Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №1-2

По дисциплине: «ОСиСП»

Тема: «Разработка приложений с графическим пользовательским интерфейсом»

Выполнил:

Студент 3 курса

Группы ПО-7

Комиссаров А.Е.

Проверил:

Булей Е.В.

2022

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

**Общее задание:**

1) Выбрать тему из перечисленных ниже или предложить свою (тематика – игры, системные

программы и утилиты для ОС Windows);

2) Вписать свою фамилию напротив выбранной темы в файле;

3) Разработать программу с графическим пользовательским интерфейсом, реализующую указанный функционал, с использованием фреймворка Qt.

**Вариант №6**

6) Игра «Тетрис». Ограниченный набор фигурок (не более 3). Параметры колодца: ширина – 15 клеток, глубина – 20 клеток. Очки начисляются за полностью заполненные горизонтальные уровни клеток, при этом такие клетки исчезают.

**Ход работы:**

**Файл tetris.py**

from PyQt5.QtWidgets import QApplication, QMainWindow, QDialog, QLabel, QPushButton, QTableWidget, QHeaderView, QTableWidgetItem

from PyQt5.QtCore import pyqtSignal, QTimer, Qt

from PyQt5.QtGui import QPainter, QColor

from PyQt5 import uic, QtGui, QtCore

import random

import sys

import keyboard

#==============================================================================================================

S = [['.....', '.....', '..00.', '.00..', '.....'], ['.....', '..0..', '..00.', '...0.', '.....']]

Z = [['.....', '.....', '.00..', '..00.', '.....'], ['.....', '..0..', '.00..', '.0...', '.....']]

I = [['..0..', '..0..', '..0..', '..0..', '.....'], ['.....', '0000.', '.....', '.....', '.....']]

O = [['.....', '.....', '.00..', '.00..', '.....']]

J = [['.....', '.0...', '.000.', '.....', '.....'], ['.....', '..00.', '..0..', '..0..', '.....'],

['.....', '.....', '.000.', '...0.', '.....'], ['.....', '..0..', '..0..', '.00..', '.....']]

L = [['.....', '...0.', '.000.', '.....', '.....'], ['.....', '..0..', '..0..', '..00.', '.....'],

['.....', '.....', '.000.', '.0...', '.....'], ['.....', '.00..', '..0..', '..0..', '.....']]

T = [['.....', '..0..', '.000.', '.....', '.....'], ['.....', '..0..', '..00.', '..0..', '.....'],

['.....', '.....', '.000.', '..0..', '.....'], ['.....', '..0..', '.00..', '..0..', '.....']]

shapes = [S, Z, I, O, J, L, T]

shape\_colors = [1,2,3,4,5,6,7]

class Figure(object):

def \_\_init\_\_(self, x, y, shape):

self.x = x

self.y = y

self.shape = shape

self.color = shape\_colors[shapes.index(shape)]

self.rotation = 0

def createField(locked\_pos={}):

board = [[0 for \_ in range(15)] for \_ in range(20)]

for i in range(len(board)):

for j in range(len(board[i])):

if (j, i) in locked\_pos:

c = locked\_pos[(j, i)]

board[i][j] = c

return board

def getShape():

return Figure(7, 1, random.choice(shapes))

def convertShapeFormat(shape):

positions = []

format = shape.shape[shape.rotation % len(shape.shape)]

for i, line in enumerate(format):

row = list(line)

for j, column in enumerate(row):

if column == '0':

positions.append((shape.x + j , shape.y + i))

for i, pos in enumerate(positions):

positions[i] = (pos[0] - 2, pos[1] - 4)

return positions

def validSpace(shape, board):

accepted\_pos = [[(j, i) for j in range(15) if board[i][j] == 0] for i in range(20)]

accepted\_pos = [j for sub in accepted\_pos for j in sub]

formatted = convertShapeFormat(shape)

for pos in formatted:

if pos not in accepted\_pos:

if pos[1] > -1:

return False

return True

def checkLost(positions):

for pos in positions:

x, y = pos

if y < 1:

