МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ОТЧЁТ ЛАБОРАТОРНОЙ РАБОТЫ №1

ПО ДИСЦИПЛИНЕ «ОПЕРАЦИОННЫЕ СИСТЕМЫ И СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ»
ЗА II СЕМЕСТР

Выполнил:
студент 3-го курса
V-го семестр
факультета ЭИС
группы ПО-4(2)
зачётная книжка №190333
Галанин П. И.
«__» _____ 2021 г.

Проверил:
ассистент
кафедры ИИТ
Дряпко А. В.
«__» _____ 2021 г.

Брест 2021

Цель работы: приобрести практические навыки проектирования и разработки приложений с графическим пользовательским интерфейсом в ОС Windows средствами Qt.

Вариант 5

Игра «Сокобан». Один уровень игры. Общая идея: имеется комната-лабиринт (15X15 ячеек), в которой необходимо расставить ящики (5 штук) на указанные позиции. Главный герой может лишь толкать ящики вперед. Таким образом, возможны конфигурации, из которых не возможно построить желаемое решение (например, если ящик был задвинут в тупик).

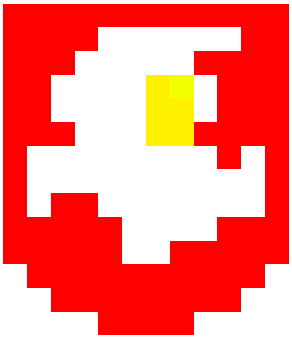

Рисунок 1 – Иконка игры

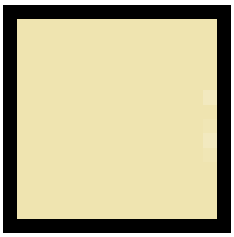

Рисунок 2 – Пустая ячейка

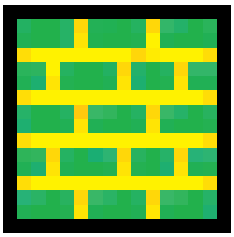

Рисунок 3 – Ячейка с коробкой

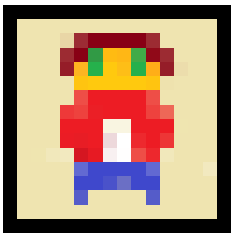

Рисунок 4 – Ячейка с игроком

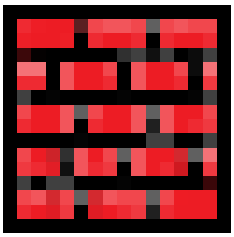

Рисунок 5 – Ячейка со стеной

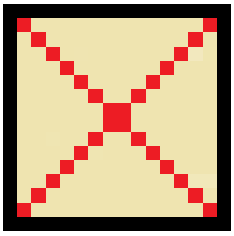

Рисунок 6 – Финиш для коробки

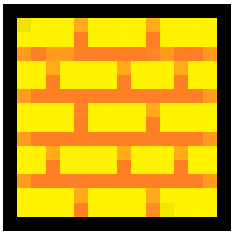

Рисунок 7 – Коробка на финише

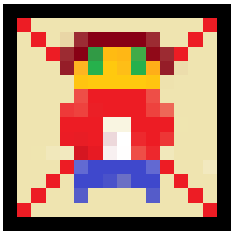

Рисунок 8 – Игрок на финише

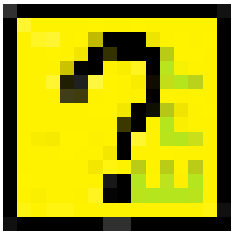

Рисунок 9 – Ячейка без функционала

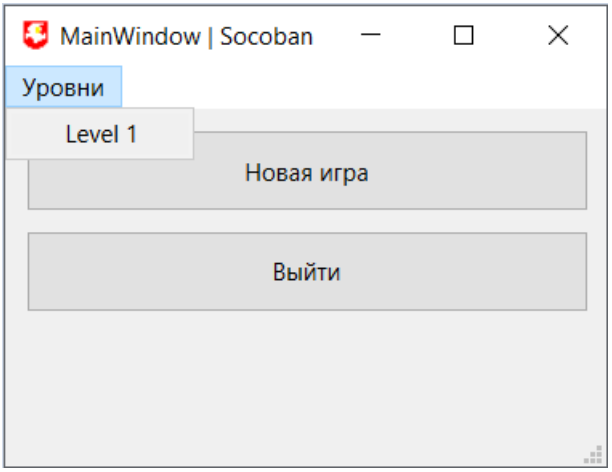

Рисунок 10 – Main Window

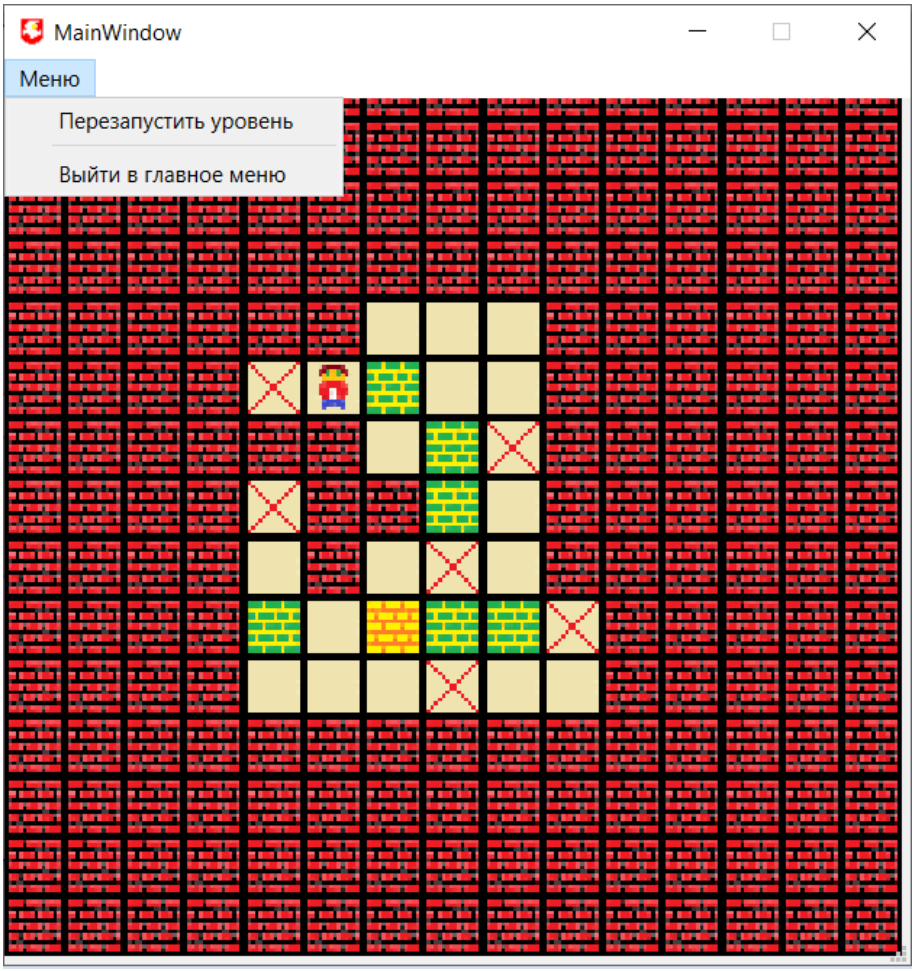

Рисунок 11 – Game Window

../gpi_osisp5_option5/gpi_osisp5_option5.pro

```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++11

# You can make your code fail to compile if it uses deprecated APIs.
# In order to do so, uncomment the following line.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs deprecated before Qt 6.0.0

SOURCES += \
    gpi_gamewindow.cpp \
    gpi_gamewindow__gpi_drawAxes.cpp \
    gpi_gamewindow__gpi_drawTextures.cpp \
    gpi_gamewindow__gpi_generateLevel1.cpp \
    gpi_gamewindow__gpi_goBottom.cpp \
    gpi_gamewindow__gpi_goLeft.cpp \
