", iqp.out);
95      delete[] ans;
96      delete[] mrx;
```

Листинг 23: constants.h

```
#ifndef LAB3_CONSTANTS_H
#define LAB3_CONSTANTS_H

#include <cstdint>

constexpr size_t arrsize = 100500;

#endif//LAB3_CONSTANTS_H
```

## 5.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль

```
What would you use for input? 1 - Console, 0 - file:
1
What would you use for output? 1 - Console, 0 - file:
1
n
6
Matrix nxn:
1 2 3 3 2 1
2 4 6 6 4 2
1 2 3 4 5 6
6 5 4 3 2 1
0 0 0 0 0 0
1 2 3 3 2 1
a) 5
b) 2    3       4       5       1 6
c) 2 5
d) 3 4
e) 1 2 5 6
Repeat? 1 - Yes, 0 - No:
0
```

# 6 Лабораторная работа 4

## 6.1 Задание

Даны натуральное число $n$, символы $s_1, \ldots, s_n$. Преобразовать последовательность $s_1, \ldots, s_n$, удалив каждый символ * и повторив каждый символ, отличный от *.

## 6.2 Требования

1. Требования 1-3 из лабораторной работы 3.
2. работа со строками исходной длины.
3. Ввод исходных данных из файла.
4. Вывод результатов в файл (Путь к файлу задаётся с консоли, если он не указывается, то используется путь к файлу по умолчанию. То есть, например программа выдаёт такой запрос: «Укажите файл для вывода результатов работы программы [ ‘rez‘out.txt]:». Если просто нажать Enter, то файлом вывода будет файл « ‘rez‘out.txt»).

## 6.3 Листинги исходных файлов программы

Листинг 24: main.cpp

```
#include "myfuncs.h"
#include "version.h"

int main() {
    std::cout << "Currently you use the " << LABS_VERSION << "
       version of the Labs.\n" << "Git commit hash: " <<
       GIT_COMMIT_HASH << "." << std::endl;
    MDialog dialog{};
    bool DS = true;
    while (DS) {
        if (!dialog.step()) break;
        std::cout << "Repeat? 1 - Yes, 0 - No: " << std::endl;
        MIO::inputBool(DS, std::cin, std::cerr);
    }
    return 0;
}
```

Листинг 25: myfuncs.h

```
#ifndef LAB4_MYFUNCS_H
#define LAB4_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
```

```
7  #include "dialog.h"
8
9  struct MDialog : public Dialog {
10     void ioXY(Ioeqpp &iqp) override;
11 };
12
13 #endif//LAB4_MYFUNCS_H
```

Листинг 26: myfuncs.cpp

```
1  #include "myfuncs.h"
2  #include "MyString.h"
3
4  namespace {
5      MyString iXY(Dialog::Ioeqpp &iqp) {
6          if (iqp.printQst) iqp.qout << "Write your string, please"
               << std::endl;
7          MyString str;
8          iqp.in >> str;
9          return str;
10     }
11
12     void oXY(MyString &str, std::ostream &out) {
13         out << str.c_str() << std::endl;
14     }
15 }
16
17 void MDialog::ioXY(Ioeqpp &iqp) {
18     MyString x = iXY(iqp);
19     MyString y(x);
20     oXY(y, iqp.out);
21 }
```

Листинг 27: myfuncs.h

```
1  #ifndef LAB4_MYSTRING_H
2  #define LAB4_MYSTRING_H
3
4  #include <iostream>
5
6  class MyString {
7      size_t size_;
8      size_t capacity_;
9      char*  buffer_;
10
11     void strExt();
12     void reset();
13
14 public:
```

```
15      MyString();
16      MyString(MyString &x);
17      explicit MyString(const char *x);
18      [[nodiscard]] size_t size() const;
19      [[nodiscard]] const char* c_str() const;
20      void readStr(std::istream &in);
21      ~MyString();
22      void f(MyString &x);
23      MyString& operator=(const char *x);
24      char operator[](size_t i);
25      friend std::istream &operator>>(std::istream &in, MyString
           &string);
26      friend std::ostream &operator<<(std::ostream &os, const
           MyString &string);
27      friend MyString operator+(const MyString& a, const MyString&
           b);
28      const char *begin();
29      const char *end();
30  };
31
32
33  #endif//LAB4_MYSTRING_H
```

Листинг 28: myfuncs.cpp

