**Міністерство освіти і науки України**

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

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи № 3 з дисципліни

«Проектування алгоритмів»

„ **Проектування структур даних**”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-12 Казаков Володимир Олексійович*

**Перевірив**

(прізвище, ім'я, по батькові)

*Головченко М.Н.*

Київ 2022

Варіант 13

Мета роботи – вивчити основні підходи проектування та обробки складних структур даних.

Програмна реалізація (Python):

import random

import linecache

import tkinter

main\_file\_path = "main\_file.txt"

index\_file\_path = "index\_file.txt"

rows\_number = 1000

window = tkinter.Tk()

window.title("СУБД")

result\_field = tkinter.Label(text="Тут будуть показані результати")

result\_field.grid(row=0, column=0)

#Функція створення основного файлу

def create\_main\_file():

main\_file\_data = []

letters = "abcdefghijklmnopqrstuvwxyz"

for i in range(0, rows\_number):

main\_file\_data.append((str(i)

+ " "

+ "".join([letters[random.randint(0, 25)] for j in range(0, 30)]))

+ "\n")

with open(main\_file\_path, "w") as file:

for record in main\_file\_data:

file.write(record)

#Функція створення файлу з індексною областю

def create\_index\_file():

index\_file = open(index\_file\_path, "w")

for i in range(0, rows\_number):

with open(main\_file\_path, "r") as file:

for j, content in enumerate(file):

value = int(content.split(" ")[0])

if i == value:

index\_file.write(str(i) + " " + str(j) + "\n")

return i

#Функція пошуку елемента

def element\_search(row\_id=None):

comparisons\_number = 0

try:

if not row\_id:

primary\_key = int(entry\_input\_field.get())

else:

primary\_key = row\_id

except:

result\_field.configure(text=f"Неправильний формат")

return

upper\_index = rows\_number

lower\_index = 0

while lower\_index <= upper\_index:

middle\_index = (upper\_index + lower\_index) // 2

value = linecache.getline(index\_file\_path, middle\_index + 1)

comparisons\_number += 1

if not value:

result\_field.configure(text=f"Елемент не знайдено \n Число порівнянь: {str(comparisons\_number)}")

return "Елемент не знайдено"

if int(value.split(" ")[0]) == primary\_key:

result\_field.configure(text=f"Рядок у базі даних знайдено \n Знайдений елемент: {linecache.getline(main\_file\_path, int(value.split(' ')[1]))} \n Число порівнянь: {str(comparisons\_number)}")

return value

elif middle\_index < primary\_key:

lower\_index = middle\_index + 1

else:

upper\_index = middle\_index - 1

result\_field.configure(text=f"Елемент не знайдено \n Число порівнянь: {str(comparisons\_number)}")

return "Елемент не знайдено"

#Функція додавання елемента

def element\_add():

global lines\_number

row = str(lines\_number) + " " + " ".join(entry\_input\_field.get().split(" ")) + "\n"

with open(main\_file\_path, "a+") as file:

file.write(row)

file.close()

with open(index\_file\_path, "a+") as file1:

file1.write(row.split(" ")[0] + " " + str(lines\_number + 1) + "\n")

file1.close()

linecache.clearcache()

lines\_number += 1

result\_field.configure(text=f"Дані було успішно додано")

print("Дані було успішно додано")

return

#Функція видалення елементу

def element\_delete():

global lines\_number

try:

row\_id = int(entry\_input\_field.get())

search\_result = element\_search(row\_id).split(" ")

index\_value1 = int(search\_result[0])

index\_value2 = int(search\_result[1].replace("\n", ""))

with open(index\_file\_path, "r+") as file:

lines = []

for index, line in enumerate(file):

if int(line.split(" ")[0]) == index\_value1:

lines.append("")

else:

lines.append(line)

file.write("")

with open(index\_file\_path, "w") as file:

file.writelines(lines)

with open(main\_file\_path, "r+") as file1:

lines = []

for index, line in enumerate(file1):

if index == index\_value2-1:

lines.append("\n")

else:

lines.append(line)

file1.write("")

with open(main\_file\_path, "w") as file1:

file1.writelines(lines)

print("Дані були успішно видалені")

result\_field.configure(text=f"Дані були успішно видалені")

linecache.clearcache()

lines\_number -= 1

return "Успіх!"

except:

result\_field.configure(text=f"Неправильний формат")

return

#Функція оновлення певного елементу

def element\_update():

try:

global lines\_number

data = entry\_input\_field.get().split(" ")

row\_id = int(data[0])

value = str(data[1])

index\_value = int(element\_search(row\_id).split(" ")[1].replace("\n", ""))

with open(main\_file\_path, "r+") as file1:

lines = []

for index, line in enumerate(file1):

if index == index\_value-1:

lines.append(str(row\_id) + " " + value + "\n")

else:

lines.append(line)

file1.write("")

file1.close()

with open(main\_file\_path, "w") as file1:

file1.writelines(lines)

print("База даних була оновлена")

result\_field.configure(text="База даних була оновлена")

linecache.clearcache()

return "Успіх!"

except:

result\_field.configure(text="Неправильний формат")