    gpi_gamewindow__gpi_goRight.cpp \
    gpi_gamewindow__gpi_goTop.cpp \
    gpi_gamewindow__gpi_sayWon.cpp \
    main.cpp \
    gpi_mainwindow.cpp

HEADERS += \
    gpi_gamewindow.hpp \
    gpi_mainwindow.hpp

FORMS += \
    gpi_gamewindow.ui \
    gpi_mainwindow.ui

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target

RESOURCES += \
    gpi_gamewindow.qrc
```

../gpi_osisp5_option5/main.cpp

```
#include "gpi_mainwindow.hpp"

#include <QApplication>

int main (int argc, char *argv[])
{
    QApplication a (argc, argv);
    gpi_MainWindow w;
    w.show ();
    return a.exec ();
}
```

../gpi_osisp5_option5/gpi_mainwindow.hpp

```
#ifndef GPI_MAINWINDOW_HPP
#define GPI_MAINWINDOW_HPP
```

```
3
4  #include <QMainWindow>
5  #include "gpi_gamewindow.hpp"
6
7  QT_BEGIN_NAMESPACE
8  namespace Ui { class gpi_MainWindow; }
9  QT_END_NAMESPACE
10
11 class gpi_MainWindow : public QMainWindow
12 {
13     Q_OBJECT
14
15 public:
16     gpi_MainWindow(QWidget *parent = nullptr);
17     ~gpi_MainWindow();
18
19 private slots:
20     void gpi_on_actionLevel1_triggered();
21     void on_pushButton_ShowGameWindow_clicked();
22     void on_pushButton_CloseMainWindow_clicked();
23
24 private:
25     Ui::gpi_MainWindow *ui;
26 };
27 #endif // GPI_MAINWINDOW_HPP
```

../gpi_osisp5_option5/gpi_mainwindow.cpp

```
1  #include "gpi_mainwindow.hpp"
2  #include "ui_gpi_mainwindow.h"
3
4  gpi_MainWindow::gpi_MainWindow(QWidget *parent)
5      : QMainWindow(parent)
6      , ui(new Ui::gpi_MainWindow)
7  {
8      ui->setupUi(this);
9      this->setWindowIcon(QIcon(":/@gpi@/_assets/gpi_MainWindow__favicon.png"));
10
11     connect(ui->actionLevel1, SIGNAL(triggered()), this, SLOT(gpi_on_actionLevel1_triggered()));
12 }
13
14 gpi_MainWindow::~gpi_MainWindow()
15 {
16     delete ui;
17 }
18
19 void gpi_MainWindow::gpi_on_actionLevel1_triggered()
20 {
21     gpi_GameWindow* gpi_gw = new gpi_GameWindow();
22     gpi_gw->show();
23     this->close();
24 }
25
26 void gpi_MainWindow::on_pushButton_ShowGameWindow_clicked()
27 {
28     gpi_GameWindow* gpi_gw = new gpi_GameWindow();
29     gpi_gw->show();
30     this->close();
31 }
32
```

```
33
34 void gpi_MainWindow::on_pushButton_CloseMainWindow_clicked ()
35 {
36     this->close ();
37 }
```

