# ***Приложение к руководству разработчика***

## ***Листинг файла display.py***

#!/usr/bin/python

# -\*- coding: utf-8 -\*-

def display():

"""

Автор: Волков В.Д.

Отображает основное окно

"""

# first part

import tkinter as tk

class scrollFrame(tk.Frame):

def \_\_init\_\_(self, parent, \*args, \*\*kw):

"""

Автор: Труханов А.И.

Объявление переменных в классе scrollFrame

"""

tk.Frame.\_\_init\_\_(self, parent, \*args, \*\*kw)

# create a canvas object and a vertical scrollbar for scrolling it

vscrollbar = tk.Scrollbar(self, orient=tk.VERTICAL)

vscrollbar.pack(fill=tk.Y, side=tk.RIGHT, expand=tk.FALSE)

canvas = tk.Canvas(self, bd=0, highlightthickness=0,

yscrollcommand=vscrollbar.set)

canvas.pack(side=tk.LEFT, fill=tk.BOTH, expand=tk.TRUE)

vscrollbar.config(command=canvas.yview)

# reset the view

canvas.xview\_moveto(0)

canvas.yview\_moveto(0)

# create a frame inside the canvas which will be scrolled with it

self.interior = interior = tk.Frame(canvas)

interior\_id = canvas.create\_window(0, 0, window=interior,

anchor=tk.NW)

# track changes to the canvas and frame width and sync them,

# also updating the scrollbar

def \_configure\_interior(event):

"""

Автор: Гуняшов Н.Н.

Настройки прокрутки

"""

# update the scrollbars to match the size of the inner frame

size = (interior.winfo\_reqwidth(), interior.winfo\_reqheight())

canvas.config(scrollregion="0 0 %s %s" % size)

if interior.winfo\_reqwidth() != canvas.winfo\_width():

# update the canvas's width to fit the inner frame

canvas.config(width=interior.winfo\_reqwidth())

interior.bind('<Configure>', \_configure\_interior)

def \_configure\_canvas(event):

"""

Автор: Волков В.Д.

Настройки прокручиваемого поля

"""

if interior.winfo\_reqwidth() != canvas.winfo\_width():

# update the inner frame's width to fill the canvas

canvas.itemconfigure(interior\_id, width=canvas.winfo\_width())

canvas.bind('<Configure>', \_configure\_canvas)

class MainWindow:

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

"""

Автор: Труханов А.И.

Объявление переменных в классе MainWindow

"""

self.flagSort = 0

self.currSort = ""

self.base = base#main.readData()

self.p = []

self.im = tk.Label(root)

self.im.place(x=0, y=0, relwidth=1, relheight=1)

self.frame\_all = tk.Frame(self.im)

self.scrollF = scrollFrame(self.frame\_all)

self.frame\_sort = tk.Frame(self.frame\_all)

self.frame\_add = tk.Frame(self.frame\_all)

self.frame\_search = tk.Frame(self.frame\_all)

self.frame\_exit = tk.Frame(self.frame\_all)

self.sequence = [0,2,1,3,5,6,4]

self.width = [12,8,9,9,2,16,16]

class skip:

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

"""

Автор: Гуняшов Н.Н.

Пролистывание названия в кнопке

"""

self.s = name

self.i1 = -1

self.j = 9

self.focus = self.s[self.i1:self.j]

self.flag = 0

self.flagButScroll = 0

def cancelSkip(self):

"""

Автор: Волков В.Д.

Отмена прокрутки

"""

self.flagButScroll = 1

def scroll(self,but):

"""

Автор: Труханов А.И.

Прокрутка

"""

self.flagButScroll = 0

self.i1 = -1

self.j = 9

def change():

"""

Автор: Гуняшов Н.Н.

Меняет фокус на буквы

"""

if ( self.j < len(self.s) or self.i1 == -1):

self.i1 = self.i1 + 1

self.j = self.j + 1

self.focus = self.s[self.i1:self.j]

change()

but["text"] = self.focus

def flagPlus():

"""

Автор: Волков В.Д.

