Лабораторна робота № 20.
Макровизначення

1 ВИМОГИ

1.1 Розробник

- Радєвич Владислав Романович;
- студент групи КІТ – 320;
- 15.04.2021 р.

1.2 Загальне завдання

Виконати надані завдання з лабораторної роботи 20.

2 ОПИС ПРОГРАМИ

2.1 Функціональне призначення

Програма призначена для сортування структури даних заданих птахів з файлу та виведення результату у файл або на екран, використовуючи динамічні списки.

2.2 Опис логічної структури

2.2.1 Основна функція

```
int main
```

Призначення: головна функція.

Схема алгоритму функції подана на рис. 1.

Опис роботи: демонструє роботу заданого динамічного списку елементів структури та методів оперування ним, викликаючи функція для роботи з динамічним списком.

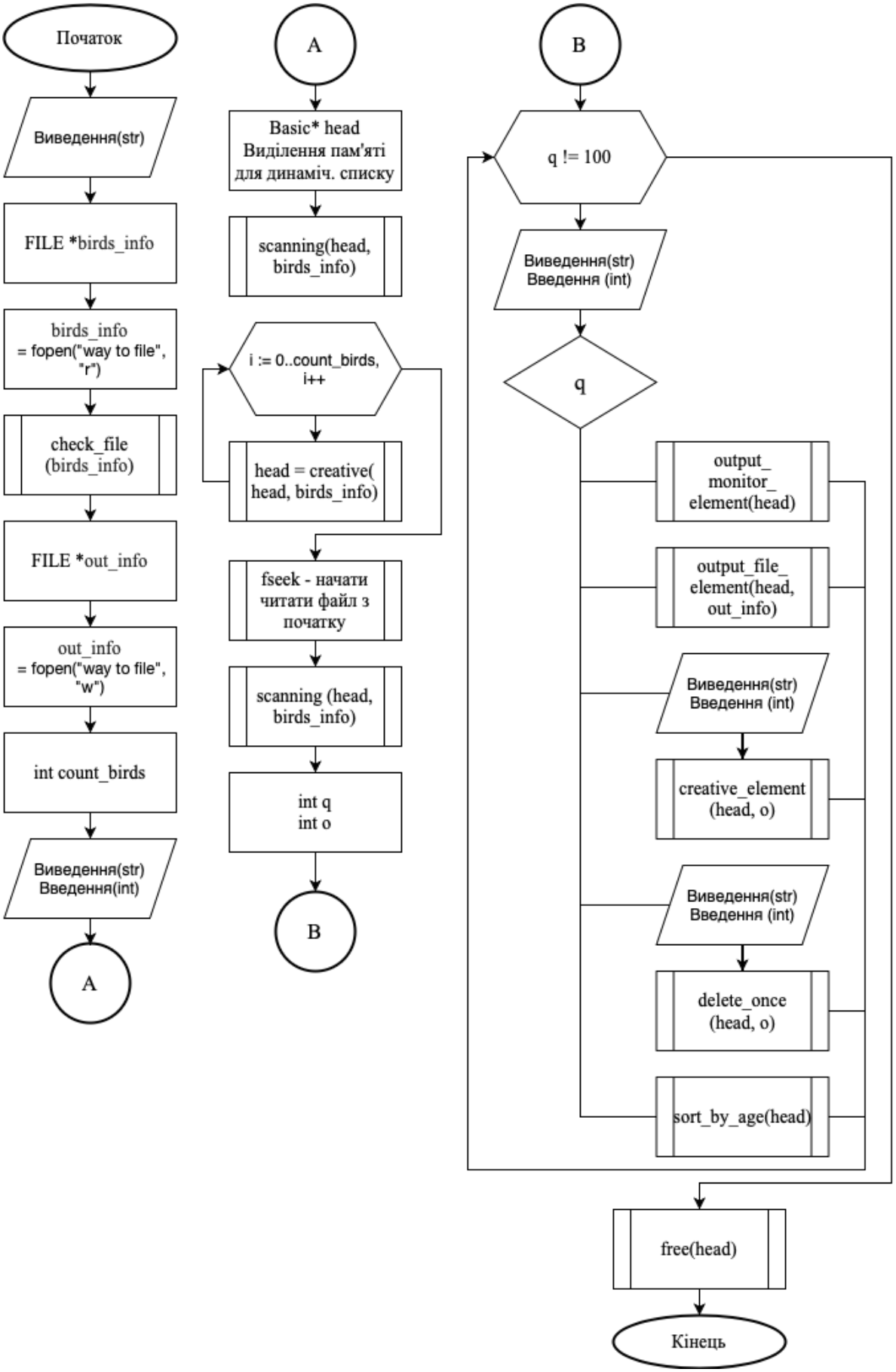

Рисунок 1 – Схема алгоритму функції main

2.2.2 Функція перевірки файлу

```
void check_file (FILE *fmatrix);
```

Призначення: перевірка місцезнаходження файлу по вказаному шляху.