../gpi_osisp5_option5/gpi_gamewindow.hpp

```
1  #ifndef GPI_GAMEWINDOW_HPP
2  #define GPI_GAMEWINDOW_HPP
3  #define LENGTH 15
4
5  #include <QMainWindow>
6  #include <QPainter>
7  #include <QMessageBox>
8  #include <QKeyEvent>
9  #include <QTimer>
10 #include <QIcon>
11 #include "gpi_MainWindow.hpp"
12
13 namespace Ui {
14 class gpi_GameWindow;
15 }
16
17 class gpi_GameWindow : public QMainWindow
18 {
19     Q_OBJECT
20
21 public:
22     explicit gpi_GameWindow (QMainWindow *parent = nullptr);
23     ~gpi_GameWindow ();
24
25 public slots:
26     void gpi_gamewindow_animate();
27     void gpi_on_actionRestartLevel_triggered();
28     void gpi_on_actionMainMenu_triggered();
29
30 protected:
31     void paintEvent (QPaintEvent *event) override;
32     void keyPressEvent (QKeyEvent *e) override;
33
34 private:
35     Ui::gpi_GameWindow *ui;
36
37     const int    gpi_length              = LENGTH;
38     const int    gpi_WinWidth            = 500;
39     const int    gpi_WinHeight           = 500;
40     int          gpi_xPlayer;
41     int          gpi_yPlayer;
42     // = = = = = = = = = = = = = = = = =
43
44     enum gpi_MapChar {
45         err,
46         wall,
47         floor,
48         player,
49         finPlayer,
50         box,
51         finBox,
52         finish,
```

```
53      } gpi_map[LENGTH][LENGTH] = {{err}};
54
55      void gpi_drawAxes (QPainter* painter);
56      void gpi_drawTextures (QPainter *gpi_painter);
57      void gpi_generateLevel1 ();
58      void gpi_goBottom ();
59      void gpi_goLeft ();
60      void gpi_goRight ();
61      void gpi_goTop ();
62      void gpi_sayWon ();
63  };
64
65  #endif // GPI_GAMEWINDOW_HPP
```

../gpi_osisp5_option5/gpi_gamewindow.cpp

```
1   #include "gpi_gamewindow.hpp"
2   #include "ui_gpi_gamewindow.h"
3
4   gpi_GameWindow::gpi_GameWindow (QMainWindow *parent) :
5       QMainWindow (parent),
6       ui (new Ui::gpi_GameWindow)
7   {
8       QTimer*          timer       = new QTimer (this);
9       // = = = = = = = = = = = = = = = = = =
10
11      ui->setupUi (this);
12      this->setWindowIcon (QIcon (":/@gpi@/_assets/gpi_MainWindow__favicon.png"));
13
14      connect(ui->actionRestartLevel, SIGNAL(triggered ()), this,
        SLOT(gpi_on_actionRestartLevel_triggered ()));
15      connect(ui->actionMainMenu, SIGNAL(triggered ()), this, SLOT(gpi_on_actionMainMenu_triggered
        ()));
16
17      connect (timer, SIGNAL (timeout ()), this, SLOT (gpi_gamewindow_animate ()));
18      timer->start (100);
19
20      this->setFixedSize (this->gpi_WinWidth, this->gpi_WinHeight);
21      this->gpi_generateLevel1 ();
22  }
23
24  gpi_GameWindow::~gpi_GameWindow ()
25  {
26      delete ui;
27  }
28
29  void gpi_GameWindow::paintEvent (QPaintEvent *event)
30  {
31      QPainter         gpi_painter (this);
32      // = = = = = = = = = = = = = = = = = =
33
34      Q_UNUSED (event);
35      this->gpi_sayWon ();
36      this->gpi_drawAxes (&gpi_painter);
37      this->gpi_drawTextures (&gpi_painter);
38  }
39
40  void gpi_GameWindow::gpi_gamewindow_animate ()
41  {
42      repaint();
```

```
}

void gpi_GameWindow::gpi_on_actionRestartLevel_triggered ()
{
    this->gpi_generateLevel1 ();
}

void gpi_GameWindow::gpi_on_actionMainMenu_triggered ()
{
    gpi_MainWindow* gpi_mw = new gpi_MainWindow ();
    gpi_mw->show ();
    this->close ();
}

void gpi_GameWindow::keyPressEvent (QKeyEvent *e)
{
    switch(e->key ())
    {
        case Qt::Key_Up:
        case Qt::Key_W:
            this->gpi_goTop ();
            break;
        case Qt::Key_Right:
        case Qt::Key_D:
            this->gpi_goRight ();
            break;
        case Qt::Key_Down:
        case Qt::Key_S:
            this->gpi_goBottom ();
            break;
        case Qt::Key_Left:
        case Qt::Key_A:
            this->gpi_goLeft ();
            break;
    }
}
```

../gpi_osisp5_option5/gpi_gamewindow.qrc

```
<RCC>
    <qresource prefix="@gpi@">
        <file>_assets/gpi_MainWindow__favicon.png</file>
        <file>_assets/gpi_GameWindow__err.png</file>
        <file>_assets/gpi_GameWindow__wall.png</file>
        <file>_assets/gpi_GameWindow__floor.png</file>
        <file>_assets/gpi_GameWindow__player.png</file>
        <file>_assets/gpi_GameWindow__finPlayer.png</file>
        <file>_assets/gpi_GameWindow__box.png</file>
        <file>_assets/gpi_GameWindow__finBox.png</file>
        <file>_assets/gpi_GameWindow__finish.png</file>
    </qresource>
</RCC>
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_drawAxes.cpp