```
1   #include "MyString.h"
2   #include <sstream>
3
4   MyString::MyString() : size_(0), capacity_(64) {
5       buffer_ = new char[capacity_];
6   }
7
8   size_t MyString::size() const {
9       return size_;
10  }
11
12  const char *MyString::c_str() const {
13      return buffer_;
14  }
15
16  void MyString::readStr(std::istream &in) {
17      char buffer[4096];
18      while (in.get(buffer, 4096)) {
19          size_t size = in.gcount();
20          if (size_ + size >= capacity_) strExt();
21          memcpy(buffer_ + size_, buffer, size);
22          size_ += size;
23          if (size != 4095  in.peek() == '\n') {
```

```
24              buffer_[size_] = '\0';
25              in.ignore();
26              break;
27          }
28      }
29  }
30
31  MyString::~MyString() {
32      delete[] buffer_;
33  }
34
35  std::ostream &operator<<(std::ostream &os, const MyString &string)
        {
36      os << "size_: " << string.size_ << " capacity_: " <<
            string.capacity_ << " buffer_: " << string.buffer_;
37      return os;
38  }
39
40  void MyString::f(MyString &x) {
41      reset();
42      const char *str = x.c_str();
43      for (size_t i = 0; i < x.size(); ++i) {
44          if (str[i] != '*') {
45              if (size_ + 2 >= capacity_) strExt();
46              buffer_[size_++] = str[i];
47              buffer_[size_++] = str[i];
48          }
49      }
50      buffer_[size_] = '\0';
51  }
52
53  void MyString::strExt() {
54      capacity_ *= 2;
55      char *tmp = new char[capacity_];
56      memcpy(tmp, buffer_, size_);
57      delete[] buffer_;
58      buffer_ = tmp;
59  }
60
61  MyString::MyString(MyString &x) : size_(0), capacity_(64),
        buffer_(nullptr)  {
62      this->f(x);
63  }
64
65  std::istream &operator>>(std::istream &in, MyString &string) {
66      string.readStr(in);
67      return in;
```

```
68 }
69
70 MyString operator+(const MyString& a, const MyString& b) {
71     MyString x;
72     x.size_ = a.size_ + b.size_;
73     x.capacity_ = std::max(a.capacity_, b.capacity_);
74     if (x.size_ >= x.capacity_) x.strExt();
75     memcpy(x.buffer_, a.buffer_, a.size_);
76     memcpy(x.buffer_ + a.size_, b.buffer_, b.size_);
77     x.buffer_[x.size_] = '\0';
78     return x;
79 }
80
81 MyString::MyString(const char *x) : size_(0), capacity_(64),
      buffer_(nullptr) {
82     (*this) = x;
83 }
84
85
86 MyString& MyString::operator=(const char *x) {
87     reset();
88     std::stringstream ss;
89     ss << x;
90     ss >> (*this);
91     return *this;
92 }
93
94 void MyString::reset() {
95     size_ = 0;
96     capacity_ = 64;
97     delete []buffer_;
98     buffer_ = new char[capacity_];
99 }
100
101 const char *MyString::begin() {
102     return buffer_;
103 }
104
105 const char *MyString::end() {
106     return buffer_ + size_;
107 }
108
109 char MyString::operator[](size_t i) {
110     if (i >= size_) throw std::out_of_range("MyStr");
111     return buffer_[i];
112 }
```

## 6.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль

```
What would you use for input? 1 - Console, 0 - file:
1
What would you use for output? 1 - Console, 0 - file:
1
Write your string, please
Hello ** world *t t
HHeellllloo    wwoorrlldd  tt  tt
Repeat? 1 - Yes, 0 - No:
0
```

2. Консоль ⟶ консоль

in.txt

