МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С.ТУРГЕНЕВА»

Кафедра информационных систем и цифровых технологий

**ОТЧЁТ**

По лабораторной работе №1-7

на тему: «Python»

по дисциплине: «Программирование на языке Python»

Выполнили: Музалевский Н.С.

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 92ПГ

Проверила: Захарова О.В.

Отметка о зачёте:

Дата: «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021г.

Орел, 2021

**1.1.PY**

import math as m

def InputArgument(argument):

while True:

try:

return int(input("Введите переменную "+argument+": "))

except:

print("Что-то при вводе переменной " + argument + " пошло не так")

a = InputArgument("a")

n = InputArgument("n")

x = InputArgument("x")

try:

answer = 5\*m.pow(a,n\*x)/(n+x)-(m.sqrt(m.fabs(m.cos(m.pow(x,m.pow(3,n))))))

print (answer)

except ZeroDivisionError:

print ("Попытка деления на 0")

**1.2.PY**

def PrintMenuItem(number,menuItem):

print(str(number) + ") " + menuItem + ": ")

return number+1

def PrintMenu():

print("\n\n\nМеню:\n")

number = 1

number = PrintMenuItem(number,"Показать значение списка")

number = PrintMenuItem(number,"Добавить новый элемент в конец списка")

number = PrintMenuItem(number,"Удаление указанного элемента")

number = PrintMenuItem(number,"Сформировать кортеж, сотроящий из вещественных чисел меньших 100")

number = PrintMenuItem(number,"Найти произведение всех целочисленных отрицательных элементов списка")

number = PrintMenuItem(number,"Из сформировать строки (значений элементов списка), посчитать сколько раз встречается в строке указанное слово")

number = PrintMenuItem(number,"Задать с клавиатуры множество M1, сформировать множество M2 из списка; вывести на экран симметричную разницу множеств M1 и M2")

number = PrintMenuItem(number,"Получить из списка словарь, ключом каждого элемента сделать позицию элемента в словаре; построчно отобразить на экране элементы словаря с нечетными значениями ключа")

def Input(str=""):

return input ("Ввод"+str+": ")

def InputInt(str=""):

while True:

try:

return int(Input(str))

except :

print("Что-то пошло не так, повторите пожалуйста")

def InputFloat(str=""):

while True:

try:

return float(Input(str))

except :

print("Что-то пошло не так, повторите пожалуйста")

def MenuItem(menuItem, mainList):

if(menuItem == "exit"):

raise SystemExit

elif(menuItem == "1"):

ShowTheList(mainList)

elif(menuItem == "2"):

AddToEndOfList(mainList)

elif(menuItem == "3"):

ShowTheList(mainList)

answer = Input(", по индексу - 1, по значению - 2")

if(answer=="1"):

DelByIndexList(mainList, InputInt(" индекса"))

elif(answer=="2"):

DelByValueList(mainList, Input(" значания переменной"))

else:

print("Почему?")

elif(menuItem == "4"):

OnlyNumbersLess100(mainList)

elif(menuItem == "5"):

FindProductNumbers(mainList)

elif(menuItem == "6"):

CountSubstringInString(mainList)

elif(menuItem == "7"):

DifferenceOfSets(mainList)

elif(menuItem == "8"):

ShowOddNumbers(mainList)

else:

print("Нет такого пукнта")

def ShowTheList(mainList):

print(mainList)

def AddToEndOfList(mainList):

print("1-INT;2-STR;3-FLOAT")

rt = InputInt()

if(rt==1):

mainList.append(InputInt())

elif(rt==2):

mainList.append(Input())

elif(rt==3):

mainList.append(InputFloat())

def DelByIndexList(mainList, index):

try:

mainList.pop(index)

except IndexError:

print("Такого индекса нет в списке")

ShowTheList(mainList)

def DelByValueList(mainList, value):

try:

mainList.remove(value)

except ValueError:

print("Элемент не найден")

ShowTheList(mainList)

def OnlyNumbersLess100(mainList):

numbersLess100 = tuple()

for n in mainList:

if(IsNumberLess100(n)):

numbersLess100 += (n,)

print("Числа меньше 100:", numbersLess100)

def IsNumberLess100(number):

try:

if(number<100):

return True

else:

return False

except TypeError:

return False

def FindProductNumbers(mainList):

productNumbers = 1;

count = 0;

for n in mainList:

if(NegativeInteger(n)):

count+=1

productNumbers\*=n

if(count > 0):

print(productNumbers)

else:

print("В списке нет орицательных челых чисел")

def NegativeInteger(number):

try:

if(number % 1 == 0 and number<0):

return True

return False

except TypeError:

return False

def CountSubstringInString(mainList):

string = ""

for n in mainList:

string+=str(n)

substring = Input(" подстроки")

print("Всего подстрока ("+ substring +") встречается " + str(FindSubstring(string,substring,0)) + " раз(а) в строке ("+ string +")")

def FindSubstring(string,substring,count):

index = string.find(substring)

while (index !=-1):

count+=1

index+=1

index = string.find(substring,index)

return count

def DifferenceOfSets(mainList):

print("Введите множество")

set1 = {Input(),Input(),Input(),Input()}

set2 = set()

for n in mainList:

if(isinstance(n, list)):

n=str(n)

set2.add(n)

print(set1)

print(set2)

print(set1.symmetric\_difference(set2))

def ShowOddNumbers(mainList):

myDict = {}

for idx,value in enumerate(mainList):

myDict[idx] = value

for idx, value in myDict.items():

if(idx%2==1):

print(value)

#mainList = ['1', "123", ['243'], "fsf", 23, -1]

mainList = ['1', "123", ['243'], "fsf"]

#mainList = [1, "123", ['243'], "fsf"]

while True:

PrintMenu()

MenuItem(Input(), mainList)

**1.3.PY**

import math as m

def PrintMenuItem(letter,menuItem):

print("«" + letter + "» - " + menuItem)

def PrintMenu():

print("\n\nМеню:\n")

PrintMenuItem("R","Площадь прямоугольника")

PrintMenuItem("T","Площадь треугольника")

PrintMenuItem("M","Площадь многоугольника")

PrintMenuItem("E","Выход из программы")

def Input(str=""):

return input ("Ввод"+str+": ")

def InputInt(str=""):

while True:

try:

return int(Input(str))

except :

print("Что-то пошло не так, повторите пожалуйста")

def MenuItem(menuItem):

if(menuItem == "E"):

raise SystemExit

elif(menuItem == "R"):

RectangleArea(InputInt(" высоты фигуры"),InputInt(" ширины фигуры"))

elif(menuItem == "T"):

TriangleArea(InputInt(" первой стороны фигуры"), InputInt(" второй стороны фигуры"), InputInt(" третей стороны фигуры"))

elif(menuItem == "M"):