```
#include "gpi_gamewindow.hpp"
#include "ui_gpi_gamewindow.h"

void gpi_GameWindow::gpi_drawAxes(QPainter *gpi_painter)
{
```

```
    int          gpi_x   = this->gpi_WinWidth / this->gpi_length;
    int          gpi_y   = this->gpi_WinHeight / this->gpi_length;
    const int    gpi_l   = this->gpi_length;
    int          gpi_i;
    int          gpi_j;
    // = = = = = = = = = = = = = = = = =

    gpi_i = 0;
    while(gpi_i < gpi_l)
    {
        gpi_j = 0;
        while(gpi_j < gpi_l)
        {
            QRect gpi_rect(gpi_x * gpi_i, gpi_y * gpi_j, gpi_x, gpi_y);
            gpi_painter->drawRect(gpi_rect);
            ++gpi_j;
        }
        ++gpi_i;
    }
}
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_drawTextures.cpp

```
#include "gpi_gamewindow.hpp"
#include "ui_gpi_gamewindow.h"

void gpi_GameWindow::gpi_drawTextures(QPainter *gpi_painter)
{
    int             gpi_x       = this->gpi_WinWidth / this->gpi_length;
    int             gpi_y       = this->gpi_WinHeight / this->gpi_length;
    const int       gpi_l       = this->gpi_length;
    QString         gpi_path    = ":/img/_pics/err.png";
    int             gpi_i;
    int             gpi_j;
    // = = = = = = = = = = = = = = = = =

    gpi_i = 0;
    while(gpi_i < gpi_l)
    {
        gpi_j = 0;
        while(gpi_j < gpi_l)
        {
            switch(this->gpi_map[gpi_i][gpi_j])
            {
                case finish:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__finish.png";
                    break;
                case floor:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__floor.png";
                    break;
                case wall:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__wall.png";
                    break;
                case player:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__player.png";
                    break;
                case finPlayer:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__finPlayer.png";
                    break;
                case box:
```

```
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__box.png";
                    break;
                case finBox:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__finBox.png";
                    break;
                default:
                    gpi_path = ":/@gpi@/_assets/gpi_GameWindow__err.png";
                    break;
            }

            QPixmap gpi_pixmap(gpi_path);
            gpi_painter->drawPixmap(gpi_x * gpi_i, gpi_y * gpi_j, gpi_x, gpi_y, gpi_pixmap);
            ++gpi_j;
        }
        ++gpi_i;
    }
}
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_generateLevel1.cpp

```
#include "gpi_gamewindow.hpp"
#include "ui_gpi_gamewindow.h"

void gpi_GameWindow::gpi_generateLevel1 ()
{
    int         gpi_iter;
    int         gpi_i;
    int         gpi_j;
    int         gpi_l           = this->gpi_length;
    char        gpi_strMap[]    = "\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxx...xxxxxx\n\
xxxxf@b..xxxxxx\n\
xxxxxx.bfxxxxxx\n\
xxxxfxxb.xxxxxx\n\
xxxx.x.f.xxxxxx\n\
xxxxb.Bbbfxxxxx\n\
xxxx...f..xxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
xxxxxxxxxxxxxxx\n\
";
    int         gpi_strMapLen   = strlen(gpi_strMap);
    // = = = = = = = = = = = = = = = =

    gpi_iter = 0;
    gpi_i = 0;
    gpi_j = 0;
    while (gpi_iter < gpi_strMapLen)
    {
        if (gpi_strMap[gpi_iter] == '\n')
        {
            ++gpi_j;
            gpi_i = 0;
        }
        else if (gpi_i > gpi_l)
```