return True

return False

def tryRotate(current\_piece, board):

current\_piece.rotation += 1

if not (validSpace(current\_piece, board)):

current\_piece.rotation -= 1

def tryMoveLeft(current\_piece, board):

current\_piece.x -= 1

if not (validSpace(current\_piece, board)):

current\_piece.x += 1

def tryMoveRight(current\_piece, board):

current\_piece.x += 1

if not (validSpace(current\_piece, board)):

current\_piece.x -= 1

def tryMoveDown(current\_piece, board):

current\_piece.y += 1

if not (validSpace(current\_piece, board)):

current\_piece.y -= 1

def clearRows(board, locked, self):

global score

inc = 0

for i in range(len(board)-1, -1, -1):

row = board[i]

if 0 not in row:

inc += 1

ind = i

for j in range(len(row)):

try:

del locked[(j,i)]

except:

continue

if inc > 0:

for key in sorted(list(locked), key = lambda x: x[1])[::-1]:

x, y = key

if y < ind:

newKey = (x, y + inc)

locked[newKey] = locked.pop(key)

score\_map = {

0: 0,

1: 40,

2: 100,

3: 300,

4: 1200

}

score += score\_map[inc]

self.scoreText.setText("Score : " + str(score))

def checkLevel(time):

time = time // 20

level = 6

if time < 60:

level = 6

elif time < 120:

level = 5

elif time < 180:

level = 4

elif time < 360:

level = 3

elif time < 600:

level = 2

else:

level = 1

return level

locked\_positions = {}

board = createField(locked\_positions)

change\_piece = False

current\_piece = getShape()

next\_piece = getShape()

score = 0

class PlayWindow(QDialog):

closed = pyqtSignal() #signal attribute for parent window

def \_\_init\_\_(self):

print("Creating PlayWindow object:")#

super(PlayWindow, self).\_\_init\_\_()

print("- Loading PlayWindow object UI...")#

uic.loadUi("UI/PlayWindow.ui", self)

#------------

print("- Creating PlayWindow object variables and finding UI elements...")#

self.pause = 1

self.buttonExit = self.findChild(QPushButton, "ExitButton")

self.tableWidget = self.findChild(QTableWidget, "PlayTable")

self.figureWidget = self.findChild(QTableWidget, "FigureWindow")

self.buttonPause = self.findChild(QPushButton, "PauseButton")

self.timeText = self.findChild(QLabel, "TimeText")

self.timestr = "Time passed : "

self.scoreText = self.findChild(QLabel, "ScoreText")

self.buttonPause.clicked.connect(self.PauseGame)

self.buttonExit.clicked.connect(self.CloseWindow)

#------------

print("- Setting PlayWindow resize modes...")#

self.tableWidget.horizontalHeader().setSectionResizeMode(QHeaderView.Stretch)

self.tableWidget.verticalHeader().setSectionResizeMode(QHeaderView.Stretch)

self.figureWidget.horizontalHeader().setSectionResizeMode(QHeaderView.Stretch)

self.figureWidget.verticalHeader().setSectionResizeMode(QHeaderView.Stretch)

print("- Creating the gameTimer...")#

self.gameTimer = QTimer()

self.gameTimer.setInterval(50)

self.gameTimer.timeout.connect(self.GameStateUpdate)

print("- Creating game vars...")#

self.time = 0

print("Done with PlayWindow.")#

def GameStateUpdate(self):#########################################

global current\_piece

global board

global change\_piece

global next\_piece

global score

if(self.pause != 1):

print("update : ", self.time)

board = createField(locked\_positions)

level = checkLevel(self.time)

if (self.time % level == 0) : current\_piece.y += 1

if not(validSpace(current\_piece, board)) and current\_piece.y > 0:

current\_piece.y -= 1

change\_piece = True

if keyboard.is\_pressed('w') or keyboard.is\_pressed('up'):

if (self.time % 2 == 0) : tryRotate(current\_piece, board)

if keyboard.is\_pressed('s') or keyboard.is\_pressed('down'):