Схема алгоритму функції подана на рис. 2

Опис роботи: функція перевіряє знаходження файлу з даними для структурами по заданій директорії.

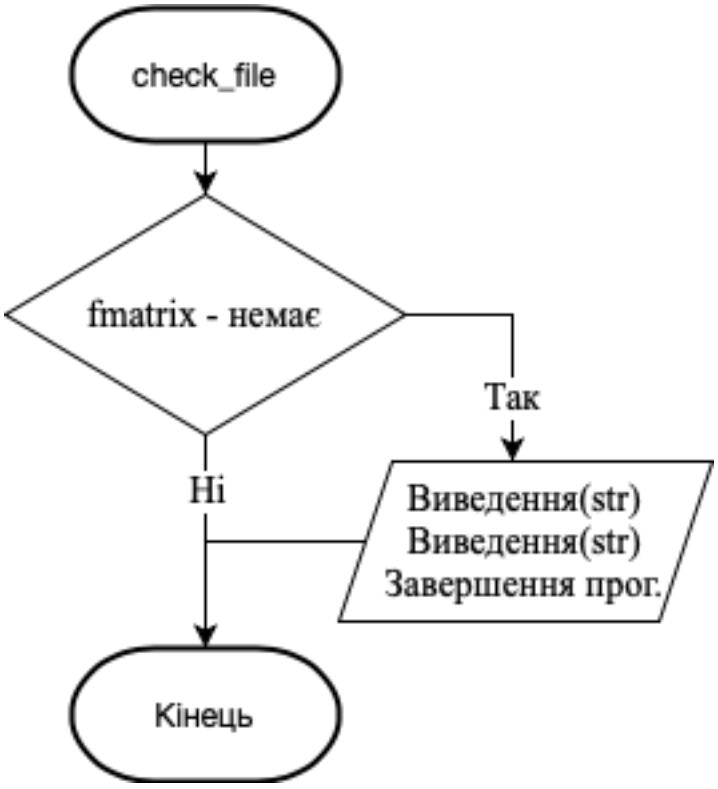

Рисунок 2 – Схема алгоритму функції check_file

2.2.3 Функція сканування даних

```
void scanning (Basic* p_bird, int count_birds, FILE *birds_info);
```

Призначення: сканування данні з файлу, в якому заздалегідь написані данні для структури.

Схема алгоритму функції подана на рис. 3.

Опис роботи: функція сканує та переносить данні з заданого файлу в задану структуру за допомогою циклу.

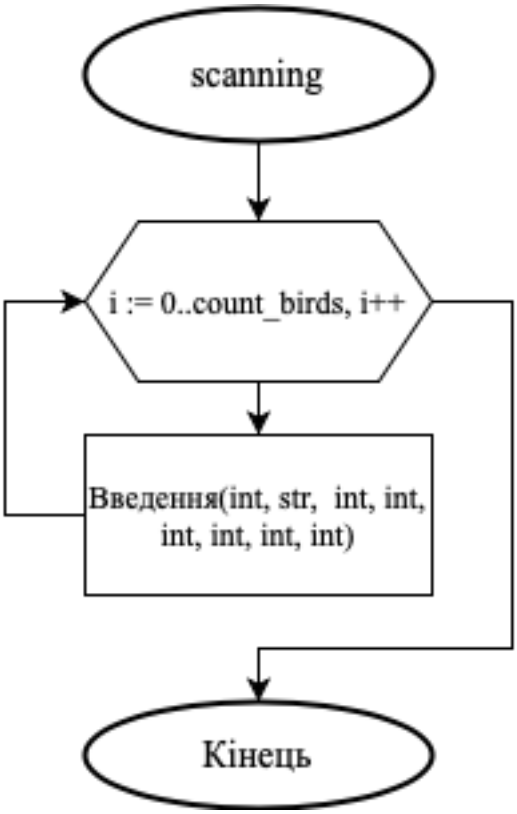

Рисунок 3 – Схема scanning

2.2.4 Функція виводу у файл

```
void output_file (Basic* p_bird, int count_birds, FILE* out_info);
```

Призначення: вивід даних у файл за певної директорією.