```
        {
            continue;
        }
        else
        {
            switch(gpi_strMap[gpi_iter])
            {
                case '.':
                    this->gpi_map[gpi_i][gpi_j] = floor;
                    break;
                case 'x':
                    this->gpi_map[gpi_i][gpi_j] = wall;
                    break;
                case '@':
                    this->gpi_map[gpi_i][gpi_j] = player;
                    this->gpi_xPlayer = gpi_i;
                    this->gpi_yPlayer = gpi_j;
                    break;
                case 'f':
                    this->gpi_map[gpi_i][gpi_j] = finish;
                    break;
                case 'A':
                    this->gpi_map[gpi_i][gpi_j] = finPlayer;
                    this->gpi_xPlayer = gpi_i;
                    this->gpi_yPlayer = gpi_j;
                    break;
                case 'b':
                    this->gpi_map[gpi_i][gpi_j] = box;
                    break;
                case 'B':
                    this->gpi_map[gpi_i][gpi_j] = finBox;
                    break;
                default:
                    this->gpi_map[gpi_i][gpi_j] = err;
                    break;
            }
            ++gpi_i;
        }
        ++gpi_iter;
    }
}
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_goBottom.cpp

```
#include "gpi_gamewindow.hpp"
#include "ui_gpi_gamewindow.h"

void gpi_GameWindow::gpi_goBottom ()
{
    int          gpi_l          = this->gpi_length;
    int          gpi_px         = this->gpi_xPlayer;
    int          gpi_py         = this->gpi_yPlayer;
    gpi_MapChar gpi_c0;
    gpi_MapChar gpi_c1;
    gpi_MapChar gpi_c2;
    // = = = = = = = = = = = = = = = = =

    if (gpi_py != gpi_l - 1)
    {
        gpi_c0 = this->gpi_map[gpi_px][gpi_py];
```

```
17          gpi_c1 = this->gpi_map[gpi_px][gpi_py + 1];
18          gpi_c2 = this->gpi_map[gpi_px][gpi_py + 2];
19
20          // player -> floor
21          // floor  -> player
22          if (gpi_c0 == player && gpi_c1 == floor)
23          {
24              gpi_py += 1;
25              this->gpi_yPlayer = gpi_py;
26              this->gpi_map[gpi_px][gpi_py - 1] = floor;
27              this->gpi_map[gpi_px][gpi_py    ] = player;
28              return;
29          }
30
31          // finPlayer -> floor
32          // finish    -> player
33          if (gpi_c0 == finPlayer && gpi_c1 == floor)
34          {
35              gpi_py += 1;
36              this->gpi_yPlayer = gpi_py;
37              this->gpi_map[gpi_px][gpi_py - 1] = finish;
38              this->gpi_map[gpi_px][gpi_py    ] = player;
39              return;
40          }
41
42          // player -> box    -> floor
43          // floor  -> player -> box
44          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == floor)
45          {
46              gpi_py += 1;
47              this->gpi_yPlayer = gpi_py;
48              this->gpi_map[gpi_px][gpi_py - 1] = floor;
49              this->gpi_map[gpi_px][gpi_py    ] = player;
50              this->gpi_map[gpi_px][gpi_py + 1] = box;
51              return;
52          }
53
54          // finPlayer -> box    -> floor
55          // finish    -> player -> box
56          if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == floor)
57          {
58              gpi_py += 1;
59              this->gpi_yPlayer = gpi_py;
60              this->gpi_map[gpi_px][gpi_py - 1] = finish;
61              this->gpi_map[gpi_px][gpi_py    ] = player;
62              this->gpi_map[gpi_px][gpi_py + 1] = box;
63              return;
64          }
65
66          // player -> finBox    -> floor
67          // floor  -> finPlayer -> box
68          if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == floor)
69          {
70              gpi_py += 1;
71              this->gpi_yPlayer = gpi_py;
72              this->gpi_map[gpi_px][gpi_py - 1] = floor;
73              this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
74              this->gpi_map[gpi_px][gpi_py + 1] = box;
75              return;
76          }
```

```
77
78          // finPlayer -> finBox    -> floor
79          // finish    -> finPlayer -> box
80          if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == floor)
81          {
82              gpi_py += 1;
83              this->gpi_yPlayer = gpi_py;
84              this->gpi_map[gpi_px][gpi_py - 1] = finish;
85              this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
86              this->gpi_map[gpi_px][gpi_py + 1] = box;
87              return;
88          }
89
90          // player -> box    -> finish
91          // floor  -> player -> finBox
92          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == finish)
93          {
94              gpi_py += 1;
95              this->gpi_yPlayer = gpi_py;
96              this->gpi_map[gpi_px][gpi_py - 1] = floor;
97              this->gpi_map[gpi_px][gpi_py    ] = player;
98              this->gpi_map[gpi_px][gpi_py + 1] = finBox;
99              return;
100         }
101
102         // finPlayer -> box    -> finish
103         // finish    -> player -> finBox
104         if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == finish)
105         {
106             gpi_py += 1;
107             this->gpi_yPlayer = gpi_py;
108             this->gpi_map[gpi_px][gpi_py - 1] = finish;
109             this->gpi_map[gpi_px][gpi_py    ] = player;
110             this->gpi_map[gpi_px][gpi_py + 1] = finBox;
111             return;
112         }
113
114         // player -> finBox    -> finish
115         // floor  -> finPlayer -> finBox
116         if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == finish)
117         {
118             gpi_py += 1;
119             this->gpi_yPlayer = gpi_py;
120             this->gpi_map[gpi_px][gpi_py - 1] = floor;
121             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
122             this->gpi_map[gpi_px][gpi_py + 1] = finBox;
123             return;
124         }
125
126         // finPlayer -> finBox    -> finish
127         // finish    -> finPlayer -> finBox
128         if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == finish)
129         {
130             gpi_py += 1;
131             this->gpi_yPlayer = gpi_py;
132             this->gpi_map[gpi_px][gpi_py - 1] = finish;
133             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
134             this->gpi_map[gpi_px][gpi_py + 1] = finBox;
135             return;
136         }
```

```
137
138         // player -> finish
139         // floor  -> finPlayer
140         if (gpi_c0 == player && gpi_c1 == finish)
141         {
142             gpi_py += 1;
143             this->gpi_yPlayer = gpi_py;
144             this->gpi_map[gpi_px][gpi_py - 1] = floor;
145             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
146             return;
147         }
148
149         // finPlayer -> finish
150         // finish    -> finPlayer
151         if (gpi_c0 == finPlayer && gpi_c1 == finish)
152         {
153             gpi_py += 1;
154             this->gpi_yPlayer = gpi_py;
155             this->gpi_map[gpi_px][gpi_py - 1] = finish;
156             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
157             return;
158         }
159     }
160 }
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_goLeft.cpp

```
1  #include "gpi_gamewindow.hpp"
2  #include "ui_gpi_gamewindow.h"
3
4  void gpi_GameWindow::gpi_goLeft ()
5  {
6      int          gpi_px          = this->gpi_xPlayer;
7      int          gpi_py          = this->gpi_yPlayer;
8      gpi_MapChar gpi_c0;
9      gpi_MapChar gpi_c1;
10     gpi_MapChar gpi_c2;
11     // = = = = = = = = = = = = = = = = =
12
13     if (gpi_px != 0)
14     {
15         gpi_c0 = this->gpi_map[gpi_px    ][gpi_py];
16         gpi_c1 = this->gpi_map[gpi_px - 1][gpi_py];
17         gpi_c2 = this->gpi_map[gpi_px - 2][gpi_py];
18
19         // player -> floor
20         // floor  -> player
21         if (gpi_c0 == player && gpi_c1 == floor)
22         {
23             gpi_px -= 1;
24             this->gpi_xPlayer = gpi_px;
25             this->gpi_map[gpi_px + 1][gpi_py] = floor;
26             this->gpi_map[gpi_px    ][gpi_py] = player;
27             return;
28         }
29
30         // finPlayer -> floor
31         // finish    -> player
32         if (gpi_c0 == finPlayer && gpi_c1 == floor)
33         {
```

