**ФГБОУ ВО “МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”**

**Лабораторная работа № 7**

Разветвляющиеся вычислительные процессы

**Задание 1 Вариант № 5**

по дисциплине:

Основы программирования

Выполнил:

студент 1 курса

группы 201-322

Кириллов Д.П

**МОСКВА 2020**

**Постановка задачи:**

Написать программу которая выполняет следующие действия:

* Корректировку или дополнение расписания с клавиатуры
* Сортировку по станции назначения или по времени отправления
* Вывод на экран информации о поездах, отходящих после введенного времени
* Запись расписания в файл под тем же или новым именем

**Теоретическая часть:**

Для решения задачи была использованы: библиотека tkinter для создания десктопного приложения, ООП, работа с файлами через объект file

**Описание программы:**

Программа написана на алгоритмическом языке Python 3.8.5, реализована в среде ОС Windows 10

**Описание алгоритма:**

1. Создание всех компонентов с помощью tkinter с использованием ООП
2. Заполнение экземпляра List данными из файла rasp.txt
3. Генерация колбеков для обработки различных действий клиентом

**Пояснение к программе:**

**saveChangeInFile –** данная функция является вспомогательной, она используется в других обработчиках. Позволяет сохранить изменения произошедшие на экране в файл rasp.txt

**addListItem –** функция-обработчик при нажатии на кнопку «Добавить». Берет значение из поля ввода и вставляет его в обьект list.list, list.listItems, а после запускает функцию saveChangeInFile()

**removeListItem –** функция-обработчик позволяет удалить выделенный элемент из списка tkinter list.list и list.listItems, также запускает функцию saveChangeInFile()

**changeListItem –** функция-обработчик которая берет значение из поля ввода и заменяет выделенный элемент на значение поля ввода. Запускает saveChangeInFile()

**onSelectItem** – функция-обработчик, подставляет выделенный элемент в поле ввода

**onClickSort** – функция-обработчик, запускается когда нажимается кнопка «Отсортировать по времени». Сортирует элементы по времени.

**onClickFilter –** функция-обработчик которая вызывается при нажатии кнопки «Показать актуальные». Берет значение из поля ввода и показывает только те пункты расписания, которые являются актуальными относительно введенной даты

**onClickShowAll** – функция-обработчик, вызывается при нажатии на кнопку «Показать все», снимает фильтрацию и сортировку по времени, показывает элементы списка в обычном состоянии

**Класс EntryInput:**

Данный класс используется как поле ввода для добавления/изменения новых пунктов расписания. Есть свойство self.entry в котором хранится сам обьект Entry из библиотки tkinter

**Класс ButtonAdd:**

Этот класс генерирует кнопку «Добавить» и привязывает обработчик addListItem() при нажатии на кнопку. Есть свойство self.btn в котором хранится сам объект Buton из библиотки tkinter

**Класс ButtonChange:**

Этот класс генерирует кнопку «Изменить» и привязывает обработчик при changeListItem() нажатии на кнопку. Есть свойство self.btn в котором хранится сам обьект Button из библиотки tkinter

**Класс ButtonRemove:**

Этот класс генерирует кнопку «Удалить» и привязывает обработчик при removeListItem() нажатии на кнопку. Есть свойство self.btn в котором хранится сам обьект Button из библиотки tkinter

**Класс List:**

Этот класс представляет из себя основной элемент программы. В экземпляре этого класса хранится список расписания (self.list), которое при инициализации заполняется из файла rasp.txt и вставляется в обьект tkinter. Функция onSelectItem обрабатывает выделение клиентом какого-либо элемента в списке, что позволяет нам вставить значение этого пункта в поле ввода

**Класс ButtonSort:**

Этот класс генерирует кнопку «Отсортировать по времени» и привязывает обработчик onClickSort() при нажатии на кнопку. Есть свойство self.btn в котором хранится сам обьект Button из библиотки tkinter

**Класс ButtonFilter:**

Этот класс генерирует кнопку «Показат актуальные» и привязывает обработчик onClickFilter() при нажатии на кнопку. Есть свойство self.btn в котором хранится сам обьект Button из библиотки tkinter

**Класс ButtonShowAll:**

Этот класс генерирует кнопку «Показать все» и привязывает обработчик onClickShowAll() при нажатии на кнопку. Есть свойство self.btn в котором хранится сам обьект Button из библиотки tkinter