#Основний код

create\_main\_file()#Створення основного файлу

lines\_number = create\_index\_file() + 1

add\_label = tkinter.Label(text="Додати елемент(формат: дані типу string без ключа)")

delete\_label = tkinter.Label(text="Видалити елемент(формат: ключ елемента)")

search\_label = tkinter.Label(text="Знайти елемент(формат: ключ елемента)")

update\_label = tkinter.Label(text="Оновити елемент(формат: ключ елемента + пробіл + оновлені дані)")

entry\_input\_field = tkinter.Entry()

add\_button = tkinter.Button(text="Додати", command=element\_add)

delete\_button = tkinter.Button(text="Видалити", command=element\_delete)

search\_button = tkinter.Button(text="Знайти", command=element\_search)

update\_button = tkinter.Button(text="Оновити", command=element\_update)

add\_button.grid(row=2, column=1)

add\_label.grid(row=2, column=0)

delete\_button.grid(row=3, column=1)

delete\_label.grid(row=3, column=0)

search\_button.grid(row=4, column=1)

search\_label.grid(row=4, column=0)

update\_button.grid(row=5, column=1)

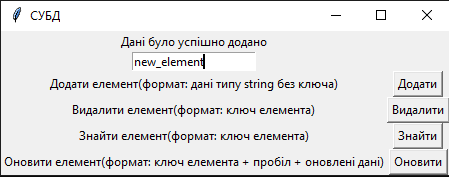
update\_label.grid(row=5, column=0)

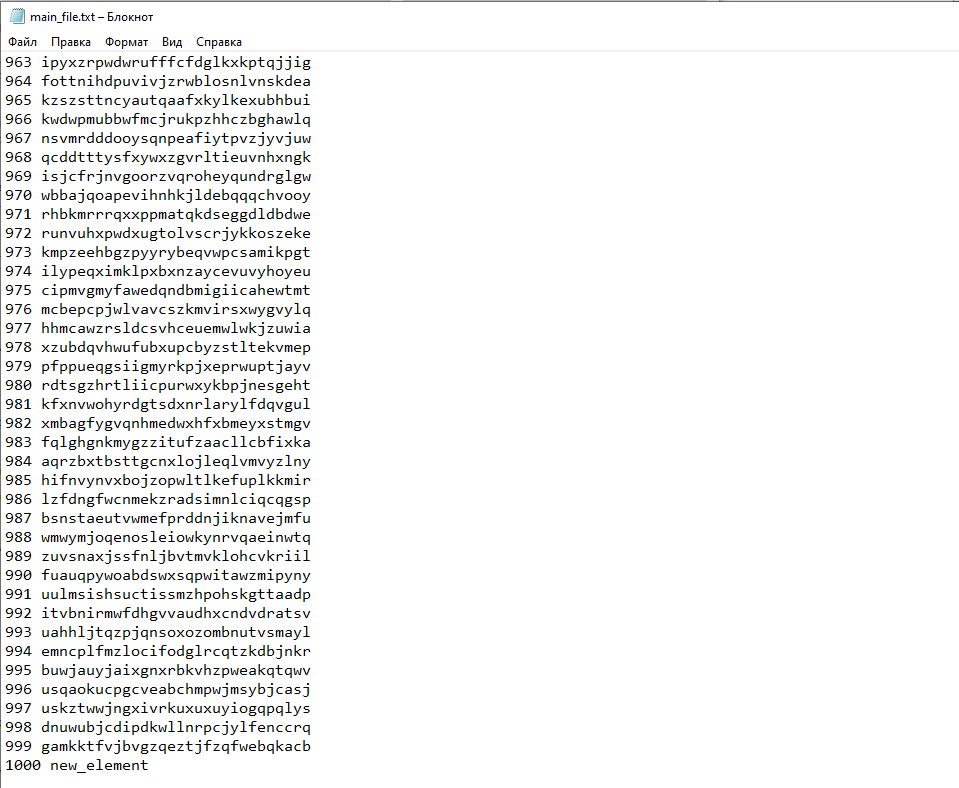
entry\_input\_field.grid(row=1, column=0)

window.mainloop()

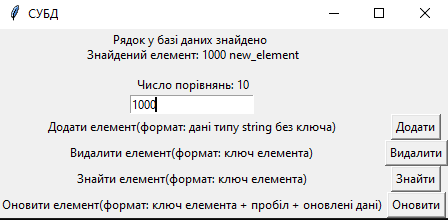
Приклади роботи програми:

Додавання алгоритму:





Пошук елементу:



Число порівнянь при спробі пошуку запису по ключу

|  |  |
| --- | --- |
| Номер спроби пошуку | Число порівнянь |
| 1 | 7 |
| 2 | 10 |
| 3 | 10 |
| 4 | 10 |
| 5 | 9 |
| 6 | 9 |
| 7 | 8 |
| 8 | 9 |
| 9 | 9 |
| 10 | 10 |
| 11 | 8 |
| 12 | 9 |
| 13 | 9 |
| 14 | 10 |
| 15 | 6 |
| 16 | 9 |
| 17 | 9 |
| 18 | 10 |
| 19 | 10 |
| 20 | 10 |

Висновок: вивчено основні підходи проектування та обробки складних структур даних, створено та протестовано власну СУБД, що працює з файлами з нещільним індексом.