Прибавляет флаг

"""

self.flag = self.flag + 1

self.flag = self.j

while (self.flag < len(self.s) ):

but["text"] = self.focus

flagPlus()

if (self.flagButScroll != 0):

break

root.after(300, change())

but["text"] = self.focus

but.update()

self.flag = 0

root.after(1000)

self.focus = self.s[0:10]

but["text"] = self.focus

but.update()

def getName(self):

"""

Автор: Труханов А.И.

Возвращает имя

"""

return self.s

def setName(self,name):

"""

Автор: Гуняшов Н.Н.

ставит имя

"""

self.s = name

def change(i,j):

"""

Автор: Волков В.Д.

Изменения в базе данных

"""

self.base[i][self.sequence[j]] = self.entr[j][i].get()

self.entr[j][i].delete(0,tk.END)

self.entr[j][i].grid\_forget()

main.writeData(self.base)

#self.\_\_init\_\_(base)

self.p[j][i].grid(row = i, column = j+1)

#Поле = {"Название игры":0, "Жанр":1, "Платформа":2, "Год выпуска":3, "Цена":4, "Разработчик":5, "Издатель":6}

# Search

self.exit = tk.Button( self.frame\_exit, text = "Выйти", command = root.destroy, bg = "white", fg="black")

self.pSkip = []

self.pos = []

self.entr = []

self.spacePos = tk.Button( self.frame\_sort, width = 10)

for j in range(7):#self.sequence:

self.pos.append(tk.Button( self.frame\_sort, width = self.width[self.sequence[j]], text = main.unfield[self.sequence[j]]))

toP = []

pToSkip = []

toEntr = []

i = 0

while ( i < len(self.base) ):

toEntr.append(tk.Entry( self.scrollF.interior, width = self.width[self.sequence[j]], bg = "white", fg="black"))

#toEntr[i-1].insert(0,self.base[i][self.sequence[j]])

pToSkip.append(skip(self.base[i][self.sequence[j]]))

toP.append(tk.Button(self.scrollF.interior, width = self.width[self.sequence[j]]))

i = i + 1

self.entr.append(toEntr)

self.pSkip.append(pToSkip)

self.p.append(toP)

i = 0

while ( i < len(self.base) ):

self.entr[j][i - 1].bind( "<Return>", lambda event, i=i, j=j: change(i-1,j))

self.p[j][i - 1].bind( "<Enter>", lambda event, i=i, j=j: self.pSkip[j][i-1].scroll(self.p[j][i-1]))

self.p[j][i - 1].bind( "<Leave>", lambda event, i=i, j=j: self.pSkip[j][i-1].cancelSkip())

self.p[j][i - 1].bind( "<Button-1>", lambda event, i=i, j=j: self.butChange(i-1,j))

i = i + 1

# Add

self.addSpace = tk.Label( self.frame\_add, width = 12 )

self.add = tk.Button( self.frame\_add, text = "Добавить", bg = "white", fg="black")

self.addNameGame = tk.Entry( self.frame\_add, width = self.width[self.sequence[0]] )

self.addPlat = tk.Entry( self.frame\_add, width = self.width[self.sequence[1]] )

self.addGenre = tk.Entry( self.frame\_add, width = self.width[self.sequence[2]] )

self.addYear = tk.Entry( self.frame\_add, width = self.width[self.sequence[3]] )

self.addDevel = tk.Entry( self.frame\_add, width = self.width[self.sequence[4]] )

self.addPublisher = tk.Entry( self.frame\_add, width = self.width[self.sequence[5]] )

self.addPrice = tk.Entry( self.frame\_add, width = self.width[self.sequence[6]]\*2 )

# init

self.init\_widget()

def init\_widget(self):

"""

Автор: Труханов А.И.