**Листинг программы:**

from tkinter import \*

from datetime import datetime, date, time

def saveChangeInFile():

file = open('rasp.txt', 'w', encoding="utf-8")

for lineItem in list.listItems:

file.write(lineItem + '\n')

file.close()

def addListItem():

textInput = entryInput.entry.get()

if (len(textInput) == 0):

return

list.list.insert(END, textInput)

list.listItems.append(textInput)

entryInput.entry.delete(0, END)

saveChangeInFile()

def removeListItem():

if (len(list.list.curselection()) == 0):

return

indexActive = list.list.curselection()[0]

list.list.delete(indexActive)

list.listItems.pop(indexActive)

entryInput.entry.delete(0, END)

saveChangeInFile()

def changeListItem():

if (len(list.list.curselection()) == 0):

return

indexActive = list.list.curselection()[0]

text = entryInput.entry.get()

list.list.delete(indexActive)

list.list.insert(indexActive, text)

list.listItems[indexActive] = text

entryInput.entry.delete(0, END)

saveChangeInFile()

def onSelectListItem(e):

indexActive = list.list.curselection()

if (len(indexActive) == 0):

return

text = list.list.get(indexActive)

entryInput.entry.delete(0, END)

entryInput.entry.insert(0, text)

def onClickSort():

file = open('rasp.txt', encoding='utf-8')

sortArr = file.readlines()

file.close()

for i in range(len(sortArr) - 1):

for j in range(len(sortArr) - i - 1):

str1 = sortArr[j][0:16]

date1 = datetime.strptime(str1, '%d/%m/%Y %H:%M')

str2 = sortArr[j + 1][0:16]

date2 = datetime.strptime(str2, '%d/%m/%Y %H:%M')

if ((date1 - date2).total\_seconds() > 0):

sortArr[j], sortArr[j + 1] = sortArr[j + 1], sortArr[j]

list.list.delete(0, END)

entryInput.entry.delete(0, END)

for line in sortArr:

lineFormat = line.replace('\n', '')

list.list.insert(END, lineFormat)

def onClickFilter():

try:

textInput = entryInput.entry.get()

dateInput = datetime.strptime(textInput[0:16], '%d/%m/%Y %H:%M')

filterArr = []

for line in list.listItems:

dateLine = datetime.strptime(line[0:16], '%d/%m/%Y %H:%M')

if ((dateLine - dateInput).total\_seconds() >= 0):

filterArr.append(line)

list.list.delete(0, END)

for line in filterArr:

lineFormat = line.replace('\n', '')

list.list.insert(END, lineFormat)

except ValueError:

return

def onClickShowAll():

list.list.delete(0, END)

for line in list.listItems:

list.list.insert(END, line)

class LabelRasp:

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

self.lbl = Label(root)

self.lbl['text'] = 'Расписание электричек'

self.lbl['fg'] = '#fff'

self.lbl['bg'] = '#222'

self.lbl['font'] = 'Arial 12'

self.lbl.pack(side = 'top', fill = 'x', ipady=8)

class EntryInput:

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

self.entry = Entry(root)

self.entry['width'] = 10

self.entry.place(relx=.5, rely=.8, anchor="c", height=20, width=250)

class ButtonAdd:

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

self.btn = Button(root)

self.btn['text'] = 'Добавить'

self.btn['command'] = addListItem

self.btn.place(relx=.2, rely=.85, anchor="c", height=20, width=70)

class ButtonChange:

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

self.btn = Button(root)

self.btn['text'] = 'Изменить'

self.btn['command'] = changeListItem

self.btn.place(relx=.5, rely=.85, anchor="c", height=20, width=70)

class ButtonRemove:

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

self.btn = Button(root)

self.btn['text'] = 'Удалить'

self.btn['command'] = removeListItem

self.btn.place(relx=.8, rely=.85, anchor="c", height=20, width=70)

class List:

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

self.list = Listbox(root)

file = open('rasp.txt', encoding='utf-8')

lines = file.readlines()

self.listItems = []

for line in lines:

lineFormat = line.replace('\n', '')

self.listItems.append(lineFormat)

file.close()

self.list.bind('<<ListboxSelect>>', onSelectListItem)

self.list['selectmode'] = 'EXTENDED'

self.list['width'] = 28

self.list['height'] = 16

self.list['font'] = 'Arial 13'

self.list.place(relx=.5, rely=.43, anchor="c")

for line in self.listItems:

self.list.insert(END, line)

class ButtonSort:

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

self.btn = Button(root)

self.btn['text'] = 'Сортировка по времени'

self.btn['command'] = onClickSort

self.btn.place(relx=.5, rely=.9, anchor="c", height=20, width=250)

class ButtonFilter:

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

self.btn = Button(root)

self.btn['text'] = 'Показать актуальные'

self.btn['command'] = onClickFilter

self.btn.place(relx=.33, rely=.95, anchor="c", height=20, width=150)

class ButtonShowAll:

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

self.btn = Button(root)

self.btn['text'] = 'Показать все'

self.btn['command'] = onClickShowAll

self.btn.place(relx=.765, rely=.95, anchor="c", height=20, width=90)

root = Tk()

root.title("Расписание электричек")

root.geometry('300x500')

labelRaspObj = LabelRasp()

entryInput = EntryInput()

btnAdd = ButtonAdd()

btnChange = ButtonChange()

btnRemove = ButtonRemove()

btnSort = ButtonSort()

btnFiler = ButtonFilter()

btnShowAll = ButtonShowAll()

list = List()

root.mainloop()

**Использованная литература**

pythonworld.ru Справочник для начинающих. Python 3

pythoner.name Сайт о языке программирования Python. Статьи о возможностях Python, перевод официальной документации