Схема алгоритму функції подана на рис. 4

Опис роботи: оброблення усіх полей структури та вивід усіх даних, що містяться в усіх структурах в певний файл.

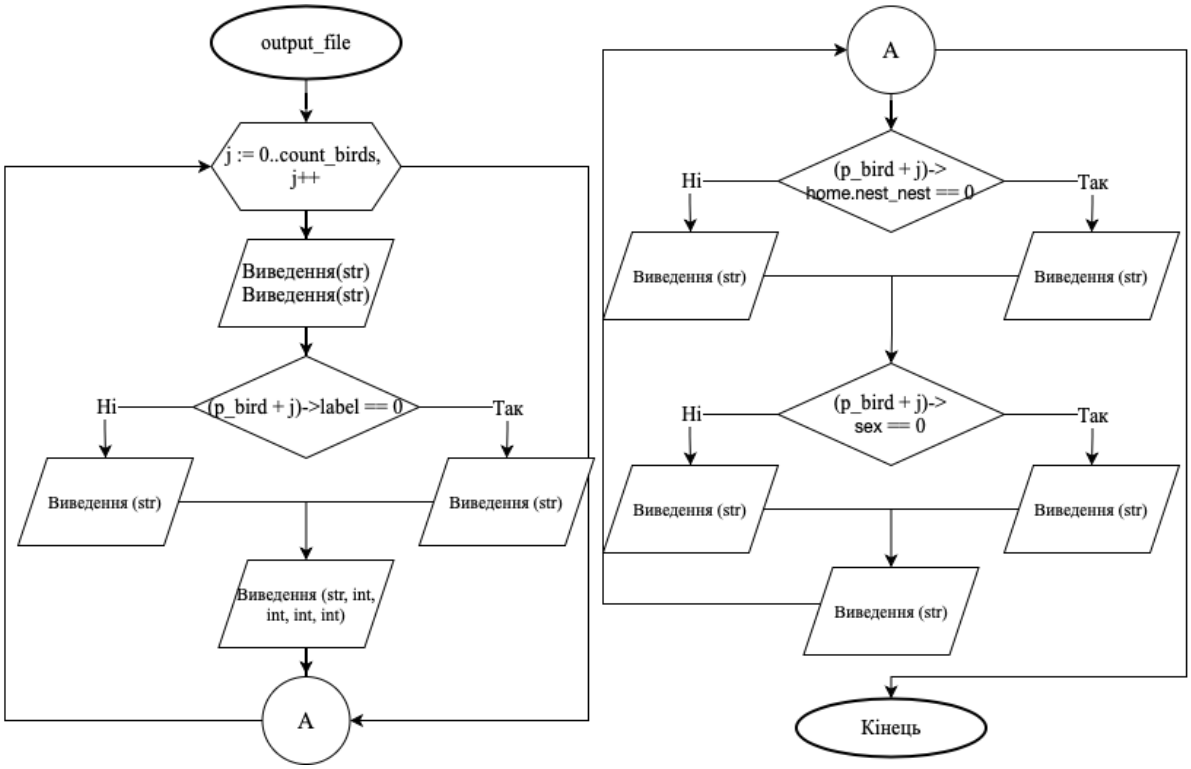

Рисунок 4 – схема алгоритму функції output_file

2.2.5 Функція виводу даних на екран

```
void output_monitor (Basic* p_bird, int count_birds);
```

Призначення: вивід окремих даних кожної структури з масиву структур на екран.

Схема алгоритму функції подана на рис. 5

Опис роботи: оброблення та вивід усіх даних кожної структури з зазначеного масиву структур.