PolygonArea(InputInt(" длины сторон многоугольника"), InputInt(" количества сторон многоугольника"))

else:

print("Нет такого пукнта")

def RectangleArea(height, width):

print(height,"x",width," = ", height\*width)

def TriangleArea(first, second, third):

p = (first+second+third)/2

print("(",first,"+",second,"+",third,")/2 =",p)

s = m.sqrt(p\*(p-first)\*(p-second)\*(p-third))

print("√("+str(p)+Pminus(p,first)+Pminus(p,second)+Pminus(p,third)+") =",s)

def Pminus(p,arg):

return ("\*("+str(p)+"-"+str(arg)+")")

def PolygonArea(a,n):

s=n\*m.pow(a,2)/(4\*(m.tan(m.radians(180/n))))

print("("+str(n)+"\*"+str(a) + "^(2))"+"/(4\*tg(180/"+str(n)+")) = " + str(s))

while True:

PrintMenu()

MenuItem(Input())

**2.1.PY**

import math as m

def InputArgument(argument):

while True:

try:

return int(input("Введите переменную "+argument+": "))

except:

print("Что-то при вводе переменной " + argument + " пошло не так")

Answer = lambda a,n,x: 5\*m.pow(a,n\*x)/(n+x)-(m.sqrt(m.fabs(m.cos(m.pow(x,m.pow(3,n))))))

a = InputArgument("a")

n = InputArgument("n")

x = InputArgument("x")

try:

print (Answer(a,n,x))

except ZeroDivisionError:

print ("Попытка деления на 0")

**2.2.PY**

import math as m

L = [['R','r','M','T','T','E'],[1,2,5],[1,23],[5,5,5],[1,1,1],[2,2,2],[1]]#[3,3],[1,1,1],[2,4],[1,1,1],[1,1,1],[1]]

idx = 1

PrintMenuItem = lambda letter,menuItem: print("«" + letter + "» - " + menuItem)

PrintMenu = lambda: [print("\n\nМеню:\n"), PrintMenuItem("R","Площадь прямоугольника"),PrintMenuItem("T","Площадь треугольника"),PrintMenuItem("M","Площадь многоугольника"),PrintMenuItem("E","Выход из программы")]

Input = lambda str="": input ("Ввод"+str+": ")

def InputInt(str=""):

while True:

try:

return int(Input(str))

except :

print("Что-то пошло не так, повторите пожалуйста")

def MenuItem(Li):

print("Пункт: "+Li)

global idx

if(Li[0] == "E"):

raise SystemExit

elif(Li[0] == "R"):

print(L[idx][1])

print(L[idx][0])

RectangleArea(L[idx][0],L[idx][1])

idx+=1

elif(Li[0] == "T"):

first = L[idx][0]

second = L[idx][1]

third = L[idx][2]

p = P(first,second,third)

s = St(first,second,third,p)

TriangleArea(first,second,third,p,s)

idx+=1

elif(Li[0] == "M"):

a = L[idx][0]

n = L[idx][1]

s = Sp(a,n)

PolygonArea(a,n,s)

idx+=1

else:

print("Нет такого пукнта")

idx+=1

RectangleArea = lambda height,width:print(height,"x",width," = ", height\*width)

P = lambda first, second, third:(first+second+third)/2

St = lambda first,second,third,p: m.sqrt(p\*(p-first)\*(p-second)\*(p-third))

TriangleArea = lambda first, second, third,p,s: [print("(",first,"+",second,"+",third,")/2 =",p),print("√("+str(p)+Pminus(p,first)+Pminus(p,second)+Pminus(p,third)+") =",s)]

Pminus = lambda p,arg: ("\*("+str(p)+"-"+str(arg)+")")

Sp = lambda a,n: n\*m.pow(a,2)/(4\*(m.tan(m.radians(180/n))))

PolygonArea = lambda a,n,s: [print("("+str(n)+"\*"+str(a) + "^(2))"+"/(4\*tg(180/"+str(n)+")) = " + str(s))]

while True:

PrintMenu()

fdsfsd = list(map(MenuItem,L[0]))

**3.1.PY**

class film:

def \_\_init\_\_(self, \*\*kwargs):

for key in kwargs:

setattr(self, key, kwargs[key])

def printElement(self):

try:

str1 = self.code

except AttributeError:

str1 = "Не указано"

try:

str2 = self.name

except AttributeError:

str2 = "Не указано"

try:

str3 = self.director

except AttributeError:

str3 = "Не указано"

try:

str4 = self.releaseDate

except AttributeError:

str4 = "Не указано"

try:

str5 = self.actors

except AttributeError:

str5 = "Не указано"

print("Код: "+ str(str1) +"; Название: "+ str(str2) +"; Режиссёр: "+ str(str3) +"; Дата выхода: "+ str(str4) +"; Главный актёр: "+ str(str5) +";")

def PrintMenuItem(number,menuItem):

print(str(number) + ") " + menuItem + ": ")

return number+1

def PrintMenu():

print("\n\n\nМеню:\n")

number = 1

number = PrintMenuItem(number,"Добавление информации в список")

number = PrintMenuItem(number,"Удаление информации о выбранном объекте списка")

number = PrintMenuItem(number,"Отображение информации обо всех объектах списка в удобном виде")

number = PrintMenuItem(number,"Поиск фильмов указанного пользователем года выпуска")

def Input(str=""):

return input ("Ввод"+str+": ")

def InputInt(str=""):

while True:

try:

return int(Input(str))

except :

print("Что-то пошло не так, повторите пожалуйста")

def Question(question):

print(question)

answer=InputInt(" Да-1, Нет-2")

if(answer==1):

return True

return False

def Add(mainList):

mainList.append(film(\*\*Params()))

def Del(mainList):

Show(mainList)

print("Какой эллемент удалить?")

try:

mainList.remove(mainList[InputInt()])

except IndexError:

print("Нет такого элемента")

def FindYear(mainList):

year = InputInt(" года")

for llist in mainList:

try:

if(llist.releaseDate == year):

llist.printElement()

except AttributeError:

pass

def Show(mainList):

if(mainList):

for llist in mainList:

llist.printElement()

else:

print("Список пуст")

def Params():

print(locals())

if(Question("Ввести код?")):

locals()["code"] = InputInt()

if(Question("Ввести название?")):

locals()["name"] = Input()

if(Question("Ввести режисёра?")):

locals()["director"] = Input()

if(Question("Ввести год выпуска?")):

locals()["releaseDate"] = InputInt()

if(Question("Ввести актёров?")):

locals()["actors"] = Input()

return locals()

def MenuItem(menuItem, mainList):

if(menuItem == "exit"):

raise SystemExit

elif(menuItem == "1"):

Add(mainList)

elif(menuItem == "2"):

Del(mainList)

elif(menuItem == "3"):

Show(mainList)

elif(menuItem == "4"):

FindYear(mainList)

else:

print("Нет такого пукнта")

mainList = []

while True:

PrintMenu()

MenuItem(Input(), mainList)

**4.1.PY**

from PyQt5 import QtWidgets

import UI

import sys

class ExampleApp(QtWidgets.QMainWindow, UI.Ui\_MainWindow):

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

super().\_\_init\_\_()

self.setupUi(self)

def main():

app = QtWidgets.QApplication(sys.argv)

window = ExampleApp()

window.show()

app.exec\_()

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

main()

**UI.PY**

from PyQt5 import QtCore, QtGui, QtWidgets

from PyQt5.QtWidgets import QMessageBox

class Ui\_MainWindow(object):

mainList = []

