Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Катедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з дисципліни
«Проєктування алгоритмів»

„ **Проєктування і аналіз алгоритмів зовнішнього сортування**”

Виконав(ла) *ІП-11 Лесів В.І.* _____
(шифр, прізвище, ім'я, по батькові)

Перевірив *Головченко М.М.* _____
(прізвище, ім'я, по батькові)

Київ 2022

ЗМІСТ

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Мета роботи – вивчити основні алгоритми зовнішнього сортування та способи їх модифікації, оцінити поріг їх ефективності.

2 ЗАВДАННЯ

Згідно варіанту (таблиця 2.1), розробити та записати алгоритм зовнішнього сортування за допомогою псевдокоду (чи іншого способу за вибором).

Виконати програмну реалізацію алгоритму на будь-якій мові програмування та відсортувати випадковим чином згенерований масив цілих чисел, що зберігається у файлі (розмір файлу має бути не менше 10 Мб, можна значно більше).

Здійснити модифікацію програми і відсортувати випадковим чином згенерований масив цілих чисел, що зберігається у файлі розміром не менше ніж двократний обсяг ОП вашого ПК. Досягти швидкості сортування з розрахунку 1Гб на 3хв. або менше.

Рекомендується попередньо впорядкувати серії елементів довжиною, що займає не менше 100Мб або використати інші підходи для пришвидшення процесу сортування.

Зробити узагальнений висновок з лабораторної роботи, у якому порівняти базову та модифіковану програми. У висновку деталізувати, які саме модифікації було виконано і який ефект вони дали.

Таблиця 2.1 – Варіанти алгоритмів

№	**Алгоритм сортування**
1	Пряме злиття
2	Природне (адаптивне) злиття
3	Збалансоване багатошляхове злиття
4	Багатофазне сортування
5	Пряме злиття
6	Природне (адаптивне) злиття
7	Збалансоване багатошляхове злиття
8	Багатофазне сортування
9	Пряме злиття
10	Природне (адаптивне) злиття

11	Збалансоване багатошляхове злиття
12	Багатофазне сортування
13	Пряме злиття
14	Природне (адаптивне) злиття
15	Збалансоване багатошляхове злиття
16	Багатофазне сортування
17	Пряме злиття
18	Природне (адаптивне) злиття
19	Збалансоване багатошляхове злиття
20	Багатофазне сортування
21	Пряме злиття
22	Природне (адаптивне) злиття
23	Збалансоване багатошляхове злиття
24	Багатофазне сортування
25	Пряме злиття
26	Природне (адаптивне) злиття
27	Збалансоване багатошляхове злиття
28	Багатофазне сортування
29	Пряме злиття
30	Природне (адаптивне) злиття
31	Збалансоване багатошляхове злиття
32	Багатофазне сортування
33	Пряме злиття
34	Природне (адаптивне) злиття
35	Збалансоване багатошляхове злиття

3 ВИКОНАННЯ

3.1 Псевдокод алгоритму

```
function splitF()
    a ← open("A.txt");
    b, c ← open("B.txt", "w"), open("C.txt", "w");
    cur ← [];
    p ← 1;
    for line in a do
        if cur = [] or int(line.split()[0]) >= int(cur[-1]) then
            cur ← cur + line.split();
        else:
            if p = 1 then
                b.write(cur);
                p ← 2;
            else:
                c.write(cur);
                p ← 1;
            cur ← line.split();
    end for;
    if p = 1 then
        b.write(cur);
    else:
        c.write(cur);
    a.close();
    b.close();
    c.close();
    mergeF();
end fuction;


function mergeF()
    a ← open("A.txt", "w");
    b, c ← open("B.txt"), open("C.txt");
    if length(b.readlines()) < length(c.readlines()) then
        b.close();
        c.close();
        c, b ← open("B.txt"), open("C.txt");
    b.seek(0);
    c.seek(0);
    for lineb in b do
        lc ← [int(i) for i in c.readline().split()];
```

```
            lb ← [int(i) for i in lineb.split()];
            la, i, j ← [], 0, 0;
            while i != length(lb) and j != length(lc) do
                if lb[i] < lc[j] then
                    la.append(lb[i]);
                    i ← i + 1;
                else:
                    la.append(lc[j]);
                    j ← j + 1;
            end while;
            while i != length(lb) do
                la.append(lb[i]);
                i ← i + 1;
            end while;
            while j != length(lc) do
                la.append(lc[j]);
                j ← j + 1;
            end while;
            a.write(" ".join([str(i) for i in la]) + "\n");
        end for;
        for linec in c do
            a.write(linec);
        end for;
        a.close();
        b.close();
        c.close();
        with open("A.txt") as a do
            if length(a.readlines()) != 1 then
                splitF();
    end function;
```