Ставит объекты в классе MainWindow

"""

self.spacePos.grid(row = 0, column = 0)

for i in range(7):

self.pos[i].bind('<ButtonRelease-1>', lambda event, i=i: self.sortDisp(event, main.unfield[self.sequence[i]]))

self.pos[i].grid( row = 0, column = i+1 )

self.exit.grid()

#self.scrollF.config(width = 500, heigth = 400)

#self.frame\_all.place( x = 100, y = 50, width = 2791, height = 1500 )

self.frame\_all.place( x = 10, y = 5, width = 1291, height = 1500 )

self.exit.bind('<ButtonRelease-1>')

#self.exit.place(x = 1050, y = 650, width = 75, height = 40)

self.frame\_sort.grid( row = 0, column = 0)

self.scrollF.grid( row = 1, column = 0)

#self.frame\_all.rowconfigure(2, weight=1)

self.frame\_add.grid( row = 3, column = 0)

#self.frame\_all.rowconfigure(4, weight=2)

self.frame\_search.grid( row = 5, column = 0)

self.frame\_exit.grid( row = 5, column = 1)

#output

def output():

"""

Автор: Гуняшов Н.Н.

Подведение итогов

"""

self.out = tk.Button( self.frame\_search, text = "Подведение итогов", bg = "white", fg="black")

self.out.bind("<Button-1>", lambda event: main.resulttxt(self.base))

self.out.grid( row = 0, column = 0)

output()

def outputBase():

"""

Автор: Волков В.Д.

Запись в файл

"""

self.outB = tk.Button( self.frame\_search, text = "Запись в файл", bg = "white", fg="black")

self.outB.bind("<Button-1>", lambda event: main.outBase(self.base))

self.outB.grid( row = 1, column = 0)

outputBase()

def searchBut():

"""

Автор: Труханов А.И.

Поиск по категориям

"""

self.sea = tk.Button( self.frame\_search, text = "Поиск по категориям", bg = "white", fg="black")

self.sea.bind("<Button-1>", lambda event: self.search())

self.sea.grid( row = 2, column = 0)

def init():

"""

Автор: Гуняшов Н.Н.

Вернуться к обычному режиму из режима просмотра поиска по категориям

"""

self.sea = tk.Button( self.frame\_search, text = "Вернуться", bg = "white", fg="black")

self.sea.bind("<Button-1>", lambda event: self.\_\_init\_\_(main.readData()))

self.sea.grid( row = 2, column = 0)

if (self.base != main.readData()):

init()

else:

searchBut()

# add

self.add.bind('<ButtonRelease-1>', lambda event: self.buttAdd(event))

self.addSpace.grid( row = 0, column = 0)

self.addNameGame.grid( row = 0, column = 1)

self.addPlat.grid( row = 0, column = 2)

self.addGenre.grid( row = 0, column = 3)

self.addYear.grid( row = 0, column = 4)

self.addDevel.grid( row = 0, column = 5)

self.addPublisher.grid( row = 0, column = 6)

self.addPrice.grid( row = 0, column = 7)

self.add.grid( row = 0, column = 8)

# functions

self.buttSort()

def buttSort(self):

"""

Автор: Волков В.Д.

Сортирует

"""

self.dele = []

def deleteBase(i):

"""

Автор: Труханов А.И.

Удаляет элемент из базы данных

"""

del self.base[i]

main.writeData(self.base)

self.\_\_init\_\_(self.base)

for j in range(7):

i = 0

while ( i < len(self.base)):

if ( j == 0 ):

self.dele.append(tk.Button(self.scrollF.interior, text = "Удалить", width = self.width[0]))

self.dele[-1].bind( '<Button-1>', lambda event, i=i: deleteBase(i) )

self.dele[-1].grid( row = i, column = 0)

self.pSkip[j][i].setName(self.base[i][self.sequence[j]])

self.p[j][i]["text"] = self.pSkip[j][i].getName()[0:10]

self.p[self.sequence[j]][i].grid( row = i, column = self.sequence[j]+1)

i = i + 1

def butChange(self, i, j ):

"""

Автор: Гуняшов Н.Н.

Меняет кнопку на текстовое поле

"""

self.p[j][i].grid\_forget()

self.entr[j][i].grid( row = i, column = j+1)

def buttAdd(self, event):

"""

Автор: Волков В.Д.