```
123456
101
Helllo *** world!
```

консоль

```
What would you use for input? 1 - Console, 0 - file:
0
What would you use for output? 1 - Console, 0 - file:
1
Write name of file (in.txt as default):

112233445566
110011
HHeellllllloo    wwoorrlldd!!
```

# 7 Лабораторная работа 5

## 7.1 Задание

Определение наличия в числе указанной битовой последовательности.

## 7.2 Требования

1. Дружественный интерфейс (удобство ввода данных, наглядность получаемых результатов).
2. Ввод исходных данных через параметры командной строки.
3. Реализацию задания в виде отдельной функции.
4. Корректность работы реализованной функции с целыми числами различных размерностей (от 1-го до 8 байт, при необходимости уточнять нюансы у преподавателя).

## 7.3 Листинги исходных файлов программы

Листинг 29: main.cpp

```
1  #include <iostream>
2  #include <sstream>
3  #include "LAB4/myfuncs.h"
4  #include "dialog.h"
5  #include "MIO.h"
6  #include "version.h"
7
8
9  int main(int argc, char **argv) {
10     std::cout << "Currently you use the " << LABS_VERSION << "
          version of the Labs.\n" << "Git commit hash: " <<
          GIT_COMMIT_HASH << "." << std::endl;
11     MDialog dialog{};
12     try {
13         if (argc < 3) throw MExc::ThereIsNothing();
14         std::stringstream sst;
15         for (int i = 1; i < argc; ++i) sst << argv[i] << "\n";
16         Dialog::Ioeqpp iqp{sst, std::cout, std::cerr, std::cout,
              false, false};
17         dialog.ioXY(iqp);
18     } catch (std::exception &err) {
19         std::cerr << err.what() << "\n" << "Give me the number and
              some sequence, please" << std::endl;
20     }
21     return 0;
22 }
```

Листинг 30: myfuncs.h

```h
#ifndef LAB5_MYFUNCS_H
#define LAB5_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
#include "dialog.h"

struct MDialog : public Dialog {
    void ioXY(Ioeqpp &iqp) override;
};

#endif//LAB5_MYFUNCS_H
```

Листинг 31: myfuncs.cpp

```cpp
#include "myfuncs.h"
#include "MIO.h"

namespace {
std::pair<uint64_t, uint64_t> iXY(Dialog::Ioeqpp &iqp) {
        if (iqp.printQst) iqp.qout << "Unsigned integer: " <<
           std::endl;
        uint64_t n;
        MIO::inputULL(n, iqp.in, iqp.eout, iqp.printErr);
        std::bitset<sizeof(n)*8> bits;
        if (iqp.printQst) iqp.qout << "Binary subsequence: " <<
           std::endl;
        MIO::inputBS<sizeof(n)*8>(bits, iqp.in, iqp.eout,
           iqp.printErr);
        return {n, bits.to_ullong()};
    }
}


void MDialog::ioXY(Ioeqpp &iqp) {
    auto [x, y] = iXY(iqp);
    std::bitset<sizeof(x)*8> bb(x);
    iqp.out << bb << "\n";
    bb = y;
    iqp.out << bb << std::endl;
    size_t msb, sz = sizeof(x)*8;
    for (msb = 0; (y >> msb) != 0; ++msb);

    bool f = false;
    for (size_t i = 0; i < sz - msb; ++i) {
        f = f  (((x << (sz - msb - i)) >> (sz - msb)) == y);
```

```
29      }
30      iqp.out << (f ? "There is the subsequence" : "There isn't the
            subsequence") << std::endl;
31  }
```

## 7.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль (здесь есть сокращения)

```
1  >  .\Labs.exe 7 10
2  000...0000111
3  000...0000010
4  There isn't the subsequence
5  >  .\Labs.exe 7 11
6  000...0000111
7  000...0000011
8  There is the subsequence
```

## 7.5 Неккорктный ввод

1. Недостаточное количество аргументов:

консоль

