**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки Кафедра ІПІ**

Звіт

з лабораторної роботи № 3 з дисципліни

«Алгоритми та структури даних 2. Структури даних»

**“Метод швидкого сортування”**

**Виконав(ла)**

Скрипець Ольга Олександрівна

(шифр, прізвище, ім'я, по батькові)

**Перевірив**

(прізвище, ім'я, по батькові)

Київ 2022

**Формальна постановка задачі**

Реалізувати наступні три модифікації алгоритму швидкого сортування (Quick Sort) та порівняти їх швидкодію. Швидкість алгоритмів порівнюється на основі підрахунку кількості порівнянь елементів масиву під час роботи алгоритмів.

**Формат вхідних/вихідних даних**

Розроблена програма повинна зчитувати вхідні дані з файлу заданого формату та записувати дані у файл заданого формату.

Вхідний файл представляє собою текстовий файл із N+1 рядків, де N — це розмірність вхідного масиву A. Першим записом є число — кількість елементів в масиві; наступні N записів містять елементи вхідного масиву.

Вихідний файл представляє текстовий файл з одним рядком. Вміст цього рядка наступний: X Y Z, де X — це кількість порівнянь під час роботи алгоритму №1 над заданим вхідним масивом, Y— кількість порівнянь алгоритму №2, та Z — кількість порівнянь алгоритму №3.

**Реалізація коду**

#алгоритм швидкого сортування 1-------------------------------------------

def quicksort\_last(array, left, right):

if left < right and len(array) > 0:

pivot = partition\_last(array, left, right)

quicksort\_last(array, left, pivot - 1)

quicksort\_last(array, pivot + 1, right)

def partition\_last(array, left, right):

global comparisons1 # лічильник порівнянь

pivot = array[right]

i = left - 1

for j in range(left, right):

if array[j] < pivot:

i += 1

array[i], array[j] = array[j], array[i]

comparisons1 += 1 # підрахунок порівнянь

array[i+1], array[right] = array[right], array[i+1]

return i+1

#алгоритм швидкого сортування 2-------------------------------------------

def median\_of\_three(a, mid, c):

return sorted([a, mid, c])[1]

def quick\_sort\_median(arr, p, r):

comparisons = 0

if p < r:

if r - p + 1 > 3:

i, comparisons = part\_of\_median(arr, p, r)

comparisons += quick\_sort\_median(arr, p, i - 1)

comparisons += quick\_sort\_median(arr, i + 1, r)

else:

comparisons = insertion\_sort(arr, p, r)

return comparisons

def part\_of\_median(arr, p, r):

comparisons = 0

mid = (p+r)//2

median = median\_of\_three(arr[p], arr[mid], arr[r])

if median == arr[p]:

pivot\_index = p

elif median == arr[mid]:

pivot\_index = mid

else:

pivot\_index = r

arr[pivot\_index], arr[r] = arr[r], arr[pivot\_index]

i = p - 1

for j in range(p, r):

comparisons += 1

if arr[j] <= median:

i += 1

arr[i], arr[j] = arr[j], arr[i]

arr[i + 1], arr[r] = arr[r], arr[i + 1]

return i + 1, comparisons

def insertion\_sort(arr, p, r):

comparisons = 0

for i in range(p+1, r+1):

key = arr[i]

j = i - 1

while j >= p and (comparisons := comparisons+1) and arr[j] > key:

arr[j+1] = arr[j]

j -= 1

arr[j+1] = key

return comparisons

def read\_input\_file(filename):

with open(filename, 'r') as f:

n = int(f.readline().strip())

arr = [int(f.readline().strip()) for \_ in range(n)]

return arr

def write\_output\_file(filename, result):

with open(filename, 'w') as f:

f.write(str(result))

# Зчитування вхідних даних з файлу та сортування

with open("input\_11\_1000.txt", "r") as infile:

n = int(infile.readline())

array1 = []

for i in range(n):

x = int(infile.readline())

array1.append(x)

array2 = array1.copy()

comparisons1 = 0

print(array1)

quicksort\_last(array1, 0, n-1)

print(array1)

comparisons2=quick\_sort\_median(array2,0,len(array1) - 1)

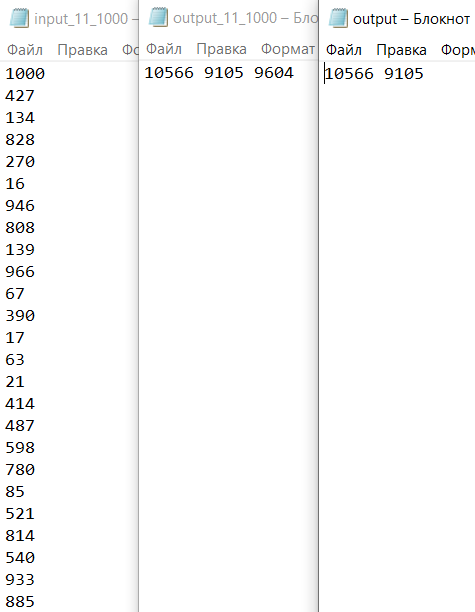
# Запис результату у вихідний файл

with open("output.txt", "w") as outfile:

#outfile.write(str(comparisons2))

outfile.write(f"{comparisons1} {comparisons2}")

**Приклад вхідного/вихідного файлу**

****