Добавление новых значений

"""

a = []

# appends

a.append(self.addNameGame.get())

a.append(self.addGenre.get())

a.append(self.addPlat.get())

a.append(self.addYear.get())

a.append(self.addPrice.get())

a.append(self.addDevel.get())

a.append(self.addPublisher.get())

flag = 1

for i in a:

if (len(i) == 0):

flag = 0

if (flag == 1):

# delete

self.addNameGame.delete(1,tk.END)

self.addPlat.delete(1,tk.END)

self.addGenre.delete(1,tk.END)

self.addYear.delete(1,tk.END)

self.addDevel.delete(1,tk.END)

self.addPublisher.delete(1,tk.END)

self.addPrice.delete(1,tk.END)

# add to base

main.addRecord(self.base,a)

self.base.append(a)

self.\_\_init\_\_(self.base)

self.buttAdd()

#self.buttSort()

self.addNameGame.place( x = 100, y = 600)

self.addPlat.place( x = 226, y = 600)

self.addGenre.place( x = 328, y = 600)

self.addYear.place( x = 413, y = 600)

self.addDevel.place( x = 515, y = 600)

self.addPublisher.place( x = 674, y = 600)

self.addPrice.place( x = 833, y = 600)

def sortDisp(self, event, newSort):

"""

Автор: Труханов А.И.

Сортирует

"""

if ( self.currSort == newSort ):

self.flagSort = (self.flagSort + 1) % 2

else:

self.flagSort = 1

self.base = main.sort(newSort, self.flagSort)

self.currSort = newSort

self.buttSort()

def search(self):

"""

Автор: Гуняшов Н.Н.

Выводит меню для параметров поиска

"""

def end():

"""

Автор: Волков В.Д.

Осуществляет поиск

"""

a = self.entryTop1.get()

b = self.entryTop2.get()

c = self.entryTop3.get()

d = self.entryTop4.get()

self.\_\_init\_\_(main.search(a,b,c,d))

self.Top.destroy()

self.Top = tk.Toplevel()

self.label = tk.Label(self.Top, text = "Нижний порог отсеивания цены")

self.label.grid( row = 0, column = 0)

self.entryTop1 = tk.Entry(self.Top, width = 10)

self.entryTop1.grid( row = 1, column = 0)

self.label = tk.Label(self.Top, text = "Верхний порог отсеивания цены")

self.label.grid( row = 2, column = 0)

self.entryTop2 = tk.Entry(self.Top, width = 10)

self.entryTop2.grid( row = 3, column = 0)

self.label = tk.Label(self.Top, text = "Нижний порог отсеивания года")

self.label.grid( row = 4, column = 0)

self.entryTop3 = tk.Entry(self.Top, width = 10)

self.entryTop3.grid( row = 5, column = 0)

self.label = tk.Label(self.Top, text = "Верхний порог отсеивания года")

self.label.grid( row = 6, column = 0)

self.entryTop4 = tk.Entry(self.Top, width = 10)

self.entryTop4.grid( row = 7, column = 0)

self.end = tk.Button(self.Top, text = "Поиск")

self.end.bind( "<Button-1>", lambda event: end())

self.end.grid( row = 8, column = 0)

root = tk.Tk()

from importlib.machinery import SourceFileLoader

main = SourceFileLoader("main.py", "../library/main.py").load\_module()

#import main

root.title("Games Date Base")

root.geometry('750x430')

window = MainWindow(main.readData())

root.mainloop()

display()

## ***Листинг файла main.py***

#!/usr/bin/python

# -\*- coding: utf-8 -\*-

field = {"Название игры":0, "Жанр":1, "Платформа":2, "Год выпуска":3, "Цена":4, "Разработчик":5, "Издатель":6}

unfield = {0:"Название игры", 1:"Жанр", 2:"Платформа", 3:"Год выпуска", 4:"Цена", 5:"Разработчик", 6:"Издатель"}

def readData():

"""

Автор Труханов А.И.

Читает базу

"""

import pickle as pi

fin = open('../data/data.pi', 'rb')

data = pi.load(fin)

return data

def writeData( data ):

"""

Автор: Гуняшов Н.Н.