tryMoveDown(current\_piece, board)

if keyboard.is\_pressed('a') or keyboard.is\_pressed('left'):

tryMoveLeft(current\_piece, board)

if keyboard.is\_pressed('d') or keyboard.is\_pressed('right'):

tryMoveRight(current\_piece, board)

if keyboard.is\_pressed('space'):

print('spacebar')

self.time += 1

self.timeText.setText(self.timestr + str(self.time//20) + "s")

shape\_pos = convertShapeFormat(current\_piece)

for i in range(len(shape\_pos)):

x, y = shape\_pos[i]

if y > -1:

board[y][x] = current\_piece.color

if change\_piece:

for pos in shape\_pos:

p = (pos[0], pos[1])

locked\_positions[p] = current\_piece.color

current\_piece = next\_piece

next\_piece = getShape()

change\_piece = False

clearRows(board, locked\_positions, self)

if checkLost(locked\_positions):

self.timeText.setText("Game Over!")

self.PauseGame()

self.buttonPause.setEnabled(False)

self.buttonPause.setStyleSheet("""

QPushButton{background-color: rgb(128, 128, 128);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(0, 0, 0);

}

QPushButton:hover{

background-color: rgb(160, 160, 160);

}

""")

self.leadb = Leaderboard(score, 6-level) #create leaderboard child window

self.leadb.closed.connect(self.show) #show main title window when closing leaderboard window

self.leadb.show()

self.UpdateCell()

self.UpdateFigure(next\_piece)

else:

if keyboard.is\_pressed('p'):

self.PauseGame()

print("update : ", self.time, " (paused)")

###############################################################

def UpdateCell(self):

for y in range(20):

for x in range(15):

item = board[y][x]

tableWidgetItem = QTableWidgetItem(str(item))

self.tableWidget.setItem(y, x, tableWidgetItem)

if(self.tableWidget.item(y,x).text() == '0'):blockColor = QtGui.QColor(0,0,0)

if(self.tableWidget.item(y,x).text() == '1'):blockColor = QtGui.QColor(255,0,0)

if(self.tableWidget.item(y,x).text() == '2'):blockColor = QtGui.QColor(0,255,0)

if(self.tableWidget.item(y,x).text() == '3'):blockColor = QtGui.QColor(0,0,255)

if(self.tableWidget.item(y,x).text() == '4'):blockColor = QtGui.QColor(255,255,0)

if(self.tableWidget.item(y,x).text() == '5'):blockColor = QtGui.QColor(255,0,255)

if(self.tableWidget.item(y,x).text() == '6'):blockColor = QtGui.QColor(0,255,255)

if(self.tableWidget.item(y,x).text() == '7'):blockColor = QtGui.QColor(255,255,255)

self.tableWidget.item(y, x).setBackground(blockColor)

def UpdateFigure(self, shape):

for y in range(5):

for x in range(5):

self.figureWidget.setItem(y, x, QTableWidgetItem(str('.')))

if(shape.color == 1):blockColor = QtGui.QColor(255,0,0)

if(shape.color == 2):blockColor = QtGui.QColor(0,255,0)

if(shape.color == 3):blockColor = QtGui.QColor(0,0,255)

if(shape.color == 4):blockColor = QtGui.QColor(255,255,0)

if(shape.color == 5):blockColor = QtGui.QColor(255,0,255)

if(shape.color == 6):blockColor = QtGui.QColor(0,255,255)

if(shape.color == 7):blockColor = QtGui.QColor(255,255,255)

format = shape.shape[shape.rotation % len(shape.shape)]

for i, line in enumerate(format):

row = list(line)

for j, column in enumerate(row):

if column == '.':

self.figureWidget.item(i, j).setBackground(QtGui.QColor(0,0,0))

if column == '0':

self.figureWidget.item(i, j).setBackground(blockColor)

def PauseGame(self):

#---------#is paused

if(self.pause):

self.buttonPause.setStyleSheet("""

QPushButton{background-color: rgb(42, 39, 37);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(85, 255, 127);