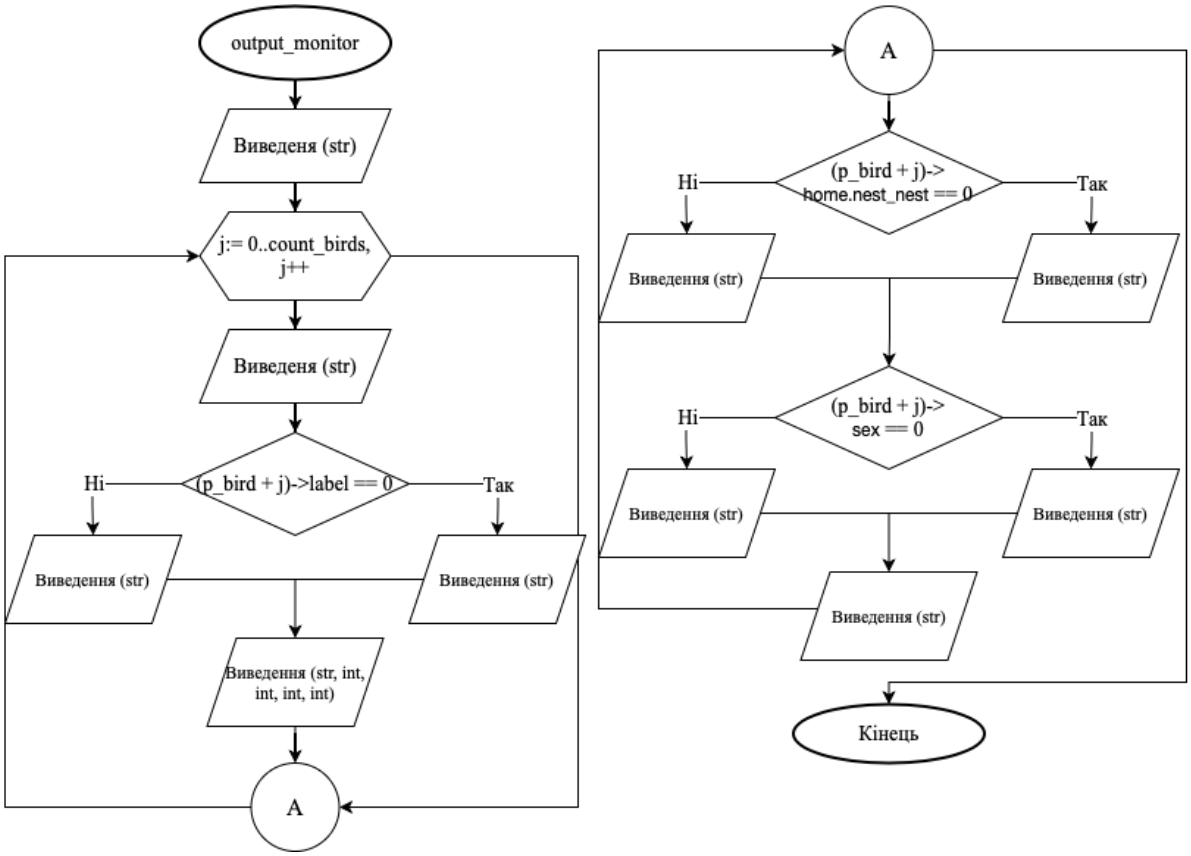

Рисунок 5 – схема алгоритму функції output_monitor

2.2.6 Функція вводу даних з файлу для списку

```
Basic* creative(Basic* head, FILE *birds_info);
```

Призначення: перепис даних з заданого файлу до динамічного списку елементів структури.

Схема алгоритму функції подана на рис. 6

Опис роботи: за допомогою функції scanning записуються дані з файлу до динамічного списку.

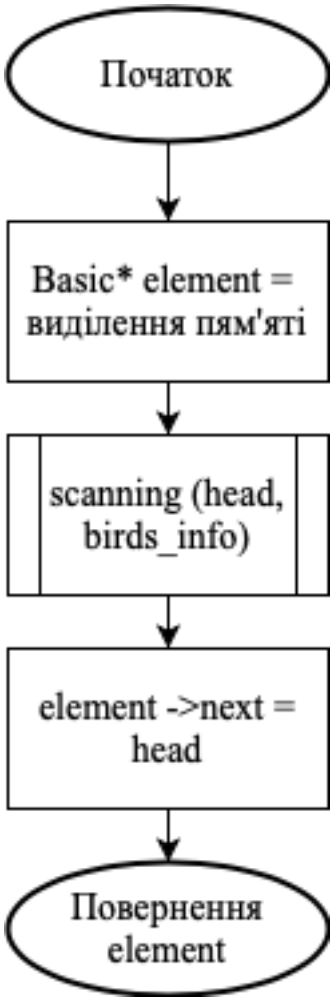

Рисунок 6 – смеха алгоритму функції creative

2.2.7 Функція створення одного елементу

```
void creative_element(Basic* head, int position);
```

Призначення: додавання елементу структури до динамічного списку.