Печатает дату

"""

import pickle as pi

fin = open('../data/data.pi', 'wb')

pi.dump(data, fin)

def addRecord(data,d):

"""

Автор: Волков В.Д.

Добавляет запись

"""

data.append(d)

writeData(data)

def posMore( vvod , vivod):

"""

Автор Труханов А.И.

Ищет по множеству параметров

"""

flag = 0

i = 0

games = readData()

print(vvod)

for a in games:

isInVivod = 0

for c in vivod:

if (c == a):

isInVivod = 1

if (not(isInVivod)):

while (i < len(vvod)):

for b in a:

if (flag != i+1):

print(b,vvod[i],b==vvod[i])

if (b == vvod[i]):

flag+=1

i+=1

if (flag == len(vvod)):

vivod.append(a)

print(a)

flag = 0

i = 0

flag = 0

def pos( vvod ):

"""

Автор: Гуняшов Н.Н.

Поиск по параметрам

"""

i = 0

j = 0

lis = []

vivod = []

while (j < len(vvod)):

if (vvod[j] == '|'):

lis.append(vvod[i:j])

posMore(lis,vivod)

lis = []

i = j+1

if (vvod[j] == '&'):

lis.append(vvod[i:j])

i = j+1

j+= 1

lis.append(vvod[i:j])

posMore(lis,vivod)

for a in vivod:

for b in a:

print(b)

print()

def sort( vvod , order ):

"""

Автор: Волков В.Д.

Сортирует

"""

output = []

priority = []

nombers = {}

games = readData()

if (vvod in field):

for a in games:

if (vvod == "Цена"):

priority.append(int(a[field[vvod]]))

else:

priority.append(a[field[vvod]])

priority = sorted(priority)

i = 0

for a in games:

i = 0

while ( a[field[vvod]] != str(priority[i]) or i in nombers):

i+=1

nombers[i]=a

if (order == 1):

j = 0

while (j < len(nombers)):

output.append(nombers[j])

j+=1

else:

if (order == 0):

j = len(nombers) - 1

while (j > -1):

output.append(nombers[j])

j-=1

else:

print("Incorrect input")

else:

print("Incorrect input")

return output

def search(a,b,c,d):

"""

Автор Труханов А.И.

Поиск в промежутке

"""

baseOut = []

base = readData()

if (len(a) > 0):

for e in base:

if (int(e[4]) > int(a)):

baseOut.append(e)

if (len(b) > 0):

if (len(baseOut) == 0):

for e in base:

if (int(e[4]) < int(b)):

baseOut.append(e)

else:

base = []

for e in baseOut:

if (int(e[4]) < int(b)):

base.append(e)

baseOut = base

if (len(c) > 0):

if (len(baseOut) == 0):

for e in base:

if (int(e[3]) > int(c)):

baseOut.append(e)

else:

base = []

for e in baseOut:

if (int(e[3]) > int(c)):

base.append(e)

baseOut = base

if (len(d) > 0):

if ( len(baseOut) == 0):

for e in base:

if (int(e[3]) < int(d)):

baseOut.append(e)

else:

base = []

for e in baseOut:

if (int(e[3]) < int(d)):

base.append(e)

baseOut = base

return baseOut

def outBase( data ):

"""

Автор: Гуняшов Н.Н.

Выводит базу

"""

fin = open('../output/base.txt', 'w')

for a in data:

for b in a:

print(b, file=fin)

print(file=fin)

fin.close()

def resulttxt(data):

"""

Автор: Волков В.Д.

Считает ср. арифм.

"""

fin = open('../output/result.txt', 'w')

d=0

summ=0

sr=0

disp=0

otkl=[]

summotkl=0

for a in data:

d=d+1

print("Кол-во записей:",file=fin)

print(d,file=fin)

for a in data:

summ=summ+int(a[4])

sr=summ/d

print("Среднее арифметическое:",file=fin)

print(int(sr),file=fin)

for a in data:

otkl.append((int(sr)-int(a[4]))\*\*2)

for a in otkl:

summotkl=summotkl+a

disp=summotkl/d

print("Дисперсия:",file=fin)

print(int(disp),file=fin)