}

QPushButton:hover{

background-color: rgb(50, 47, 45);

}

""")

self.buttonPause.setText("Pause")

self.pause = 0

self.gameTimer.start()

#---------#is unpaused

else:

self.buttonPause.setStyleSheet("""

QPushButton{background-color: rgb(85, 255, 127);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(42, 39, 37);

}

QPushButton:hover{

background-color: rgb(100, 255, 140);

}

""")

self.buttonPause.setText("Start")

self.pause = 1

def CloseWindow(self):

self.pause = 0

self.PauseGame()

self.gameTimer.stop()

print("game stopped")

self.close()

QtCore.QCoreApplication.quit()

status = QtCore.QProcess.startDetached(sys.executable, sys.argv)

#emit signal on window closure

def closeEvent(self, event):

self.closed.emit()

QDialog.closeEvent(self, event)

#==============================================================================================================

#Leaderboard (opens after pressing Leaderboard button on title window)

class Leaderboard(QDialog):

closed = pyqtSignal() #signal attribute for parent window

def \_\_init\_\_(self, score, level):

super(Leaderboard, self).\_\_init\_\_()

uic.loadUi("UI/Leaderboard.ui", self)

#------------

self.buttonClose = self.findChild(QPushButton, "CloseButton")

self.buttonClose.clicked.connect(self.CloseWindow)

self.scoreText = self.findChild(QLabel, "ScoreText")

self.levelText = self.findChild(QLabel, "LevelText")

#------------

self.scoreText.setText("Score : " + str(score))

self.levelText.setText("Level : " + str(level))

#exit button is pressed

def CloseWindow(self):

self.close() #exit leaderboard window

QtCore.QCoreApplication.quit()

status = QtCore.QProcess.startDetached(sys.executable, sys.argv)

#emit signal on window closure

def closeEvent(self, event):

self.closed.emit()

QDialog.closeEvent(self, event)

#==============================================================================================================

#StartWindow (title window, opened on program startup)

class StartWindow(QDialog):

def \_\_init\_\_(self):

super(StartWindow, self).\_\_init\_\_()

uic.loadUi("UI/Title.ui", self)

#------------

self.buttonStart = self.findChild(QPushButton, "StartButton")

self.buttonStart.clicked.connect(self.StartB)

self.buttonExit = self.findChild(QPushButton, "ExitButton")

self.buttonExit.clicked.connect(self.ExitB)

#------------

self.mainw = PlayWindow() #create playwindow child

self.mainw.closed.connect(self.show) #show title when closing playwindow

self.show()

#Start button is pressed

def StartB(self):

self.hide()

self.mainw.show()

#Exit button is pressed

def ExitB(self):

quit()

#==============================================================================================================

app = QApplication(sys.argv)

UIWindow = StartWindow()

app.exec\_()

**Содержимое файла PlayWindow.UI**

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>Dialog</class>

<widget class="QDialog" name="Dialog">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>640</width>

<height>480</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="windowTitle">

<string>Dialog</string>

</property>

<property name="styleSheet">

<string notr="true">background-color: rgb(60, 53, 46);</string>

</property>

<widget class="QTableWidget" name="PlayTable">

<property name="enabled">

<bool>false</bool>

</property>

<property name="geometry">

<rect>

<x>50</x>

<y>40</y>

<width>300</width>

<height>400</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>300</width>

<height>400</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>300</width>

<height>400</height>

</size>

</property>

<property name="font">

<font>

<pointsize>2</pointsize>

</font>

</property>

<property name="cursor" stdset="0">

<cursorShape>CrossCursor</cursorShape>

</property>

<property name="focusPolicy">

<enum>Qt::NoFocus</enum>

</property>

<property name="styleSheet">

<string notr="true">font-size: 0px;

color: white;

text-indent: 50px;

letter-spacing: -10px;

line-height: 0.8;

white-space: nowrap;</string>

</property>

<property name="frameShape">

<enum>QFrame::Panel</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Plain</enum>