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(989, 607)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit.setGeometry(QtCore.QRect(50, 30, 171, 31))

self.lineEdit.setObjectName("lineEdit")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(50, 10, 171, 20))

self.label.setAlignment(QtCore.Qt.AlignCenter)

self.label.setObjectName("label")

self.lineEdit\_2 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_2.setGeometry(QtCore.QRect(230, 30, 171, 31))

self.lineEdit\_2.setObjectName("lineEdit\_2")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(230, 10, 171, 20))

self.label\_2.setAlignment(QtCore.Qt.AlignCenter)

self.label\_2.setObjectName("label\_2")

self.label\_3 = QtWidgets.QLabel(self.centralwidget)

self.label\_3.setGeometry(QtCore.QRect(410, 10, 171, 20))

self.label\_3.setAlignment(QtCore.Qt.AlignCenter)

self.label\_3.setObjectName("label\_3")

self.lineEdit\_4 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_4.setGeometry(QtCore.QRect(590, 30, 171, 31))

self.lineEdit\_4.setObjectName("lineEdit\_4")

self.label\_4 = QtWidgets.QLabel(self.centralwidget)

self.label\_4.setGeometry(QtCore.QRect(590, 10, 171, 20))

self.label\_4.setAlignment(QtCore.Qt.AlignCenter)

self.label\_4.setObjectName("label\_4")

self.label\_5 = QtWidgets.QLabel(self.centralwidget)

self.label\_5.setGeometry(QtCore.QRect(770, 10, 211, 20))

self.label\_5.setAlignment(QtCore.Qt.AlignCenter)

self.label\_5.setObjectName("label\_5")

self.pushButtonAddFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonAddFilm.setGeometry(QtCore.QRect(10, 30, 31, 31))

self.pushButtonAddFilm.setObjectName("pushButton\_3")

self.pushButtonAddFilm.clicked.connect(self.OnClickAddFilm)

self.comboBox = QtWidgets.QComboBox(self.centralwidget)

self.comboBox.setGeometry(QtCore.QRect(410, 30, 171, 31))

self.comboBox.setObjectName("comboBox")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox\_2 = QtWidgets.QComboBox(self.centralwidget)

self.comboBox\_2.setGeometry(QtCore.QRect(770, 30, 211, 31))

self.comboBox\_2.setObjectName("comboBox\_2")

self.comboBox\_2.addItem("")

self.comboBox\_2.addItem("")

self.pushButtonDelFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonDelFilm.setGeometry(QtCore.QRect(760, 570, 221, 31))

self.pushButtonDelFilm.setObjectName("pushButton\_4")

self.pushButtonDelFilm.clicked.connect(self.OnClickDelFilm)

self.tableWidget = QtWidgets.QTableWidget(self.centralwidget)

self.tableWidget.setGeometry(QtCore.QRect(10, 70, 971, 491))

self.tableWidget.setObjectName("tableWidget")

self.tableWidget.setColumnCount(5)

self.tableWidget.setRowCount(0)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(0, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(1, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(2, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(3, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(4, item)

self.tableWidget.horizontalHeader().setCascadingSectionResizes(True)

self.tableWidget.horizontalHeader().setDefaultSectionSize(192)

self.tableWidget.horizontalHeader().setSortIndicatorShown(True)

self.tableWidget.horizontalHeader().setStretchLastSection(True)

self.pushButtonShowFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonShowFilm.setGeometry(QtCore.QRect(450, 570, 221, 31))

self.pushButtonShowFilm.setObjectName("pushButton")

self.pushButtonShowFilm.clicked.connect(self.OnClickShowFilm)

self.lineEdit\_3 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_3.setGeometry(QtCore.QRect(10, 570, 221, 31))

self.lineEdit\_3.setObjectName("lineEdit\_3")

self.pushButtonFindYearFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonFindYearFilm.setGeometry(QtCore.QRect(240, 570, 111, 31))

self.pushButtonFindYearFilm.setObjectName("pushButton\_2")

self.pushButtonFindYearFilm.clicked.connect(self.OnClickFindYearFilm)

MainWindow.setCentralWidget(self.centralwidget)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))

self.label.setText(\_translate("MainWindow", "Код"))

self.label\_2.setText(\_translate("MainWindow", "Название"))

self.label\_3.setText(\_translate("MainWindow", "Режисёр"))

self.label\_4.setText(\_translate("MainWindow", "Год"))

self.label\_5.setText(\_translate("MainWindow", "Актёры"))

self.pushButtonAddFilm.setText(\_translate("MainWindow", "+"))

self.comboBox.setItemText(0, \_translate("MainWindow", "Андрей Тарковский"))

self.comboBox.setItemText(1, \_translate("MainWindow", "Кристофер Нолан"))

self.comboBox.setItemText(2, \_translate("MainWindow", "Квентин Тарантино"))

self.comboBox.setItemText(3, \_translate("MainWindow", "Уэс Андерсон"))

self.comboBox.setItemText(4, \_translate("MainWindow", "Дени Вильнёв"))

self.comboBox\_2.setItemText(0, \_translate("MainWindow", "Леонардо ДиКаприо"))

self.comboBox\_2.setItemText(1, \_translate("MainWindow", "Кевин Спейси"))

self.pushButtonDelFilm.setText(\_translate("MainWindow", "Удалить выбранный элемент"))

item = self.tableWidget.horizontalHeaderItem(0)

item.setText(\_translate("MainWindow", "Код"))

item = self.tableWidget.horizontalHeaderItem(1)

item.setText(\_translate("MainWindow", "Название"))

item = self.tableWidget.horizontalHeaderItem(2)

item.setText(\_translate("MainWindow", "Режиссёр"))

item = self.tableWidget.horizontalHeaderItem(3)

item.setText(\_translate("MainWindow", "Год выхода"))

item = self.tableWidget.horizontalHeaderItem(4)

item.setText(\_translate("MainWindow", "Актёры"))

self.pushButtonShowFilm.setText(\_translate("MainWindow", "Отобразить весь список"))

self.pushButtonFindYearFilm.setText(\_translate("MainWindow", "Поиск"))

def OnClickAddFilm(self):

lerr = self.Params()

del lerr['self']

self.mainList.append(film(\*\*lerr))

self.OnClickShowFilm()

def OnClickDelFilm(self):

print(self.tableWidget.currentRow())

try:

self.mainList.pop(self.tableWidget.currentRow())

except IndexError:

