Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №5**

**«ИССЛЕДОВАНИЕ АЛГОРИТМОВ СОРТИРОВКИ»**

**ПО ДИСЦИПЛИНЕ «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»**

Выполнил: студент учебной группы

ИСПк-203-52-00

Куликов Иван Алексеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров, 2022

**Цель работы:** получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами.

**Задание (Вариант 12):**

1. Реализовать сортировку данных с помощью вставок.
2. Реализовать сортировку данных с помощью быстрого алгоритма.
3. В обоих случаях необходимо предусмотреть возможность изменения компаратора (реализация компаратора в виде передаваемой в подпрограмму функции).
4. Считывание и вывод данных необходимо производить из текстового файла.
5. Для демонстрации работы программных реализаций самостоятельно подготовить варианты входных данных (при этом объём текстовых файлов должен позволять оценить скорость работы программ).

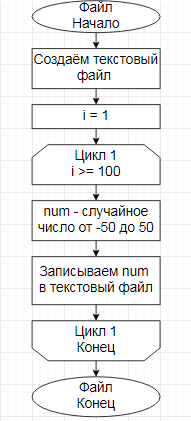
**Схема алгоритма**

Рисунок 1 – Схема алгоритма подпрограммы File

**Код программы:**

from tkinter import \*

from tkinter import messagebox

import random

import time

def file():

    with open('input.txt', 'w', encoding='utf-8') as file:

        for i in range(100):

            num = random.randint(-50, 51)

            file.write(str(num) + '\n')

    return file

def vstavki\_sort():

    start = time.perf\_counter()

    f = open('input.txt',encoding = 'utf-8')

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

    alist = []

    for line in f:

        alist.append(line.rstrip())

    n = len(alist)

    mass = alist

    f.close()

    for i in range(len(mass)):

        for j in range(n-i-1):

            if mass[j] > mass[j+1]:

                mass[j], mass[j+1] = mass[j+1], mass[j]

            i -= 1

    w.write(str(mass))

    w.close()

    stop  = time.perf\_counter()

    itog = stop - start

    print(itog)

    messagebox.showinfo('Время выполнения',f'{itog}')

def vstavki\_sort\_ub():

    start = time.perf\_counter()

    f = open('input.txt',encoding = 'utf-8')

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

    alist = []

    for line in f:

        alist.append(line.rstrip())

    n = len(alist)

    mass = alist

    f.close()

    for i in range(len(mass)):

        for j in range(n-i-1):

            if mass[j] < mass[j+1]:

                mass[j], mass[j+1] = mass[j+1], mass[j]

            i -= 1

    w.write(str(mass))

    w.close()

    stop  = time.perf\_counter()

    itog = stop - start

    print(itog)

    messagebox.showinfo('Время выполнения',f'{itog}')

def quickSort():

    start = time.perf\_counter()

    f = open('input.txt',encoding = 'utf-8')

    w = open("out.txt","w")

    alist = []

    for line in f:

        alist.append(line.rstrip())

    mass = alist

    f.close()

    q = random.choice(mass)

    s\_nums = []

    m\_nums = []

    e\_nums = []

    for n in mass:

        if n < q:

            s\_nums.append(n)

        elif n > q:

            m\_nums.append(n)

        else:

            e\_nums.append(n)

    w.write(str(s\_nums + m\_nums + e\_nums))

    w.close()

    stop  = time.perf\_counter()

    itog = stop - start

    print(itog)

    messagebox.showinfo('Время выполнения',f'{itog}')

def quickSort\_ub():

    start = time.perf\_counter()

    f = open('input.txt',encoding = 'utf-8')

    w = open("out.txt","w")

    alist = []

    for line in f:

        alist.append(line.rstrip())

    mass = alist

    f.close()

    q = random.choice(mass)

    s\_nums = []

    m\_nums = []

    e\_nums = []

    for n in mass:

        if n < q:

            s\_nums.append(n)

        elif n > q:

            m\_nums.append(n)

        else:

            e\_nums.append(n)

    w.write(str(e\_nums + m\_nums + s\_nums))

    w.close()

    stop  = time.perf\_counter()

    itog = stop - start

    print(itog)

    messagebox.showinfo('Время выполнения',f'{itog}')

def ExitApp():

    MsgBox = messagebox.askquestion ('Выход из программы','Вы уверены, что хотите выйти?',icon = 'error')

    if MsgBox == 'yes':

       root.destroy()

    else:

        messagebox.showinfo('С возвращением!','Мы рады, что вы остались с нами!')

root = Tk()

root.title("Сортировки")

root.geometry("670x325")

root.attributes('-alpha',0.95)

frame = Frame(root,padx=10,pady=10)

frame.pack(expand=True)

label = Label(frame,text='Задание (Вариант 12):',font=("Arial", 16))

label.pack()

label\_1 = Label(frame,text='1.Реализовать сортировку данных с помощью вставок.\n'

