```
import tkinter as tk
import time

class finalApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Сортировка чисел")

        self.label1 = tk.Label(root, text="Введите последовательность чисел через запятую:")
        self.label1.pack()

        self.entry = tk.Entry(root)
        self.entry.pack()

        self.label2 = tk.Label(root, text="Выберите тип сортировки:")
        self.label2.pack()

        self.sort_type = tk.StringVar(root)
        self.sort_type.set("Пирамидальная")
        self.option_menu = tk.OptionMenu(root, self.sort_type, "Пирамидальная", "Пузырьком")
        self.option_menu.pack()

        self.start_button = tk.Button(root, text="Start", command=self.start_sorting)
        self.start_button.pack()

        self.output_label = tk.Label(root, text="Результат:")
        self.output_label.pack()

        self.output_text = tk.Text(root, height=10, width=50)
```

```
    self.output_text.pack()

def start_sorting(self):
    input_text = self.entry.get()
    if not input_text:
        self.output_text.insert(tk.END, "Введите последовательность чисел!")
        return

    try:
        numbers = [int(x.strip()) for x in input_text.split(",")]
    except ValueError:
        self.output_text.insert(tk.END, "Ошибка! Проверьте правильность ввода чисел.")
        return

    start_time = time.time()
    if self.sort_type.get() == "Пирамидальная":
        sorted_numbers = self.heap_sort(numbers)
    else:
        sorted_numbers = self.bubble_sort(numbers)
    end_time = time.time()

    self.output_text.insert(tk.END, f"Отсортированная последовательность: {sorted_numbers}\n")
    self.output_text.insert(tk.END, f"Время сортировки: {end_time - start_time:.6f} сек.")

# Реализация пирамидальной сортировки
def heapify(self, arr, n, i):
    largest = i
    l = 2 * i + 1  # левый потомок
    r = 2 * i + 2  # правый потомок
```

```
    if l < n and arr[i] < arr[l]:
        largest = l

    if r < n and arr[largest] < arr[r]:
        largest = r

    if largest != i:
        arr[i], arr[largest] = arr[largest], arr[i]  # меняем местами
        self.heapify(arr, n, largest)

def heap_sort(self, arr):
    n = len(arr)

    # Строим максимальную пирамиду.
    for i in range(n // 2 - 1, -1, -1):
        self.heapify(arr, n, i)

    # Постепенно извлекаем элементы
    for i in range(n - 1, 0, -1):
        arr[i], arr[0] = arr[0], arr[i]  # меняем местами
        self.heapify(arr, i, 0)

    return arr

# Реализация пузырька
def bubble_sort(self, arr):
    n = len(arr)
    for i in range(n - 1):
```

```
            for j in range(0, n - i - 1):
                if arr[j] > arr[j + 1]:
                    arr[j], arr[j + 1] = arr[j + 1], arr[j]
        return arr


root = tk.Tk()
app = finalApp(root)
root.mainloop()
```