3.2 Програмна реалізація алгоритму

3.2.1 Вихідний код

main.py

```
from datetime import datetime
from generation import generation
from sorting import splitF

s = datetime.now()
generation()
t = datetime.now() - s
print("Generation took",t)
input("Press Enter to start sorting.")
s = datetime.now()
splitF()
```

```
t = datetime.now() - s
print("Sorting took",t)
```

generation.py

```
from random import randint


def generation():
    n = float(input("Size of the file (MB): "))
    with open("A.txt", "w") as a:
        size = 0
        while size < 1048576 * (n+2):
            tmp = str(randint(0, 99999999)) + "\n"
            size += len(tmp) + 1
            a.write(tmp)
```

sorting.py

```
def splitF():
    a = open("A.txt")
    b, c = open("B.txt", "w"), open("C.txt", "w")
    cur = []
    p = 1
    for line in a:
        if cur == [] or int(line.split()[0]) >= int(cur[-1]):
            cur += line.split()
        else:
            if p == 1:
                b.write(" ".join(cur) + "\n")
                p = 2
            else:
                c.write(" ".join(cur) + "\n")
                p = 1
            cur = line.split()
    if p == 1:
        b.write(" ".join(cur))
    else:
        c.write(" ".join(cur))
    a.close()
    b.close()
    c.close()
    mergeF()


def mergeF():
    a = open("A.txt", "w")
    b, c = open("B.txt"), open("C.txt")
    if len(b.readlines()) < len(c.readlines()):
        b.close()
        c.close()
        c, b = open("B.txt"), open("C.txt")
    b.seek(0)
    c.seek(0)
    for lineb in b:
        lc = [int(i) for i in c.readline().split()]
        lb = [int(i) for i in lineb.split()]
        la, i, j = [], 0, 0
        while i != len(lb) and j != len(lc):
            if lb[i] < lc[j]:
                la.append(lb[i])
                i += 1
```

```
            else:
                la.append(lc[j])
                j += 1
        while i != len(lb):
            la.append(lb[i])
            i += 1
        while j != len(lc):
            la.append(lc[j])
            j += 1
        a.write(" ".join([str(i) for i in la]) + "\n")
    for linec in c:
        a.write(linec)
    a.close()
    b.close()
    c.close()
    with open("A.txt") as a:
        if len(a.readlines()) != 1:
            splitF()
```

ВИСНОВОК

При виконанні даної лабораторної роботи я отримав практичні навички роботи з алгоритмами зовнішнього сортування, а саме записав псевдокод алгоритму природного (адаптивного) злиття, виконав програмну реалізацію цього методу, а також протестував її.

У якості додаткової пам’яті використовується два додаткові файли. Вміст початкового файлу послідовно відсортованими частинами розподіляється між додатковими файлами, а потім відповідні частини двох файлів, об’єднуючись, у зберігають відсортований стан у початковому файлі.

Отже, зовнішнє сортування злиттям зручно використовувати тоді, коли дані, які потрібно відсортувати, занадто великі, щоб помістится в оперативну пам'ять.

КРИТЕРІЇ ОЦІНЮВАННЯ

У випадку здачі лабораторної роботи до 09.10.2022 включно максимальний бал дорівнює – 5. Після 09.10.2022 максимальний бал дорівнює – 1.

Критерії оцінювання у відсотках від максимального балу:

- псевдокод алгоритму – 15%;
- програмна реалізація алгоритму – 40%;
- програмна реалізація модифікацій – 40%;
- висновок – 5%.