```
1  >  .\Labs.exe
2  There is nothing
3  Give me the number and some sequence, please
4  >  .\Labs.exe 1
5  There is nothing
6  Give me the number and some sequence, please
```

# 8 Лабораторная работа 6

## 8.1 Задание

В лабораторной работе нужно реализовать в программе работу с некоторой структурой представления данных (однонаправленный список, содержит в себе {ФИО; Должность; Место проживания}), то есть обеспечить доступ к её данным в рамках заданного перечня функций:

1. Подсчитать: сколько имеется элементов с заданным содержимым одного из полей;
2. Печать всех элементов (вывод на консоль);
3. Сброс значений всех элементов (например обнуление);
4. Чтение из файла, запись в файл.

## 8.2 Требования

1. Дружественный интерфейс (удобство ввода данных, наглядность получаемых результатов);
2. Возможность проверки (использования) всех функций реализованных в программе без её перезапуска, то есть должно быть реализовано некоторое «консольное» меню (выводится список вариантов действия и предлагается сделать выбор);
3. Для сохранения списка в программе предусмотреть текстовый файл (для чтения из него и записи в него). Файл должен содержать все элементы списка со всеми их полями, представленные в табличном виде;

## 8.3 Листинги исходных файлов программы

Листинг 32: main.cpp

```
1  #include "myfuncs.h"
2  #include "version.h"
3
4  int main() {
5      std::cout << "Currently you use the " << LABS_VERSION << "
           version of the Labs.\n" << "Git commit hash: " <<
           GIT_COMMIT_HASH << "." << std::endl;
6      MDialog dialog{};
7      bool DS = true;
8      while (DS) {
9          if (!dialog.step()) break;
10         std::cout << "Repeat? 1 - Yes, 0 - No: " << std::endl;
11         MIO::inputBool(DS, std::cin, std::cerr);
12     }
13     return 0;
14 }
```

Листинг 33: myfuncs.h

```
#ifndef LAB6_MYFUNCS_H
#define LAB6_MYFUNCS_H

#include <iostream>
#include <cmath>
#include <fstream>
#include "dialog.h"

struct MDialog : public Dialog {
    void ioXY(Ioeqpp &iqp) override;
};

#endif//LAB6_MYFUNCS_H
```

Листинг 34: myfuncs.cpp

```
#include "myfuncs.h"
#include "MLL.h"

namespace {
    void cntD(MLL *mll, Dialog::Ioeqpp &iqp) {
        if (iqp.printQst) iqp.qout << "Write string, which ends by \'$\' sign:" << std::endl;
        std::string str;
        getline(iqp.in, str, '$');
        iqp.in.ignore();
        std::vector<Dialog::FuncWithDesc<uint64_t(MLL *, std::string &)>> options = {
                {"same full name", [](MLL *mll, std::string &str) { return mll->cnt(str, 0); }},
                {"same job",       [](MLL *mll, std::string &str) { return mll->cnt(str, 1); }},
                {"same place",     [](MLL *mll, std::string &str) { return mll->cnt(str, 2); }},
        };
        iqp.out << options[Dialog::optionD<uint64_t(MLL *, std::string &)>(options, iqp)].func(mll, str) << std::endl;
    }
}

void MDialog::ioXY(Ioeqpp &iqp) {
    auto *mll = new MLL;
    std::vector<Dialog::FuncWithDesc<bool(MLL *, Dialog::Ioeqpp &)>> options = {
            {"print elements",     [](MLL *mll, Dialog::Ioeqpp &iqp) {
```