'2.Реализовать сортировку данных с помощью быстрого алгоритма.\n'

'3.В обоих случаях необходимо предусмотреть возможность изменения компаратора \n(реализация компаратора в виде передаваемой в подпрограмму функции).\n'

'4.Считывание и вывод данных необходимо производить из текстового файла.\n'

'5.Для демонстрации работы программных реализаций самостоятельно подготовить варианты \n входных данных (при этом объём текстовых файлов должен позволять оценить скорость работы программ).',justify=LEFT,font=('Arial',9))

label\_1.pack()

button = Button(frame, text="Изменить числа в исходном файле", command=file)

button.pack()

btn = Button(frame, text='Сортировка вставками (по возрастанию)',command = vstavki\_sort)

btn.pack(fill =X)

btnk = Button(frame, text='Сортировка вставками (по убыванию)',command = vstavki\_sort\_ub)

btnk.pack(fill =X)

butn = Button(frame,text = 'Быстрая сортировка (по возрастанию)',command = quickSort)

butn.pack(fill =X)

butnn= Button(frame,text = 'Быстрая сортировка (по убыванию)',command = quickSort\_ub)

butnn.pack(fill =X)

buttonEg = Button (frame, text='Выход',command=ExitApp)

buttonEg.pack(anchor=SE)

def motionUP(event):

    children = frame.winfo\_children()

    if event.widget in children:

        index = children.index(event.widget)

        index -= 1

        if index > -1:

            children[index].focus\_set()

def motionDOWN(event):

    children = frame.winfo\_children()

    if event.widget in children:

        index = children.index(event.widget)

        index += 1

        if index < len(children):

            children[index].focus\_set()

root.bind('<Up>', motionUP)

root.bind('<Down>', motionDOWN)

root.mainloop()

**Результат выполнения программы**

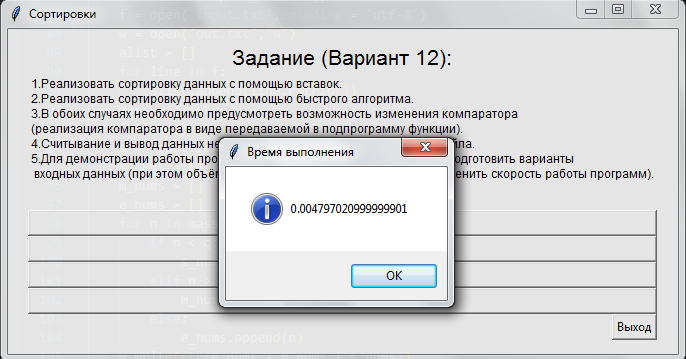
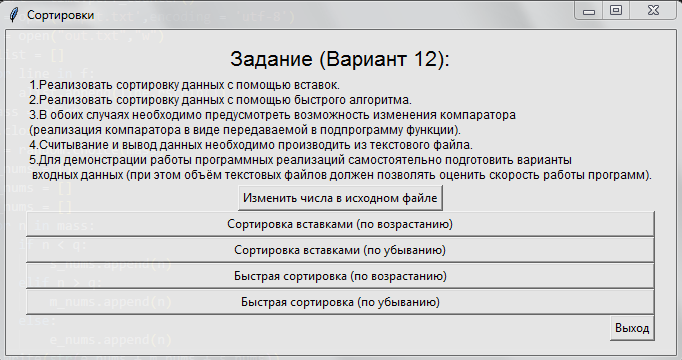
****

Рисунок 2 – Результат выполнения программы

Рисунок 2 – Пользовательский интерфейс

**Вывод:**

Исходя из общего положения и пользуясь в качестве довода и логического обоснования совокупность ранее упомянутых эмпирических знаний можно сделать закономерный вывод, что в ходе работы мы использовали знания, полученные нами на столь замечательной и бесподобной дисциплине, как Основы алгоритмизации и программирования. После домашней контрольной работы №5 мы можем смело сказать, что полученные на лекционных занятиях знания и умения в полной мере пригодились для грамотного и быстрого выполнения данной работы.

В ходе работы была изучена кроссплатформенная событийно-ориентированная графическая стандартная библиотека Python – Tkinter, благодаря которой был написан пользовательский интерфейс для программы, сортировки текстового файла. Также были отработаны навыки работы с функциями и процедурами в языке программирования Python. Их использование было необходимо для избежание дублирования кода при многократном его использовании.

Входе работы были получены навыки с работы с текстовыми файлами в Python. Также были изучены алгоритмы сортировки.

Подводя итоги, можно смело утверждать, что выполнение домашней контрольной работы №5 помогло получить базовые сведения о наиболее известных алгоритмах сортировки, изучить принципы работы с текстовыми файлами.