import tkinter as tk

from tkinter import ttk

from tkinter import messagebox

import time

import unittest

def bubble\_sort(sequence):

"""

функция пузырьковой сортировки

"""

n = len(sequence) #длинна массива

for i in range(n - 1): #цикл для итераций проверки

for j in range(0, n - i - 1): #цикл для прохода по массиву

if sequence[j] > sequence[j + 1]: #сравнение и перестановка

sequence[j], sequence[j + 1] = sequence[j + 1], sequence[j]

return sequence #возврат отсортированного массива

def quick\_sort(sequence):

"""

функция быстрой сортировки

"""

if len(sequence) <= 1:

return sequence

pivot = sequence[len(sequence) // 2]

left = [x for x in sequence if x < pivot] #Добавляем все которые меньше

middle = [x for x in sequence if x == pivot] #Добавляем все которые равны

right = [x for x in sequence if x > pivot] #Добавляем все которые больше

return quick\_sort(left) + middle + quick\_sort(right) #Рекурсивно уходим на 2 получившихся массива и возвращаем итог

def sort\_sequence(sequence, sorting\_method):

"""

функция запуска сортировки

"""

start\_time = time.time()

try:

sequence = [float(num) for num in sequence.split(',')] #Выбор функции в соответствии с выбранным методом

if sorting\_method == 'Пузырьком':

sorted\_sequence = bubble\_sort(sequence)

elif sorting\_method == 'Быстрая':

sorted\_sequence = quick\_sort(sequence)

else:

raise ValueError("Неверный метод сортировки")

end\_time = time.time()

return sorted\_sequence, end\_time - start\_time #возвращение результата и времени

except ValueError as e:

raise ValueError("Ошибка ввода: " + str(e))

"""

Создаем окно программы

"""

class SortingApp(tk.Tk): #Окно программы

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

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

self.title("Сортировка чисел") #Сборка окна

self.geometry("400x300")

self.label = tk.Label(self, text="Введите последовательность чисел через запятую:")

self.label.pack(pady=10)

self.entry = tk.Entry(self)

self.entry.pack(pady=10)

"""

Раскрывающееся меню

"""

self.sorting\_options = ['Пузырьком', 'Быстрая']

self.sorting\_var = tk.StringVar(self)

self.sorting\_var.set(self.sorting\_options[0])

self.sorting\_menu = ttk.Combobox(self, values=self.sorting\_options, textvariable=self.sorting\_var)

self.sorting\_menu.pack(pady=10)

self.result\_text = tk.Text(self, height=5, width=40)

self.result\_text.pack(pady=10)

self.start\_button = tk.Button(self, text="Отсортировать", command=self.start\_sorting)

self.start\_button.pack(pady=10)

def start\_sorting(self):

"""

Определяем функцю для кнопки "Отсортировать"

"""

try:

input\_sequence = self.entry.get() #вытянули массив и выбранный метод

sorting\_method = self.sorting\_var.get()

sorted\_sequence, sorting\_time = sort\_sequence(input\_sequence, sorting\_method) #запуск сортировки

#результат в поле

result\_message = f"Итог({self.sorting\_var.get()}): {', '.join(map(str, sorted\_sequence))}\n" \

f"Время сортировки: {sorting\_time:.6f} сек"

self.result\_text.delete(1.0, tk.END) #Отчиста поля и вывод получившегося результата

self.result\_text.insert(tk.END, result\_message)

except ValueError as e:

messagebox.showerror("Ошибка", str(e))

"""

Опредлеяем тесты

Проверка на правильность сортировки

Проверка на правильный тип времени

Проверка на не отрицательность времни

"""

class TestSortingApp(unittest.TestCase): #Класс для юнит тестов

def test\_sort\_sequence(self):

# Проверка корректной сортировки

sorted\_sequence, sorting\_time = sort\_sequence("3,1,4,1,5,9", 'Пузырьком')

self.assertEqual(sorted\_sequence, [1.0, 1.0, 3.0, 4.0, 5.0, 9.0]) #Проверка на правильность сортировки (равенство двух значений)

self.assertIsInstance(sorting\_time, float) #Проверка на правильный тип времени

self.assertTrue(sorting\_time >= 0.0) #Проверка на неотрицательность времни

sorted\_sequence, sorting\_time = sort\_sequence("3,1,4,1,5,9", 'Быстрая')

self.assertEqual(sorted\_sequence, [1.0, 1.0, 3.0, 4.0, 5.0, 9.0]) #Проверка на правильность сортировки

self.assertIsInstance(sorting\_time, float) #Проверка на правильный тип времени

self.assertTrue(sorting\_time >= 0.0) #Проверка на неотрицательность времни

# Проверка сортировки с неверным методом

with self.assertRaises(ValueError):

sort\_sequence("3,1,4,1,5,9", 'Неверный метод')

# Проверка сортировки с неверным методом

with self.assertRaises(ValueError):

sort\_sequence("3,1,4,1,5,9", 'Неверный метод')

#unittest.main()

app = SortingApp()

app.mainloop()