```
23                  mll->print(iqp.out);
24                  iqp.qout << std::endl;
25                  return true;
26              }},
27              {"print table",         [](MLL *mll, Dialog::Ioeqpp
                   &iqp) {
28                  mll->printT(iqp.out);
29                  return true;
30              }},
31              {"count coincidences", [](MLL *mll, Dialog::Ioeqpp
                   &iqp) {
32                  cntD(mll, iqp);
33                  return true;
34              }},
35              {"reset",               [](MLL *mll, Dialog::Ioeqpp
                   &iqp) {
36                  mll->clean();
37                  return true;
38              }},
39              {"read file",           [](MLL *mll, Dialog::Ioeqpp
                   &iqp) {
40                  auto fin = Dialog::fileD<std::ifstream>(iqp);
41                  while (!fin.eof()) fin >> *mll;
42                  fin.close();
43                  return true;
44              }},
45              {"write to file",       [](MLL *mll, Dialog::Ioeqpp
                   &iqp) {
46                  auto fout = Dialog::fileD<std::ofstream>(iqp);
47                  fout << *mll;
48                  fout.close();
49                  return true;
50              }},
51              {"exit",                [](MLL *mll, Dialog::Ioeqpp
                   &iqp) { return false; }}
52      };
53
54      bool f = true;
55      while (f) f = options[Dialog::optionD<bool(MLL *,
            Dialog::Ioeqpp &)>(options, iqp)].func(mll, iqp);
56
57      delete mll;
58  }
```

Листинг 35: myfuncs.h

```
1  #ifndef LAB6_MLL_H
2  #define LAB6_MLL_H
```

```
3
4  #include <iostream>
5  #include <string>
6
7  struct Node {
8      Node *next = nullptr;
9      std::string NSF_, job_, place_;
10     friend std::istream &operator>>(std::istream &in, Node &n);
11     friend bool operator==(const Node &a, const Node &b);
12     friend std::ostream &operator<<(std::ostream &out, Node &n);
13 };
14
15 class MLL {
16     Node *first_ = nullptr, *last_ = nullptr;
17
18 public:
19     bool empty();
20     void r_f();
21     void clean();
22     void print(std::ostream &out = std::cout, std::ostream &eout =
           std::cerr, std::ostream &qout = std::cout);
23     void printT(std::ostream &out = std::cout, std::ostream &eout
           = std::cerr, std::ostream &qout = std::cout);
24     void push_back(Node *node);
25     uint64_t cnt(std::string &s, uint64_t type);
26     ~MLL();
27
28     friend std::istream &operator>>(std::istream &in, MLL &mll);
29     friend std::ostream &operator<<(std::ostream &out, MLL &mll);
30 };
31
32
33 #endif//LAB6_MLL_H
```

Листинг 36: myfuncs.cpp

```
1  #include "MLL.h"
2  #include "MExc.h"
3  #include <iomanip>
4
5  std::istream &operator>>(std::istream &in, Node &n) {
6      try {
7          std::getline(in, n.NSF_, '$');
8          std::getline(in, n.job_, '$');
9          std::getline(in, n.place_, '$');
10     }
11     catch (std::exception &err) {}
12     return in;
```

```
13 }
14
15 bool operator==(const Node &a, const Node &b) {
16     return a.NSF_ == b.NSF_ && a.job_ == b.job_ && a.place_ ==
       b.place_;
17 }
18
19 std::ostream &operator<<(std::ostream &out, Node &n) {
20     out << "{"
21         << "\"Full name\": " << "\"" << n.NSF_ << "\"" << ", "
22         << "\"Job\": " << "\"" << n.job_ << "\"" << ", "
23         << "\"Place\": " << "\"" << n.place_ << "\"" << "}";
24     return out;
25 }
26
27 bool MLL::empty() {
28     return first_ == nullptr;
29 }
30
31 void MLL::r_f() {
32     if (empty()) return;
33     Node *p = first_;
34     first_ = p->next;
35     delete p;
36 }
37
38 void MLL::clean() {
39     while (!empty()) r_f();
40 }
41
42 void MLL::print(std::ostream &out, std::ostream &eout,
     std::ostream &qout) {
43     for (Node *p = first_; p != nullptr; p = p->next)
44         out << *p << ", ";
45 }
46
47 void MLL::printT(std::ostream &out, std::ostream &eout,
     std::ostream &qout) {
48     out << std::setw(32) << std::left << "Full name"
49         << std::setw(32) << std::left << "Job"
50         << std::setw(32) << std::left << "Place" << '\n';
51     for (Node *p = first_; p != nullptr; p = p->next)
52         out << std::left << std::setw(32) << p->NSF_
53             << std::left << std::setw(32) << p->job_
54             << std::left << std::setw(32) << p->place_ << '\n';
55     out << std::endl;
56 }
```