```
34              gpi_px -= 1;
35              this->gpi_xPlayer = gpi_px;
36              this->gpi_map[gpi_px + 1][gpi_py] = finish;
37              this->gpi_map[gpi_px     ][gpi_py] = player;
38              return;
39          }
40
41          // player -> box    -> floor
42          // floor  -> player -> box
43          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == floor)
44          {
45              gpi_px -= 1;
46              this->gpi_xPlayer = gpi_px;
47              this->gpi_map[gpi_px + 1][gpi_py] = floor;
48              this->gpi_map[gpi_px     ][gpi_py] = player;
49              this->gpi_map[gpi_px - 1][gpi_py] = box;
50              return;
51          }
52
53          // finPlayer -> box    -> floor
54          // finish    -> player -> box
55          if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == floor)
56          {
57              gpi_px -= 1;
58              this->gpi_xPlayer = gpi_px;
59              this->gpi_map[gpi_px + 1][gpi_py] = finish;
60              this->gpi_map[gpi_px     ][gpi_py] = player;
61              this->gpi_map[gpi_px - 1][gpi_py] = box;
62              return;
63          }
64
65          // player -> finBox    -> floor
66          // floor  -> finPlayer -> box
67          if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == floor)
68          {
69              gpi_px -= 1;
70              this->gpi_xPlayer = gpi_px;
71              this->gpi_map[gpi_px + 1][gpi_py] = floor;
72              this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
73              this->gpi_map[gpi_px - 1][gpi_py] = box;
74              return;
75          }
76
77          // finPlayer -> finBox    -> floor
78          // finish    -> finPlayer -> box
79          if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == floor)
80          {
81              gpi_px -= 1;
82              this->gpi_xPlayer = gpi_px;
83              this->gpi_map[gpi_px + 1][gpi_py] = finish;
84              this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
85              this->gpi_map[gpi_px - 1][gpi_py] = box;
86              return;
87          }
88
89          // player -> box    -> finish
90          // floor  -> player -> finBox
91          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == finish)
92          {
93              gpi_px -= 1;
```

```
94              this->gpi_xPlayer = gpi_px;
95              this->gpi_map[gpi_px + 1][gpi_py] = floor;
96              this->gpi_map[gpi_px     ][gpi_py] = player;
97              this->gpi_map[gpi_px - 1][gpi_py] = finBox;
98              return;
99          }
100
101         // finPlayer -> box    -> finish
102         // finish    -> player -> finBox
103         if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == finish)
104         {
105             gpi_px -= 1;
106             this->gpi_xPlayer = gpi_px;
107             this->gpi_map[gpi_px + 1][gpi_py] = finish;
108             this->gpi_map[gpi_px     ][gpi_py] = player;
109             this->gpi_map[gpi_px - 1][gpi_py] = finBox;
110             return;
111         }
112
113         // player -> finBox    -> finish
114         // floor  -> finPlayer -> finBox
115         if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == finish)
116         {
117             gpi_px -= 1;
118             this->gpi_xPlayer = gpi_px;
119             this->gpi_map[gpi_px + 1][gpi_py] = floor;
120             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
121             this->gpi_map[gpi_px - 1][gpi_py] = finBox;
122             return;
123         }
124
125         // finPlayer -> finBox    -> finish
126         // finish    -> finPlayer -> finBox
127         if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == finish)
128         {
129             gpi_px -= 1;
130             this->gpi_xPlayer = gpi_px;
131             this->gpi_map[gpi_px + 1][gpi_py] = finish;
132             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
133             this->gpi_map[gpi_px - 1][gpi_py] = finBox;
134             return;
135         }
136
137         // player -> finish
138         // floor  -> finPlayer
139         if (gpi_c0 == player && gpi_c1 == finish)
140         {
141             gpi_px -= 1;
142             this->gpi_xPlayer = gpi_px;
143             this->gpi_map[gpi_px + 1][gpi_py] = floor;
144             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
145             return;
146         }
147
148         // finPlayer -> finish
149         // finish    -> finPlayer
150         if (gpi_c0 == finPlayer && gpi_c1 == finish)
151         {
152             gpi_px -= 1;
153             this->gpi_xPlayer = gpi_px;
```