QMessageBox.critical(self, "Ошибка ", "Выделенно не то что нужно", QMessageBox.Ok)

return

self.OnClickShowFilm()

def OnClickShowFilm(self):

self.tableWidget.setRowCount(0)

for llist in self.mainList:

self.ShowElementList(llist)

def ShowElementList(self,llist):

try:

str1 = llist.code

except AttributeError:

str1 = "Не указано"

try:

str2 = llist.name

except AttributeError:

str2 = "Не указано"

try:

str3 = llist.director

except AttributeError:

str3 = "Не указано"

try:

str4 = llist.releaseDate

except AttributeError:

str4 = "Не указано"

try:

str5 = llist.actors

except AttributeError:

str5 = "Не указано"

rowPosition = self.tableWidget.rowCount()

self.tableWidget.insertRow(rowPosition)

self.tableWidget.setItem(rowPosition, 0, QtWidgets.QTableWidgetItem(str1))

self.tableWidget.setItem(rowPosition, 1, QtWidgets.QTableWidgetItem(str2))

self.tableWidget.setItem(rowPosition, 2, QtWidgets.QTableWidgetItem(str3))

self.tableWidget.setItem(rowPosition, 3, QtWidgets.QTableWidgetItem(str4))

self.tableWidget.setItem(rowPosition, 4, QtWidgets.QTableWidgetItem(str5))

def OnClickFindYearFilm(self):

self.tableWidget.setRowCount(0)

year = self.lineEdit\_3.text()

for llist in self.mainList:

try:

if(llist.releaseDate == year):

self.ShowElementList(llist)

except AttributeError:

pass

def Params(self):

print(locals())

code = self.lineEdit.text()

if(code==""):

locals()["code"] = code

name = self.lineEdit\_2.text()

if(name==""):

locals()["name"] = name

director = self.comboBox.currentText()

if(director == ""):

locals()["director"] = director

releaseDate = self.lineEdit\_4.text()

if(releaseDate == ""):

locals()["releaseDate"] = releaseDate

actors = self.comboBox\_2.currentText()

if(actors == ""):

locals()["actors"] = actors

print(locals())

return locals()

class film:

def \_\_init\_\_(self, \*\*kwargs):

for key in kwargs:

setattr(self, key, kwargs[key])

**5.1.PY**

from PyQt5 import QtWidgets

import UI

import sys

class ExampleApp(QtWidgets.QMainWindow, UI.Ui\_MainWindow):

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

super().\_\_init\_\_()

self.setupUi(self)

def main():

app = QtWidgets.QApplication(sys.argv)

window = ExampleApp()

window.show()

app.exec\_()

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

main()

**UI.PY**

from PyQt5 import QtCore, QtGui, QtWidgets

from PyQt5.QtWidgets import QMessageBox

from PyQt5.QtWidgets import QFileDialog

class Ui\_MainWindow(object):

mainList = []

fileName = ""

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(989, 635)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.lineEdit = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit.setGeometry(QtCore.QRect(50, 30, 171, 31))

self.lineEdit.setObjectName("lineEdit")

self.label = QtWidgets.QLabel(self.centralwidget)

self.label.setGeometry(QtCore.QRect(50, 10, 171, 20))

self.label.setAlignment(QtCore.Qt.AlignCenter)

self.label.setObjectName("label")

self.lineEdit\_2 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_2.setGeometry(QtCore.QRect(230, 30, 171, 31))

self.lineEdit\_2.setObjectName("lineEdit\_2")

self.label\_2 = QtWidgets.QLabel(self.centralwidget)

self.label\_2.setGeometry(QtCore.QRect(230, 10, 171, 20))

self.label\_2.setAlignment(QtCore.Qt.AlignCenter)

self.label\_2.setObjectName("label\_2")

self.label\_3 = QtWidgets.QLabel(self.centralwidget)

self.label\_3.setGeometry(QtCore.QRect(410, 10, 171, 20))

self.label\_3.setAlignment(QtCore.Qt.AlignCenter)

self.label\_3.setObjectName("label\_3")

self.lineEdit\_4 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_4.setGeometry(QtCore.QRect(590, 30, 171, 31))

self.lineEdit\_4.setObjectName("lineEdit\_4")

self.label\_4 = QtWidgets.QLabel(self.centralwidget)

self.label\_4.setGeometry(QtCore.QRect(590, 10, 171, 20))

self.label\_4.setAlignment(QtCore.Qt.AlignCenter)

self.label\_4.setObjectName("label\_4")

self.label\_5 = QtWidgets.QLabel(self.centralwidget)

self.label\_5.setGeometry(QtCore.QRect(770, 10, 211, 20))

self.label\_5.setAlignment(QtCore.Qt.AlignCenter)

self.label\_5.setObjectName("label\_5")

self.pushButtonAddFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonAddFilm.setGeometry(QtCore.QRect(10, 30, 31, 31))

self.pushButtonAddFilm.setObjectName("pushButton\_3")

self.pushButtonAddFilm.clicked.connect(self.OnClickAddFilm)

self.comboBox = QtWidgets.QComboBox(self.centralwidget)

self.comboBox.setGeometry(QtCore.QRect(410, 30, 171, 31))

self.comboBox.setObjectName("comboBox")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox.addItem("")

self.comboBox\_2 = QtWidgets.QComboBox(self.centralwidget)

self.comboBox\_2.setGeometry(QtCore.QRect(770, 30, 211, 31))

self.comboBox\_2.setObjectName("comboBox\_2")

self.comboBox\_2.addItem("")

self.comboBox\_2.addItem("")

self.pushButtonDelFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonDelFilm.setGeometry(QtCore.QRect(760, 570, 221, 31))

self.pushButtonDelFilm.setObjectName("pushButton\_4")

self.pushButtonDelFilm.clicked.connect(self.OnClickDelFilm)

self.tableWidget = QtWidgets.QTableWidget(self.centralwidget)

self.tableWidget.setGeometry(QtCore.QRect(10, 70, 971, 491))

self.tableWidget.setObjectName("tableWidget")

self.tableWidget.setColumnCount(5)

self.tableWidget.setRowCount(0)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(0, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(1, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(2, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(3, item)

item = QtWidgets.QTableWidgetItem()

self.tableWidget.setHorizontalHeaderItem(4, item)

self.tableWidget.horizontalHeader().setCascadingSectionResizes(True)

self.tableWidget.horizontalHeader().setDefaultSectionSize(192)

self.tableWidget.horizontalHeader().setSortIndicatorShown(True)

self.tableWidget.horizontalHeader().setStretchLastSection(True)

self.pushButtonShowFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonShowFilm.setGeometry(QtCore.QRect(450, 570, 221, 31))

self.pushButtonShowFilm.setObjectName("pushButton")

self.pushButtonShowFilm.clicked.connect(self.OnClickShowFilm)

self.lineEdit\_3 = QtWidgets.QLineEdit(self.centralwidget)

self.lineEdit\_3.setGeometry(QtCore.QRect(10, 570, 221, 31))

self.lineEdit\_3.setObjectName("lineEdit\_3")

self.pushButtonFindYearFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonFindYearFilm.setGeometry(QtCore.QRect(240, 570, 111, 31))

self.pushButtonFindYearFilm.setObjectName("pushButton\_2")

self.pushButtonFindYearFilm.clicked.connect(self.OnClickFindYearFilm)

self.menuBar = QtWidgets.QMenuBar(MainWindow)

self.menuBar.setGeometry(QtCore.QRect(0, 0, 989, 22))

self.menuBar.setObjectName("menuBar")

self.menu = QtWidgets.QMenu(self.menuBar)

self.menu.setObjectName("menu")

self.menu\_2 = QtWidgets.QMenu(self.menuBar)

self.menu\_2.setObjectName("menu\_2")