```
57
58 void MLL::push_back(Node *node) {
59     if (empty()) {
60         first_ = node;
61         last_ = node;
62         return;
63     }
64     last_->next = node;
65     last_ = node;
66 }
67
68 uint64_t MLL::cnt(std::string &s, uint64_t type) {
69     uint64_t res = 0;
70     if (type > 2) throw MExc::NotThatType();
71     for (Node *p = first_; p != nullptr; p = p->next) {
72         std::string *ss[] = {&p->NSF_, &p->job_, &p->place_};
73         if (s == *ss[type]) ++res;
74     }
75     return res;
76 }
77
78 MLL::~MLL() {
79     clean();
80 }
81
82 std::istream &operator>>(std::istream &in, MLL &mll) {
83     Node *n = new Node;
84     in >> *n;
85     mll.push_back(n);
86     return in;
87 }
88
89 std::ostream &operator<<(std::ostream &out, MLL &mll) {
90     for (const Node *p = mll.first_; p != nullptr; p = p->next)
91         out << p->NSF_ << "$" << p->job_ << "$" << p->place_ << (p
              != mll.last_ ? "$" : "");
92     return out;
93 }
```

## 8.4 Примеры выполнения программы

1. Консоль ⟶ консоль

консоль

```
1 What would you use for input? 1 - Console, 0 - file:
2 1
3 What would you use for output? 1 - Console, 0 - file:
```

```
4  1
5  Choose an option:
6  1 - print elements;
7  2 - print table;
8  3 - count coincidences;
9  4 - reset;
10 5 - read file;
11 6 - write to file;
12 7 - exit;
13 5
14 Write name of file (in.txt as default):
15
16 Choose an option:
17 1 - print elements;
18 2 - print table;
19 3 - count coincidences;
20 4 - reset;
21 5 - read file;
22 6 - write to file;
23 7 - exit;
24 1
25 {"Full name": "NDD", "Job": "Student", "Place": "Kazan"},
      {"Full name": "TPI", "Job": "Lecturer", "Place": "Kazan"},
      {"Full name": "GSI", "Job": "Tutor", "Place": "Kazan"},
26 Choose an option:
27 1 - print elements;
28 2 - print table;
29 3 - count coincidences;
30 4 - reset;
31 5 - read file;
32 6 - write to file;
33 7 - exit;
34 2
35 Full name                      Job
      Place
36 NDD                              Student
                               Kazan
37 TPI                              Lecturer
                              Kazan
38 GSI                              Tutor
                                 Kazan
39
40 Choose an option:
41 1 - print elements;
42 2 - print table;
43 3 - count coincidences;
44 4 - reset;
```

```
5 - read file;
6 - write to file;
7 - exit;
7
Repeat? 1 - Yes, 0 - No:
0
```

in.txt

```
NDD$Student$Kazan$TPI$Lecturer$Kazan$GSI
   $Tutor$Kazan$NDD$Student$Kazan$TPI$
   Lecturer$Kazan$GSI$Tutor$Kazan
```

2. Файл ⟶ файл

in1.txt

```
5
in.txt
1
7
```

консоль

```
What would you use for input? 1 - Console, 0 - file:
0
What would you use for output? 1 - Console, 0 - file:
0
Write name of file (out.txt as default):

Write name of file (in.txt as default):
in1.txt
```

in.txt

```
NDD$Student$Kazan$TPI$Lecturer$Kazan$GSI
   $Tutor$Kazan$NDD$Student$Kazan$TPI$
   Lecturer$Kazan$GSI$Tutor$Kazan
```

out.txt

```
{"Full name": "NDD", "Job": "Student", "Place": "Kazan"},
   {"Full name": "TPI", "Job": "Lecturer", "Place": "Kazan"},
   {"Full name": "GSI", "Job": "Tutor", "Place": "Kazan"},
   {"Full name": "NDD", "Job": "Student", "Place": "Kazan"},
   {"Full name": "TPI", "Job": "Lecturer", "Place": "Kazan"},
   {"Full name": "GSI", "Job": "Tutor", "Place": "Kazan"},
```