</property>

<property name="verticalScrollBarPolicy">

<enum>Qt::ScrollBarAlwaysOff</enum>

</property>

<property name="horizontalScrollBarPolicy">

<enum>Qt::ScrollBarAlwaysOff</enum>

</property>

<property name="sizeAdjustPolicy">

<enum>QAbstractScrollArea::AdjustToContents</enum>

</property>

<property name="autoScroll">

<bool>false</bool>

</property>

<property name="autoScrollMargin">

<number>0</number>

</property>

<property name="editTriggers">

<set>QAbstractItemView::NoEditTriggers</set>

</property>

<property name="tabKeyNavigation">

<bool>false</bool>

</property>

<property name="showDropIndicator" stdset="0">

<bool>false</bool>

</property>

<property name="dragDropOverwriteMode">

<bool>false</bool>

</property>

<property name="selectionMode">

<enum>QAbstractItemView::NoSelection</enum>

</property>

<property name="textElideMode">

<enum>Qt::ElideMiddle</enum>

</property>

<property name="showGrid">

<bool>true</bool>

</property>

<property name="gridStyle">

<enum>Qt::DashLine</enum>

</property>

<property name="wordWrap">

<bool>true</bool>

</property>

<property name="rowCount">

<number>20</number>

</property>

<property name="columnCount">

<number>15</number>

</property>

<attribute name="horizontalHeaderVisible">

<bool>false</bool>

</attribute>

<attribute name="horizontalHeaderMinimumSectionSize">

<number>0</number>

</attribute>

<attribute name="horizontalHeaderDefaultSectionSize">

<number>20</number>

</attribute>

<attribute name="horizontalHeaderHighlightSections">

<bool>false</bool>

</attribute>

<attribute name="verticalHeaderVisible">

<bool>false</bool>

</attribute>

<attribute name="verticalHeaderMinimumSectionSize">

<number>0</number>

</attribute>

<attribute name="verticalHeaderDefaultSectionSize">

<number>20</number>

</attribute>

<attribute name="verticalHeaderHighlightSections">

<bool>false</bool>

</attribute>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<row/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<column/>

<item row="0" column="0">

<property name="text">

<string>1</string>

</property>

</item>

</widget>

<widget class="QLabel" name="ScoreText">

<property name="geometry">

<rect>

<x>370</x>

<y>60</y>

<width>240</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>18</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>IBeamCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">Border: 1px solid black;

border-radius: 20%;

padding: 10px;

color: rgb(255, 215, 0);</string>

</property>

<property name="text">

<string>Score : 0</string>

</property>

</widget>

<widget class="QLabel" name="TimeText">

<property name="geometry">

<rect>

<x>370</x>

<y>130</y>

<width>240</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>18</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>IBeamCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">Border: 1px solid black;

border-radius: 20%;

padding: 10px;

color: rgb(255, 215, 0);</string>

</property>

<property name="text">

<string>Time passed : 0</string>

</property>

</widget>

<widget class="QPushButton" name="ExitButton">

<property name="geometry">

<rect>

<x>500</x>

<y>360</y>

<width>110</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>110</width>

<height>50</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>110</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2</family>

<pointsize>25</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>PointingHandCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">QPushButton{background-color: rgb(42, 39, 37);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(255, 9, 1);}

QPushButton:hover{

background-color: rgb(50, 47, 45);

}</string>

</property>

<property name="text">

<string>Exit</string>

</property>

</widget>

<widget class="QPushButton" name="PauseButton">

<property name="geometry">

<rect>

<x>380</x>

<y>360</y>

<width>110</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>110</width>

<height>50</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>110</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2</family>

<pointsize>25</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>PointingHandCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">QPushButton{background-color: rgb(85, 255, 127);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(42, 39, 37);

}

QPushButton:hover{

background-color: rgb(100, 255, 140);