MainWindow.setMenuBar(self.menuBar)

self.action = QtWidgets.QAction(MainWindow)

self.action.setObjectName("action")

self.action.triggered.connect(self.Create)

self.action\_2 = QtWidgets.QAction(MainWindow)

self.action\_2.setObjectName("action\_2")

self.action\_2.triggered.connect(self.Open)

self.action\_3 = QtWidgets.QAction(MainWindow)

self.action\_3.setObjectName("action\_3")

self.action\_3.triggered.connect(self.Save)

self.action\_4 = QtWidgets.QAction(MainWindow)

self.action\_4.setObjectName("action\_4")

self.action\_4.triggered.connect(self.SaveAs)

self.action\_5 = QtWidgets.QAction(MainWindow)

self.action\_5.setObjectName("action\_5")

self.action\_5.triggered.connect(self.Exit)

self.action\_6 = QtWidgets.QAction(MainWindow)

self.action\_6.setObjectName("action\_6")

self.action\_6.triggered.connect(self.Author)

self.menu.addAction(self.action)

self.menu.addAction(self.action\_2)

self.menu.addAction(self.action\_3)

self.menu.addAction(self.action\_4)

self.menu.addAction(self.action\_5)

self.menu\_2.addAction(self.action\_6)

self.menuBar.addAction(self.menu.menuAction())

self.menuBar.addAction(self.menu\_2.menuAction())

MainWindow.setCentralWidget(self.centralwidget)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "MainWindow"))

self.label.setText(\_translate("MainWindow", "Код"))

self.label\_2.setText(\_translate("MainWindow", "Название"))

self.label\_3.setText(\_translate("MainWindow", "Режисёр"))

self.label\_4.setText(\_translate("MainWindow", "Год"))

self.label\_5.setText(\_translate("MainWindow", "Актёры"))

self.pushButtonAddFilm.setText(\_translate("MainWindow", "+"))

self.comboBox.setItemText(0, \_translate("MainWindow", "Андрей Тарковский"))

self.comboBox.setItemText(1, \_translate("MainWindow", "Кристофер Нолан"))

self.comboBox.setItemText(2, \_translate("MainWindow", "Квентин Тарантино"))

self.comboBox.setItemText(3, \_translate("MainWindow", "Уэс Андерсон"))

self.comboBox.setItemText(4, \_translate("MainWindow", "Дени Вильнёв"))

self.comboBox\_2.setItemText(0, \_translate("MainWindow", "Леонардо ДиКаприо"))

self.comboBox\_2.setItemText(1, \_translate("MainWindow", "Кевин Спейси"))

self.pushButtonDelFilm.setText(\_translate("MainWindow", "Удалить выбранный элемент"))

item = self.tableWidget.horizontalHeaderItem(0)

item.setText(\_translate("MainWindow", "Код"))

item = self.tableWidget.horizontalHeaderItem(1)

item.setText(\_translate("MainWindow", "Название"))

item = self.tableWidget.horizontalHeaderItem(2)

item.setText(\_translate("MainWindow", "Режиссёр"))

item = self.tableWidget.horizontalHeaderItem(3)

item.setText(\_translate("MainWindow", "Год выхода"))

item = self.tableWidget.horizontalHeaderItem(4)

item.setText(\_translate("MainWindow", "Актёры"))

self.pushButtonShowFilm.setText(\_translate("MainWindow", "Отобразить весь список"))

self.pushButtonFindYearFilm.setText(\_translate("MainWindow", "Поиск"))

self.menu.setTitle(\_translate("MainWindow", "Файл"))

self.menu\_2.setTitle(\_translate("MainWindow", "Справка"))

self.action.setText(\_translate("MainWindow", "Создать"))

self.action\_2.setText(\_translate("MainWindow", "Открыть"))

self.action\_3.setText(\_translate("MainWindow", "Сохранить"))

self.action\_4.setText(\_translate("MainWindow", "Сохранить как..."))

self.action\_5.setText(\_translate("MainWindow", "Выход"))

self.action\_6.setText(\_translate("MainWindow", "Автор"))

def Create(self):

self.tableWidget.setRowCount(0)

self.lineEdit.clear()

self.lineEdit\_2.clear()

self.lineEdit\_3.clear()

self.lineEdit\_4.clear()

self.mainList.clear()

def Open(self):

self.fileName = QFileDialog.getOpenFileName(self,"Открыть файл", "", "(\*.films)")

self.Create()

with open(self.fileName[0], mode="r",encoding='utf-8') as file:

listData = file.readlines()

print(listData)

for llist in listData:

if(len(llist.split('$'))>1):

lerr = self.ParamsForOpen(llist.split('$'))

del lerr['self']

del lerr['llist']

self.mainList.append(film(\*\*lerr))

def Save(self):

if(self.fileName!=""):

self.SaveAll()

else:

self.SaveAs()

def SaveAs(self):

self.fileName = QFileDialog.getSaveFileName(self, "Save File","","(\*.films)");

if(self.fileName[0]!=""):

self.SaveAll()

def SaveAll(self):

with open(self.fileName[0], "w",encoding='utf-8') as file:

for llist in self.mainList:

try:

file.write(llist.code+"$")

except AttributeError:

file.write("Не указано"+"$")

try:

file.write(llist.name+"$")

except AttributeError:

file.write("Не указано"+"$")

try:

file.write(llist.director+"$")

except AttributeError:

file.write("Не указано"+"$")

try:

file.write(llist.releaseDate+"$")

except AttributeError:

file.write("Не указано"+"$")

try:

file.write(llist.actors)

except AttributeError:

file.write("Не указано")

file.write("\n")

def Exit(self):

raise SystemExit

def Author(self):

QMessageBox.about(self,"Автор","Музалевский Никита")

def OnClickAddFilm(self):

lerr = self.Params()

del lerr['self']

self.mainList.append(film(\*\*lerr))

self.OnClickShowFilm()

def OnClickDelFilm(self):

print(self.tableWidget.currentRow())

try:

self.mainList.pop(self.tableWidget.currentRow())

except IndexError:

QMessageBox.critical(self, "Ошибка ", "Выделенно не то что нужно", QMessageBox.Ok)

return

self.OnClickShowFilm()

def OnClickShowFilm(self):

self.tableWidget.setRowCount(0)

for llist in self.mainList:

self.ShowElementList(llist)

def ShowElementList(self,llist):

try:

str1 = llist.code

except AttributeError:

str1 = "Не указано"

try:

str2 = llist.name

except AttributeError:

str2 = "Не указано"

try:

str3 = llist.director

except AttributeError:

str3 = "Не указано"

try:

str4 = llist.releaseDate

except AttributeError:

str4 = "Не указано"

try:

str5 = llist.actors

except AttributeError:

str5 = "Не указано"

rowPosition = self.tableWidget.rowCount()

self.tableWidget.insertRow(rowPosition)

self.tableWidget.setItem(rowPosition, 0, QtWidgets.QTableWidgetItem(str1))

self.tableWidget.setItem(rowPosition, 1, QtWidgets.QTableWidgetItem(str2))

self.tableWidget.setItem(rowPosition, 2, QtWidgets.QTableWidgetItem(str3))

self.tableWidget.setItem(rowPosition, 3, QtWidgets.QTableWidgetItem(str4))

self.tableWidget.setItem(rowPosition, 4, QtWidgets.QTableWidgetItem(str5))

def OnClickFindYearFilm(self):

self.tableWidget.setRowCount(0)

year = self.lineEdit\_3.text()

for llist in self.mainList:

try:

if(llist.releaseDate == year):

self.ShowElementList(llist)

except AttributeError:

pass

def Params(self):

print(locals())

code = self.lineEdit.text()

if(code==""):

locals()["code"] = code

name = self.lineEdit\_2.text()

if(name==""):

locals()["name"] = name

director = self.comboBox.currentText()

if(director == ""):

locals()["director"] = director

releaseDate = self.lineEdit\_4.text()

if(releaseDate == ""):

locals()["releaseDate"] = releaseDate

actors = self.comboBox\_2.currentText()

if(actors == ""):

locals()["actors"] = actors

print(locals())

return locals()

def ParamsForOpen(self,llist):

code = llist[0]

if(code==""):

locals()["code"] = code

name = llist[1]

if(name==""):

locals()["name"] = name

director = llist[2]

if(director == ""):

locals()["director"] = director

releaseDate = llist[3]

if(releaseDate == ""):

locals()["releaseDate"] = releaseDate

actors = llist[4]

if(actors == ""):

locals()["actors"] = actors

return locals()

class film:

def \_\_init\_\_(self, \*\*kwargs):

for key in kwargs:

setattr(self, key, kwargs[key])