```
154             this->gpi_map[gpi_px + 1][gpi_py] = finish;
155             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
156             return;
157         }
158     }
159 }
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_goRight.cpp

```
1  #include "gpi_gamewindow.hpp"
2  #include "ui_gpi_gamewindow.h"
3
4  void gpi_GameWindow::gpi_goRight ()
5  {
6      int          gpi_l          = this->gpi_length;
7      int          gpi_px         = this->gpi_xPlayer;
8      int          gpi_py         = this->gpi_yPlayer;
9      gpi_MapChar gpi_c0;
10     gpi_MapChar gpi_c1;
11     gpi_MapChar gpi_c2;
12     // = = = = = = = = = = = = = = = = = =
13
14     if (gpi_px != gpi_l - 1)
15     {
16         gpi_c0 = this->gpi_map[gpi_px     ][gpi_py];
17         gpi_c1 = this->gpi_map[gpi_px + 1][gpi_py];
18         gpi_c2 = this->gpi_map[gpi_px + 2][gpi_py];
19
20         // player -> floor
21         // floor  -> player
22         if (gpi_c0 == player && gpi_c1 == floor)
23         {
24             gpi_px += 1;
25             this->gpi_xPlayer = gpi_px;
26             this->gpi_map[gpi_px - 1][gpi_py] = floor;
27             this->gpi_map[gpi_px     ][gpi_py] = player;
28             return;
29         }
30
31         // finPlayer -> floor
32         // finish    -> player
33         if (gpi_c0 == finPlayer && gpi_c1 == floor)
34         {
35             gpi_px += 1;
36             this->gpi_xPlayer = gpi_px;
37             this->gpi_map[gpi_px - 1][gpi_py] = finish;
38             this->gpi_map[gpi_px     ][gpi_py] = player;
39             return;
40         }
41
42         // player -> box    -> floor
43         // floor  -> player -> box
44         if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == floor)
45         {
46             gpi_px += 1;
47             this->gpi_xPlayer = gpi_px;
48             this->gpi_map[gpi_px - 1][gpi_py] = floor;
49             this->gpi_map[gpi_px     ][gpi_py] = player;
50             this->gpi_map[gpi_px + 1][gpi_py] = box;
51             return;
```

```
52          }
53
54          // finPlayer -> box    -> floor
55          // finish    -> player -> box
56          if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == floor)
57          {
58              gpi_px += 1;
59              this->gpi_xPlayer = gpi_px;
60              this->gpi_map[gpi_px - 1][gpi_py] = finish;
61              this->gpi_map[gpi_px     ][gpi_py] = player;
62              this->gpi_map[gpi_px + 1][gpi_py] = box;
63              return;
64          }
65
66          // finPlayer -> finBox -> floor
67          // finish    -> finPlayer -> box
68          if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == floor)
69          {
70              gpi_px += 1;
71              this->gpi_xPlayer = gpi_px;
72              this->gpi_map[gpi_px - 1][gpi_py] = finish;
73              this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
74              this->gpi_map[gpi_px + 1][gpi_py] = box;
75              return;
76          }
77
78          // player -> finBox    -> floor
79          // floor  -> finPlayer -> box
80          if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == floor)
81          {
82              gpi_px += 1;
83              this->gpi_xPlayer = gpi_px;
84              this->gpi_map[gpi_px - 1][gpi_py] = floor;
85              this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
86              this->gpi_map[gpi_px + 1][gpi_py] = box;
87              return;
88          }
89
90          // player -> box    -> finish
91          // floor  -> player -> finBox
92          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == finish)
93          {
94              gpi_px += 1;
95              this->gpi_xPlayer = gpi_px;
96              this->gpi_map[gpi_px - 1][gpi_py] = floor;
97              this->gpi_map[gpi_px     ][gpi_py] = player;
98              this->gpi_map[gpi_px + 1][gpi_py] = finBox;
99              return;
100         }
101
102         // finPlayer -> box    -> finish
103         // finish    -> player -> finBox
104         if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == finish)
105         {
106             gpi_px += 1;
107             this->gpi_xPlayer = gpi_px;
108             this->gpi_map[gpi_px - 1][gpi_py] = finish;
109             this->gpi_map[gpi_px     ][gpi_py] = player;
110             this->gpi_map[gpi_px + 1][gpi_py] = finBox;
111             return;
```

```
112         }
113
114         // player -> finBox    -> finish
115         // floor  -> finPlayer -> finBox
116         if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == finish)
117         {
118             gpi_px += 1;
119             this->gpi_xPlayer = gpi_px;
120             this->gpi_map[gpi_px - 1][gpi_py] = floor;
121             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
122             this->gpi_map[gpi_px + 1][gpi_py] = finBox;
123             return;
124         }
125
126         // finPlayer -> finBox    -> finish
127         // finish    -> finPlayer -> finBox
128         if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == finish)
129         {
130             gpi_px += 1;
131             this->gpi_xPlayer = gpi_px;
132             this->gpi_map[gpi_px - 1][gpi_py] = finish;
133             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
134             this->gpi_map[gpi_px + 1][gpi_py] = finBox;
135             return;
136         }
137
138         // player -> finish
139         // floor  -> finPlayer
140         if (gpi_c0 == player && gpi_c1 == finish)
141         {
142             gpi_px += 1;
143             this->gpi_xPlayer = gpi_px;
144             this->gpi_map[gpi_px - 1][gpi_py] = floor;
145             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
146             return;
147         }
148
149         // finPlayer -> finish
150         // finish    -> finPlayer
151         if (gpi_c0 == finPlayer && gpi_c1 == finish)
152         {
153             gpi_px += 1;
154             this->gpi_xPlayer = gpi_px;
155             this->gpi_map[gpi_px - 1][gpi_py] = finish;
156             this->gpi_map[gpi_px     ][gpi_py] = finPlayer;
157             return;
158         }
159     }
160 }
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_goTop.cpp

```
1  #include "gpi_gamewindow.hpp"
2  #include "ui_gpi_gamewindow.h"
3
4  void gpi_GameWindow::gpi_goTop ()
5  {
6      int          gpi_px          = this->gpi_xPlayer;
7      int          gpi_py          = this->gpi_yPlayer;
8      gpi_MapChar gpi_c0;
```