}</string>

</property>

<property name="text">

<string>Start</string>

</property>

</widget>

<widget class="QTableWidget" name="FigureWindow">

<property name="geometry">

<rect>

<x>510</x>

<y>220</y>

<width>100</width>

<height>100</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>100</width>

<height>100</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>100</width>

<height>100</height>

</size>

</property>

<property name="font">

<font>

<pointsize>8</pointsize>

</font>

</property>

<property name="focusPolicy">

<enum>Qt::NoFocus</enum>

</property>

<property name="styleSheet">

<string notr="true">border: 1px solid rgb(125, 109, 0);

border-left: 0px;</string>

</property>

<property name="frameShape">

<enum>QFrame::Panel</enum>

</property>

<property name="frameShadow">

<enum>QFrame::Plain</enum>

</property>

<property name="verticalScrollBarPolicy">

<enum>Qt::ScrollBarAlwaysOff</enum>

</property>

<property name="horizontalScrollBarPolicy">

<enum>Qt::ScrollBarAlwaysOff</enum>

</property>

<property name="sizeAdjustPolicy">

<enum>QAbstractScrollArea::AdjustToContents</enum>

</property>

<property name="autoScroll">

<bool>false</bool>

</property>

<property name="autoScrollMargin">

<number>0</number>

</property>

<property name="editTriggers">

<set>QAbstractItemView::NoEditTriggers</set>

</property>

<property name="tabKeyNavigation">

<bool>false</bool>

</property>

<property name="showDropIndicator" stdset="0">

<bool>false</bool>

</property>

<property name="dragDropOverwriteMode">

<bool>false</bool>

</property>

<property name="selectionMode">

<enum>QAbstractItemView::NoSelection</enum>

</property>

<property name="textElideMode">

<enum>Qt::ElideMiddle</enum>

</property>

<property name="gridStyle">

<enum>Qt::DashLine</enum>

</property>

<property name="cornerButtonEnabled">

<bool>false</bool>

</property>

<property name="rowCount">

<number>5</number>

</property>

<property name="columnCount">

<number>5</number>

</property>

<attribute name="horizontalHeaderVisible">

<bool>false</bool>

</attribute>

<attribute name="horizontalHeaderMinimumSectionSize">

<number>0</number>

</attribute>

<attribute name="horizontalHeaderDefaultSectionSize">

<number>20</number>

</attribute>

<attribute name="horizontalHeaderHighlightSections">

<bool>false</bool>

</attribute>

<attribute name="verticalHeaderVisible">

<bool>false</bool>

</attribute>

<attribute name="verticalHeaderMinimumSectionSize">

<number>0</number>

</attribute>

<attribute name="verticalHeaderDefaultSectionSize">

<number>20</number>

</attribute>

<attribute name="verticalHeaderHighlightSections">

<bool>false</bool>

</attribute>

<row/>

<row/>

<row/>

<row/>

<row/>

<column/>

<column/>

<column/>

<column/>

<column/>

</widget>

<widget class="QLabel" name="textFigure">

<property name="geometry">

<rect>

<x>370</x>

<y>220</y>

<width>140</width>

<height>100</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>140</width>

<height>100</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>140</width>

<height>100</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>18</pointsize>

</font>

</property>

<property name="styleSheet">

<string notr="true">color: rgb(255, 215, 0);

border: 1px solid rgb(125, 109, 0);

border-right: 0px;</string>

</property>

<property name="text">

<string>Next figure : </string>

</property>

</widget>

</widget>

<resources/>

<connections/>

</ui>

**Содержимое файла Leaderboard.UI**

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>Dialog</class>

<widget class="QDialog" name="Dialog">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>470</width>

<height>290</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="windowTitle">