**CLIENT.PY**

import sys

from PyQt5 import QtWidgets

import UI

class ExampleApp(QtWidgets.QMainWindow, UI.Ui\_MainWindow):

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

super().\_\_init\_\_()

self.setupUi(self)

def function(self):

mess=self.TextChange()

message = mess.encode()

sock.sendall(message)

def main():

app = QtWidgets.QApplication(sys.argv)

window = ExampleApp()

window.show()

app.exec\_()

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

main()

**SERVER.PY**

import socket

import sys

morseDict = {

"А":"\*-" ,"Б":"-\*\*\*" ,

"В":"-\*\*" ,"Г":"--\*" ,

"Д":"-\*\*" ,"Е":"\*" ,

"Ё":"\*" ,"Ж":"\*\*\*-" ,

"З":"--\*\*" ,"И":"\*\*" ,

"Й":"\*---" ,"К":"-\*-" ,

"Л":"\*-\*\*" ,"М":"--" ,

"Н":"-\*" ,"О":"---" ,

"П":"\*--\*" ,"Р":"\*-\*" ,

"С":"\*\*\*" ,"Т":"−" ,

"У":"\*\*-" ,"Ф":"\*\*-\*" ,

"Х":"\*\*\*\*" ,"Ц":"-\*-\*" ,

"Ч":"---\*" ,"Ш":"----" ,

"Щ":"--\*-" ,"Ъ":"--\*--" ,

"Ы":"-\*--" ,"Ь":"-\*\*-" ,

"Э":"\*\*-\*\*","Ю":"\*\*--" ,

"Я":"\*-\*-" ,"!":"--\*\*--" ,

"A":"\*-" , "B":"-\*" ,

"C":"-\*-\*" , "D":"-\*" ,

"E":"\*" , "F":"\*-\*" ,

"G":"--\*" , "H":"\*" ,

"I":"\*" , "J":"\*---" ,

"K":"-\*-" , "L":"\*-\*" ,

"M":"--" , "N":"-\*" ,

"O":"---" , "P":"\*--\*" ,

"Q":"--\*-" , "R":"\*-\*" ,

"S":"\*" , "T":"-" ,

"U":"\*-" , "V":"\*-" ,

"W":"\*--" , "X":"-\*-" ,

"Y":"-\*--" , "Z":"--\*" ,

"1":"\*----", "2":"\*---" ,

"3":"\*--" , "4":"\*-" ,

"5":"\*" , "6":"-\*" ,

"7":"--\*" , "8":"---\*" ,

"9":"----\*", "0":"-----" ,

",":"--\*--", "\*":"\*-\*-\*-",

"?":"\*--\*" , "/":"-\*-\*" ,

"-":"-\*-" , "(":"-\*--\*" ,

")":"-\*--\*-"

}

def Normal(data):

str1 = ""

data=data.upper()

for letter in data:

try:

str1+=morseDict[letter]

except KeyError:

str1+=str(letter)

return(str1)

def Morze(data):

str1 = ""

data=data.upper()

for letter in data:

try:

if(letter=="\*"):

str1+="Е"

elif(letter=="-"):

str1+="Т"

else:

str1+=str(letter)

except KeyError:

str1+=str(letter)

return(str1)

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

serverAddress = ('localhost', 8080)

sock.bind(serverAddress)

sock.listen(1)

while True:

print('Ожидание соединения...')

connection, client\_address = sock.accept()

try:

print('Подключён:', client\_address)

while True:

data = connection.recv(512)

data = data.decode('utf-8')

print(data)

if data:

if(data[0]=="1"):

data = data[1 :]

data = Normal(data)

elif(data[0]=="2"):

data = data[1 :]

data = Morze(data)

else:

break

connection.sendall(data.encode())

else:

print('Нет данных от:', client\_address)

break

finally:

connection.close()

**UI.PY**

from PyQt5 import QtCore, QtGui, QtWidgets

import socket

sock = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

server\_address = ('localhost', 8080)

print('Подключено к {} порт {}'.format(\*server\_address))

sock.connect(server\_address)

class Ui\_MainWindow(object):

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(570, 478)

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.textEdit1 = QtWidgets.QTextEdit(self.centralwidget)

self.textEdit1.setGeometry(QtCore.QRect(20, 20, 541, 211))

self.textEdit1.setObjectName("textEdit1")

self.textEdit2 = QtWidgets.QTextEdit(self.centralwidget)

self.textEdit2.setEnabled(False)

self.textEdit2.setGeometry(QtCore.QRect(20, 250, 541, 211))

self.textEdit2.setObjectName("textEdit2")

self.label1 = QtWidgets.QLabel(self.centralwidget)

self.label1.setGeometry(QtCore.QRect(236, 0, 100, 20))

font = QtGui.QFont()

font.setPointSize(12)

self.label1.setFont(font)

self.label1.setObjectName("label1")

self.label2 = QtWidgets.QLabel(self.centralwidget)

self.label2.setGeometry(QtCore.QRect(260, 230, 51, 20))

font = QtGui.QFont()

font.setPointSize(12)

self.label2.setFont(font)

self.label2.setObjectName("label2")

self.radioButton1 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton1.setGeometry(QtCore.QRect(5, 110, 21, 20))

self.radioButton1.setText("")

self.radioButton1.setChecked(True)

self.radioButton1.setObjectName("radioButton1")

self.radioButton2 = QtWidgets.QRadioButton(self.centralwidget)

self.radioButton2.setGeometry(QtCore.QRect(5, 350, 21, 20))

self.radioButton2.setText("")

self.radioButton2.setObjectName("radioButton2")

self.radioButton1.clicked.connect(self.RadioButton1)

self.radioButton2.clicked.connect(self.RadioButton2)

MainWindow.setCentralWidget(self.centralwidget)

self.textEdit1.textChanged.connect(self.TextChangeNormal)

self.textEdit2.textChanged.connect(self.TextChangeMorze)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "Шифратор"))

self.label1.setText(\_translate("MainWindow", "Естественный"))

self.label2.setText(\_translate("MainWindow", "Морзе"))

def RadioButton1(self):