```
9       gpi_MapChar gpi_c1;
10      gpi_MapChar gpi_c2;
11      // = = = = = = = = = = = = = = = = =
12
13      if (gpi_py != 0)
14      {
15          gpi_c0 = this->gpi_map[gpi_px][gpi_py];
16          gpi_c1 = this->gpi_map[gpi_px][gpi_py - 1];
17          gpi_c2 = this->gpi_map[gpi_px][gpi_py - 2];
18
19          // player -> floor
20          // floor  -> player
21          if (gpi_c0 == player && gpi_c1 == floor)
22          {
23              gpi_py -= 1;
24              this->gpi_yPlayer = gpi_py;
25              this->gpi_map[gpi_px][gpi_py + 1] = floor;
26              this->gpi_map[gpi_px][gpi_py    ] = player;
27              return;
28          }
29
30          // finPlayer -> floor
31          // finish    -> player
32          if (gpi_c0 == finPlayer && gpi_c1 == floor)
33          {
34              gpi_py -= 1;
35              this->gpi_yPlayer = gpi_py;
36              this->gpi_map[gpi_px][gpi_py + 1] = finish;
37              this->gpi_map[gpi_px][gpi_py    ] = player;
38              return;
39          }
40
41          // player -> box    -> floor
42          // floor  -> player -> box
43          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == floor)
44          {
45              gpi_py -= 1;
46              this->gpi_yPlayer = gpi_py;
47              this->gpi_map[gpi_px][gpi_py + 1] = floor;
48              this->gpi_map[gpi_px][gpi_py    ] = player;
49              this->gpi_map[gpi_px][gpi_py - 1] = box;
50              return;
51          }
52
53          // finPlayer -> box    -> floor
54          // finish    -> player -> box
55          if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == floor)
56          {
57              gpi_py -= 1;
58              this->gpi_yPlayer = gpi_py;
59              this->gpi_map[gpi_px][gpi_py + 1] = finish;
60              this->gpi_map[gpi_px][gpi_py    ] = player;
61              this->gpi_map[gpi_px][gpi_py - 1] = box;
62              return;
63          }
64
65          // player -> box    -> finish
66          // floor  -> player -> finBox
67          if (gpi_c0 == player && gpi_c1 == box && gpi_c2 == finish)
68          {
```

```
69              gpi_py -= 1;
70              this->gpi_yPlayer = gpi_py;
71              this->gpi_map[gpi_px][gpi_py + 1] = floor;
72              this->gpi_map[gpi_px][gpi_py    ] = player;
73              this->gpi_map[gpi_px][gpi_py - 1] = finBox;
74              return;
75          }
76
77          // finPlayer -> box    -> finish
78          // finish    -> player -> finBox
79          if (gpi_c0 == finPlayer && gpi_c1 == box && gpi_c2 == finish)
80          {
81              gpi_py -= 1;
82              this->gpi_yPlayer = gpi_py;
83              this->gpi_map[gpi_px][gpi_py + 1] = finish;
84              this->gpi_map[gpi_px][gpi_py    ] = player;
85              this->gpi_map[gpi_px][gpi_py - 1] = finBox;
86              return;
87          }
88
89          // player -> finBox    -> floor
90          // floor  -> finPlayer -> box
91          if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == floor)
92          {
93              gpi_py -= 1;
94              this->gpi_yPlayer = gpi_py;
95              this->gpi_map[gpi_px][gpi_py + 1] = floor;
96              this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
97              this->gpi_map[gpi_px][gpi_py - 1] = box;
98              return;
99          }
100
101         // finPlayer -> finBox    -> floor
102         // finish    -> finPlayer -> box
103         if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == floor)
104         {
105             gpi_py -= 1;
106             this->gpi_yPlayer = gpi_py;
107             this->gpi_map[gpi_px][gpi_py + 1] = finish;
108             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
109             this->gpi_map[gpi_px][gpi_py - 1] = box;
110             return;
111         }
112
113         // player -> finBox    -> finish
114         // floor  -> finPlayer -> finBox
115         if (gpi_c0 == player && gpi_c1 == finBox && gpi_c2 == finish)
116         {
117             gpi_py -= 1;
118             this->gpi_yPlayer = gpi_py;
119             this->gpi_map[gpi_px][gpi_py + 1] = floor;
120             this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
121             this->gpi_map[gpi_px][gpi_py - 1] = finBox;
122             return;
123         }
124
125         // finPlayer -> finBox    -> finish
126         // finish    -> finPlayer -> finBox
127         if (gpi_c0 == finPlayer && gpi_c1 == finBox && gpi_c2 == finish)
128         {
```

```
            gpi_py -= 1;
            this->gpi_yPlayer = gpi_py;
            this->gpi_map[gpi_px][gpi_py + 1] = finish;
            this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
            this->gpi_map[gpi_px][gpi_py - 1] = finBox;
            return;
        }

        // player -> finish
        // floor  -> finPlayer
        if (gpi_c0 == player && gpi_c1 == finish)
        {
            gpi_py -= 1;
            this->gpi_yPlayer = gpi_py;
            this->gpi_map[gpi_px][gpi_py + 1] = floor;
            this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
            return;
        }

        // finPlayer -> finish
        // finish    -> finPlayer
        if (gpi_c0 == finPlayer && gpi_c1 == finish)
        {
            gpi_py -= 1;
            this->gpi_yPlayer = gpi_py;
            this->gpi_map[gpi_px][gpi_py + 1] = finish;
            this->gpi_map[gpi_px][gpi_py    ] = finPlayer;
            return;
        }
    }
}
```

../gpi_osisp5_option5/gpi_gamewindow__gpi_sayWon.cpp

```
#include "gpi_gamewindow.hpp"
#include "ui_gpi_gamewindow.h"

void gpi_GameWindow::gpi_sayWon ()
{
    const int    gpi_w           = this->gpi_WinWidth;
    const int    gpi_l           = this->gpi_length;
    int          gpi_i;
    int          gpi_j;
    QMessageBox gpi_msgBox;
    // = = = = = = = = = = = = = = = = =

    gpi_i = 0;
    while (gpi_i < gpi_l)
    {
        gpi_j = 0;
        while (gpi_j < gpi_l)
        {
            if (
                    this->gpi_map[gpi_i][gpi_j] == finish
                    || this->gpi_map[gpi_i][gpi_j] == finPlayer
            )
            {
                return;
            }
            ++gpi_j;
```

```
27          }
28          ++gpi_i;
29      }
30
31      this->close ();
32      gpi_msgBox.setStyleSheet ("QLabel{min-width: " + QString::number (gpi_w) + "px; }");
33      gpi_msgBox.setText ("You won");
34      gpi_msgBox.setWindowTitle ("You won");
35      gpi_msgBox.exec ();
36  }
```