<string>Dialog</string>

</property>

<property name="styleSheet">

<string notr="true">background-color: rgb(60, 53, 46);</string>

</property>

<widget class="QLabel" name="textLeaderboard">

<property name="geometry">

<rect>

<x>95</x>

<y>20</y>

<width>280</width>

<height>40</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>280</width>

<height>40</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>280</width>

<height>40</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>22</pointsize>

<weight>75</weight>

<bold>true</bold>

</font>

</property>

<property name="styleSheet">

<string notr="true">background-color: rgb(42, 39, 37);

border-radius: 8%;

color: rgb(109, 199, 117);</string>

</property>

<property name="text">

<string>GAME OVER</string>

</property>

<property name="alignment">

<set>Qt::AlignCenter</set>

</property>

</widget>

<widget class="QPushButton" name="CloseButton">

<property name="geometry">

<rect>

<x>135</x>

<y>220</y>

<width>200</width>

<height>40</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>200</width>

<height>40</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>200</width>

<height>40</height>

</size>

</property>

<property name="font">

<font>

<family>TF2</family>

<pointsize>25</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>PointingHandCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">QPushButton{background-color: rgb(42, 39, 37);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(255, 9, 1);}

QPushButton:hover{

background-color: rgb(50, 47, 45);

}</string>

</property>

<property name="text">

<string>Close</string>

</property>

</widget>

<widget class="QLabel" name="LevelText">

<property name="geometry">

<rect>

<x>140</x>

<y>150</y>

<width>190</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>0</width>

<height>0</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>18</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>IBeamCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">Border: 1px solid black;

border-radius: 20%;

padding: 10px;

color: rgb(255, 215, 0);</string>

</property>

<property name="text">

<string>Level : 1</string>

</property>

</widget>

<widget class="QLabel" name="ScoreText">

<property name="geometry">

<rect>

<x>140</x>

<y>80</y>

<width>190</width>

<height>50</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>0</width>

<height>0</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>240</width>

<height>50</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>18</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>IBeamCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">Border: 1px solid black;

border-radius: 20%;

padding: 10px;

color: rgb(255, 215, 0);</string>

</property>

<property name="text">

<string>Score : 0</string>

</property>

</widget>

</widget>

<resources/>

<connections/>

</ui>

**Содержимое файла Title.UI**

<?xml version="1.0" encoding="UTF-8"?>

<ui version="4.0">

<class>Dialog</class>

<widget class="QDialog" name="Dialog">

<property name="geometry">

<rect>

<x>0</x>

<y>0</y>

<width>360</width>

<height>480</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>300</width>

<height>480</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>640</width>

<height>480</height>

</size>

</property>

<property name="windowTitle">

<string>Dialog</string>

</property>

<property name="styleSheet">

<string notr="true">background-color: rgb(60, 53, 46);</string>

</property>

<widget class="QPushButton" name="StartButton">

<property name="geometry">

<rect>

<x>60</x>

<y>250</y>

<width>240</width>

<height>60</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>240</width>

<height>60</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>240</width>

<height>60</height>

</size>

</property>

<property name="font">

<font>

<family>TF2</family>

<pointsize>25</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>PointingHandCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">QPushButton {

background-color: rgb(42, 39, 37);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(255, 215, 0);

}

QPushButton:hover{

background-color: rgb(50, 47, 45);