\_translate = QtCore.QCoreApplication.translate

self.textEdit1.setEnabled(True)

self.textEdit2.setEnabled(False)

self.setWindowTitle(\_translate("MainWindow", "Шифратор"))

self.TextChangeNormal()

def RadioButton2(self):

\_translate = QtCore.QCoreApplication.translate

self.textEdit1.setEnabled(False)

self.textEdit2.setEnabled(True)

self.setWindowTitle(\_translate("MainWindow", "Дешифратор"))

self.TextChangeMorze()

def TextChangeNormal(self):

mess = self.textEdit1.toPlainText()

if(mess):

if(self.textEdit1.isEnabled()):

message = ("1"+mess).encode()

sock.sendall(message)

data = sock.recv(512)

mess = data.decode()

self.textEdit2.setText(mess)

elif(self.textEdit2.toPlainText()):

self.textEdit2.setText("")

def TextChangeMorze(self):

mess = self.textEdit2.toPlainText()

if(mess):

if(self.textEdit2.isEnabled()):

message = ("2"+mess).encode()

sock.sendall(message)

data = sock.recv(512)

mess = data.decode()

self.textEdit1.setText(mess)

elif(self.textEdit1.toPlainText()):

self.textEdit1.setText("")

**7.1.PY**

from PyQt5 import QtWidgets

import UI

import sys

class ExampleApp(QtWidgets.QMainWindow, UI.Ui\_MainWindow):

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

super().\_\_init\_\_()

self.setupUi(self)

def main():

app = QtWidgets.QApplication(sys.argv)

window = ExampleApp()

window.show()

app.exec\_()

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

main()

**DB.SQL**

CREATE SCHEMA films;

use films;

CREATE TABLE country (

country\_id SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

name VARCHAR(45) NOT NULL,

PRIMARY KEY (country\_id));

CREATE TABLE director (

director\_id MEDIUMINT UNSIGNED NOT NULL AUTO\_INCREMENT,

firstname VARCHAR(45) NOT NULL,

lastname VARCHAR(45) NOT NULL,

country\_id SMALLINT UNSIGNED NOT NULL,

PRIMARY KEY (director\_id),

FOREIGN KEY (country\_id) REFERENCES country (country\_id));

CREATE TABLE film (

film\_id SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

name VARCHAR(45) NOT NULL,

release\_date DATE,

director\_id MEDIUMINT UNSIGNED NOT NULL,

PRIMARY KEY (film\_id),

FOREIGN KEY (director\_id) REFERENCES director (director\_id));

CREATE TABLE film\_country (

film\_id SMALLINT UNSIGNED NOT NULL,

country\_id SMALLINT UNSIGNED NOT NULL,

PRIMARY KEY (film\_id, country\_id),

FOREIGN KEY (film\_id) REFERENCES film (film\_id),

FOREIGN KEY (country\_id) REFERENCES country (country\_id));

INSERT INTO country(name)

VALUES ('Россия'),('США'),('Великобритания'),('Канада');

SELECT \* FROM country;

INSERT INTO director(firstname,lastname,country\_id)

VALUES ('Андрей','Тарковский',1),('Кристофер','Нолан',3),('Квентин','Тарантино',2),('Уэс','Андерсон',2),('Дени','Вильнёв',4);

**UI.PY**

from PyQt5 import QtCore, QtGui, QtWidgets

import pymysql

con = pymysql.connect(host='localhost',

user='root',

password='1234',

database='films',

charset='utf8mb4')

countryDict = {"Россия":"1", "США":"2", "Великобритания":"3", "Канада":"4"}

cur = con.cursor()

countries=list()

class Ui\_MainWindow(object):

id\_film = 1

def setupUi(self, MainWindow):

MainWindow.setObjectName("MainWindow")

MainWindow.resize(230, 505)

MainWindow.setMaximumSize(QtCore.QSize(16777210, 16777215))

self.centralwidget = QtWidgets.QWidget(MainWindow)

self.centralwidget.setObjectName("centralwidget")

self.groupBoxAddFil = QtWidgets.QGroupBox(self.centralwidget)

self.groupBoxAddFil.setGeometry(QtCore.QRect(10, 220, 211, 280))

self.groupBoxAddFil.setTitle("")

self.groupBoxAddFil.setObjectName("groupBoxAddFil")

self.textEditFilms = QtWidgets.QTextEdit(self.centralwidget)

self.textEditFilms.setGeometry(QtCore.QRect(10, 10, 211, 160))

self.textEditFilms.setObjectName("textEditFilms")

self.textEditFilms.setLineWrapMode(QtWidgets.QTextEdit.NoWrap)

self.lineEditName = QtWidgets.QLineEdit(self.groupBoxAddFil)

self.lineEditName.setGeometry(QtCore.QRect(0, 20, 211, 31))

self.lineEditName.setObjectName("lineEditName")

self.dateEdit = QtWidgets.QDateEdit(self.groupBoxAddFil)

self.dateEdit.setGeometry(QtCore.QRect(0, 70, 211, 31))

self.dateEdit.setObjectName("dateEdit")

self.pushButtonAddCountry = QtWidgets.QPushButton(self.groupBoxAddFil)

self.pushButtonAddCountry.setGeometry(QtCore.QRect(0, 170, 31, 31))

self.pushButtonAddCountry.setObjectName("pushButtonAddCountry")

self.pushButtonAddCountry.clicked.connect(self.OnClickAddCountry)

self.lineEditCountry = QtWidgets.QLineEdit(self.groupBoxAddFil)

self.lineEditCountry.setGeometry(QtCore.QRect(0, 210, 211, 21))

self.lineEditCountry.setObjectName("lineEditCountry")

self.lineEditCountry.setEnabled(False)

self.pushButtonDelCountry = QtWidgets.QPushButton(self.groupBoxAddFil)

self.pushButtonDelCountry.setGeometry(QtCore.QRect(180, 170, 31, 31))

self.pushButtonDelCountry.setObjectName("pushButtonDelCountry")

self.pushButtonDelCountry.clicked.connect(self.OnClickDelCountry)

self.label = QtWidgets.QLabel(self.groupBoxAddFil)

self.label.setGeometry(QtCore.QRect(0, 0, 211, 21))

self.label.setAlignment(QtCore.Qt.AlignCenter)

self.label.setObjectName("label")

self.label\_2 = QtWidgets.QLabel(self.groupBoxAddFil)

self.label\_2.setGeometry(QtCore.QRect(0, 50, 211, 21))

self.label\_2.setAlignment(QtCore.Qt.AlignCenter)

self.label\_2.setObjectName("label\_2")

self.pushButtonAddFilm = QtWidgets.QPushButton(self.groupBoxAddFil)

self.pushButtonAddFilm.setGeometry(QtCore.QRect(40, 240, 131, 31))

self.pushButtonAddFilm.setObjectName("pushButtonAddFilm")

self.label\_4 = QtWidgets.QLabel(self.groupBoxAddFil)

self.label\_4.setGeometry(QtCore.QRect(0, 100, 211, 21))

self.label\_4.setAlignment(QtCore.Qt.AlignCenter)

self.label\_4.setObjectName("label\_4")

self.comboBoxDirector = QtWidgets.QComboBox(self.groupBoxAddFil)

self.comboBoxDirector.setGeometry(QtCore.QRect(0, 120, 211, 31))