}

</string>

</property>

<property name="text">

<string>Start</string>

</property>

</widget>

<widget class="QLabel" name="textTetris">

<property name="geometry">

<rect>

<x>20</x>

<y>100</y>

<width>320</width>

<height>60</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>320</width>

<height>60</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>320</width>

<height>60</height>

</size>

</property>

<property name="font">

<font>

<family>TF2 Secondary</family>

<pointsize>46</pointsize>

<weight>75</weight>

<italic>false</italic>

<bold>true</bold>

<underline>false</underline>

<strikeout>false</strikeout>

<stylestrategy>PreferDefault</stylestrategy>

<kerning>true</kerning>

</font>

</property>

<property name="styleSheet">

<string notr="true">color: rgb(255, 215, 0);</string>

</property>

<property name="frameShape">

<enum>QFrame::NoFrame</enum>

</property>

<property name="text">

<string>Tetris</string>

</property>

<property name="alignment">

<set>Qt::AlignCenter</set>

</property>

</widget>

<widget class="QPushButton" name="ExitButton">

<property name="geometry">

<rect>

<x>80</x>

<y>330</y>

<width>200</width>

<height>60</height>

</rect>

</property>

<property name="sizePolicy">

<sizepolicy hsizetype="Fixed" vsizetype="Fixed">

<horstretch>0</horstretch>

<verstretch>0</verstretch>

</sizepolicy>

</property>

<property name="minimumSize">

<size>

<width>200</width>

<height>60</height>

</size>

</property>

<property name="maximumSize">

<size>

<width>200</width>

<height>60</height>

</size>

</property>

<property name="font">

<font>

<family>TF2</family>

<pointsize>25</pointsize>

</font>

</property>

<property name="cursor">

<cursorShape>PointingHandCursor</cursorShape>

</property>

<property name="styleSheet">

<string notr="true">QPushButton{background-color: rgb(42, 39, 37);

border: 1px solid rgb(125, 109, 0);

border-radius: 8%;

color: rgb(255, 9, 1);}

QPushButton:hover{

background-color: rgb(50, 47, 45);

}</string>

</property>

<property name="text">

<string>Exit</string>

</property>

</widget>

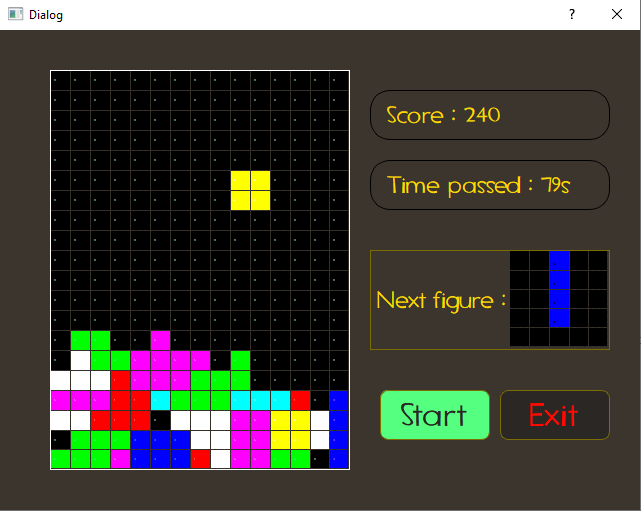
</widget>

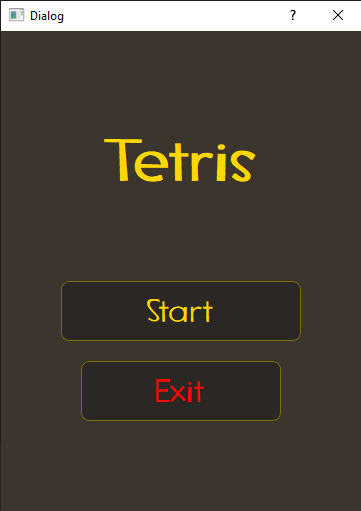
<resources/>

<connections/>

</ui>

**Результат работы программы:**

****

****

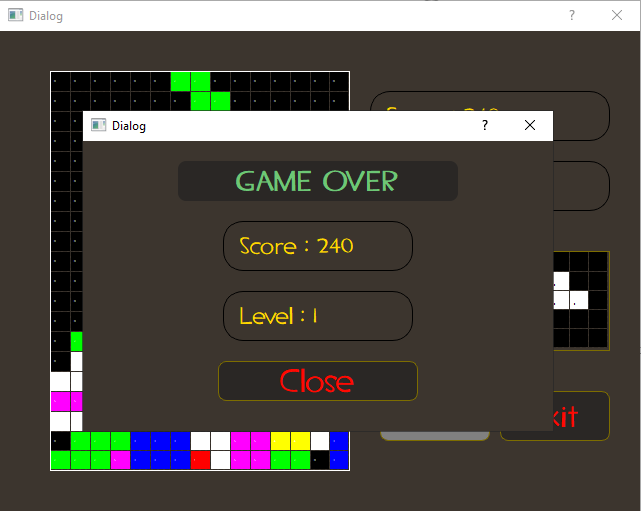
****

Рис. 1,2,3 – Результат работы программы

**Вывод:** я приобрёл практические навыки проектирования и разработки приложений с графическим пользовательским интерфейсом в ОС Windows средствами Qt.