self.comboBoxDirector.setObjectName("comboBoxDirector")

cur.execute("SELECT firstname,lastname FROM director")

rows = cur.fetchall()

for row in rows:

self.comboBoxDirector.addItems([row[0]+" "+row[1]])

self.comboBoxCountry = QtWidgets.QComboBox(self.groupBoxAddFil)

self.comboBoxCountry.setGeometry(QtCore.QRect(40, 170, 131, 31))

self.comboBoxCountry.setObjectName("comboBoxCountry")

cur.execute("SELECT name FROM country")

rows = cur.fetchall()

for row in rows:

self.comboBoxCountry.addItems([row[0]])

self.label\_3 = QtWidgets.QLabel(self.groupBoxAddFil)

self.label\_3.setGeometry(QtCore.QRect(0, 150, 211, 21))

self.label\_3.setAlignment(QtCore.Qt.AlignCenter)

self.label\_3.setObjectName("label\_3")

self.comboBoxFilm = QtWidgets.QComboBox(self.centralwidget)

self.comboBoxFilm.setGeometry(QtCore.QRect(10, 180, 170, 31))

self.comboBoxFilm.setObjectName("comboBoxFilm")

self.pushButtonDelFilm = QtWidgets.QPushButton(self.centralwidget)

self.pushButtonDelFilm.setGeometry(QtCore.QRect(190, 180, 31, 31))

self.pushButtonDelFilm.setObjectName("pushButtonDelFilm")

self.pushButtonDelFilm.clicked.connect(self.OnClickDelFilm)

self.pushButtonAddFilm.clicked.connect(self.OnClickAddFilm)

MainWindow.setCentralWidget(self.centralwidget)

self.retranslateUi(MainWindow)

QtCore.QMetaObject.connectSlotsByName(MainWindow)

def retranslateUi(self, MainWindow):

\_translate = QtCore.QCoreApplication.translate

MainWindow.setWindowTitle(\_translate("MainWindow", "База данных фильмов"))

self.pushButtonAddCountry.setText(\_translate("MainWindow", "+"))

self.pushButtonDelCountry.setText(\_translate("MainWindow", "-"))

self.label.setText(\_translate("MainWindow", "Название"))

self.label\_2.setText(\_translate("MainWindow", "Дата выхода"))

self.pushButtonAddFilm.setText(\_translate("MainWindow", "Дабавить"))

self.label\_4.setText(\_translate("MainWindow", "Режиссёр"))

self.label\_3.setText(\_translate("MainWindow", "Страны"))

self.pushButtonDelFilm.setText(\_translate("MainWindow", "-"))

self.ShowFilms()

def OnClickAddFilm(self):

cur.execute("INSERT INTO film(name,release\_date,director\_id) VALUES ('"+self.lineEditName.text()+"','"+self.dateEdit.dateTime().toString('yyyy-MM-dd') +"','"+str(self.comboBoxDirector.currentIndex()+1)+"')");

cur.execute("SELECT film\_id FROM film ORDER BY film\_id DESC LIMIT 1")

rows = cur.fetchall()

print(rows)

for country in countries:

cur.execute("INSERT INTO film\_country(film\_id,country\_id) VALUES ('"+str(rows[0][0])+"','"+ countryDict[country] +"')");

self.ShowFilms()

con.commit()

def ShowFilms(self):

cur.execute("SELECT film\_id,f.name,release\_date,firstname,lastname,c.name FROM film f JOIN director d USING(director\_id) JOIN country c USING(country\_id)")

rows = cur.fetchall()

self.textEditFilms.setText("")

for row in rows:

cur.execute("SELECT c.name FROM film\_country fc JOIN film f USING(film\_id) JOIN country c USING(country\_id) WHERE f.film\_id="+str(row[0]))

rows1 = cur.fetchall()

str1=""

for row1 in rows1:

for r in row1:

str1+=str(r)+";"

self.textEditFilms.setText(self.textEditFilms.toPlainText()+"Название: "+row[1]+"; Дата выхода: "+str(row[2])+"; Режиссёр: "+ row[3]+" "+row[4] +" из " + row[5] + "; Страны которые участвоаали в создании: " +str1+"\n")

print("Название: "+row[1]+"; Дата выхода: "+str(row[2])+"; Режиссёр: "+ row[3]+" "+row[4] +" из " + row[5] + "; Страны которые участвоаали в создании: " +str1)

cur.execute("SELECT name FROM film")

rows = cur.fetchall()

self.comboBoxFilm.clear()

for row in rows:

self.comboBoxFilm.addItems([row[0]])

self.lineEditCountry.setText("")

self.lineEditName.setText("")

countries.clear()

con.commit()

def OnClickDelFilm(self):

print("DELETE FROM film\_country WHERE film\_id=(SELECT film\_id FROM film WHERE name='"+self.comboBoxFilm.currentText()+"' LIMIT 1);")

cur.execute("DELETE FROM film\_country WHERE film\_id=(SELECT film\_id FROM film WHERE name='"+self.comboBoxFilm.currentText()+"' LIMIT 1);")

cur.execute("DELETE FROM film WHERE name='"+self.comboBoxFilm.currentText()+"';")

self.ShowFilms()

con.commit()

def OnClickAddCountry(self):

textInComboBoxCountry = self.comboBoxCountry.currentText()

if(countries):

for country in countries:

if(textInComboBoxCountry==country):

return

countries.append(textInComboBoxCountry)

self.lineEditCountry.setText(self.lineEditCountry.text()+textInComboBoxCountry+";")

else:

countries.append(textInComboBoxCountry)

self.lineEditCountry.setText(self.lineEditCountry.text()+textInComboBoxCountry+";")

def OnClickDelCountry(self):

textInComboBoxCountry = self.comboBoxCountry.currentText()

self.lineEditCountry.setText("")

for country in countries:

if(textInComboBoxCountry==country):

countries.remove(country)

for country in countries:

self.lineEditCountry.setText(self.lineEditCountry.text()+country+";")

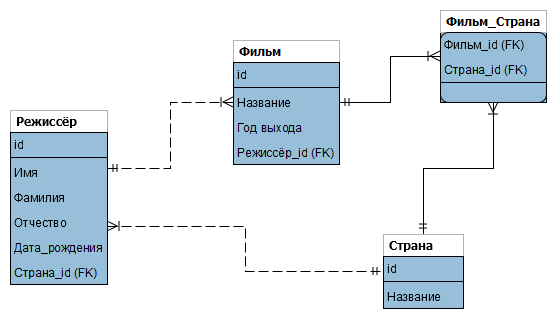


Рисунок 1 – LScheme

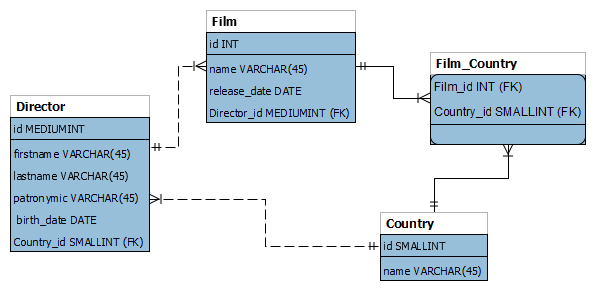


Рисунок 2 – PScheme