Схема алгоритму функції подана на рис. 7

Опис роботи: функція додає, у певну позицію, до заданого динамічного списку ще один елемент структури.

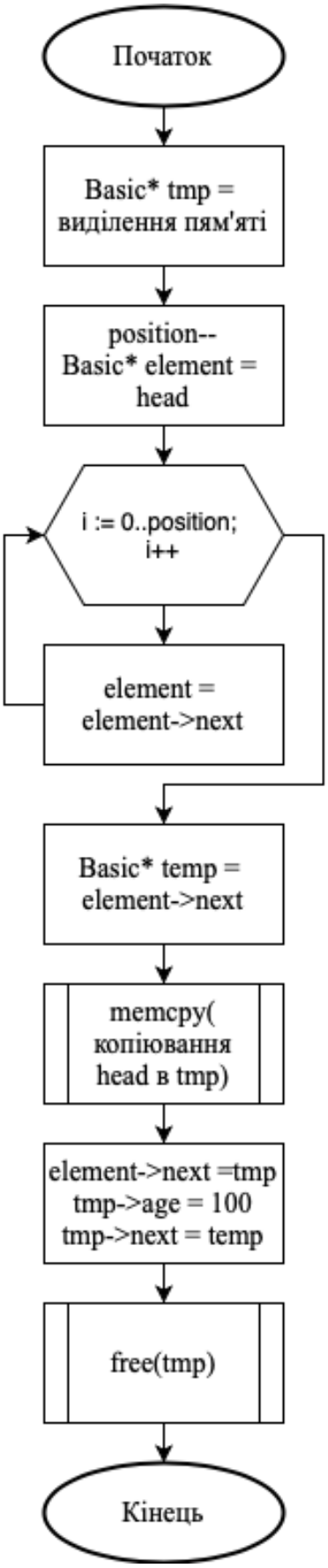

Рисунок 7 – смеха алгоритму функції creative_element

2.2.8 Функція видалення елементу

```
void delete_once (Basic* head, int n);
```

Призначення: видалення елементу структури з динамічного списку.

Схема алгоритму функції подана на рис. 8

Опис роботи: функція видаляє певний елемент структури з динамічного списку елементів структури.

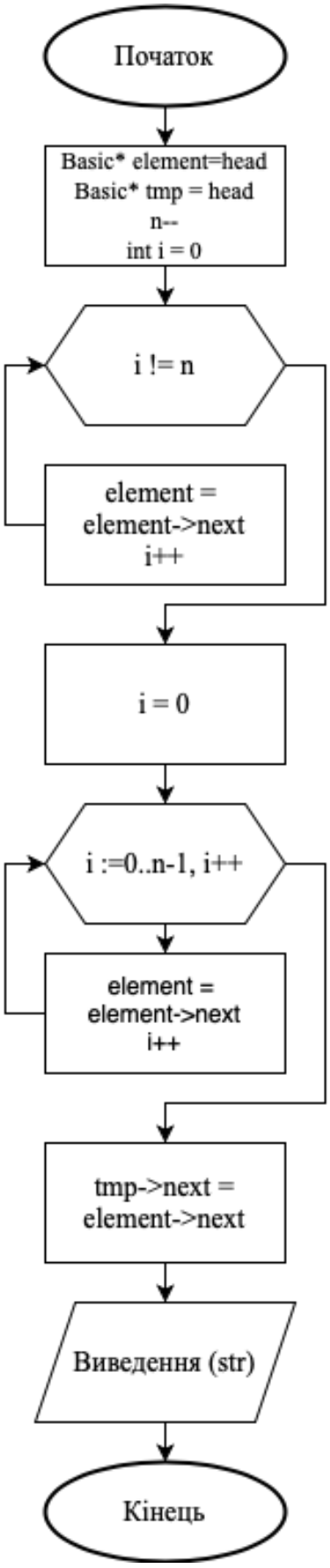

Рисунок 8 – смеха алгоритму функції delete_once

2.2.9 Функція виводу у файл списку

```
void output_file_element (Basic* head, FILE *out_info);
```

Призначення: виведення у файл динамічний список.

Схема алгоритму функції подана на рис. 9

Опис роботи: виведення у заданий файл динамічного списки елементів структури.

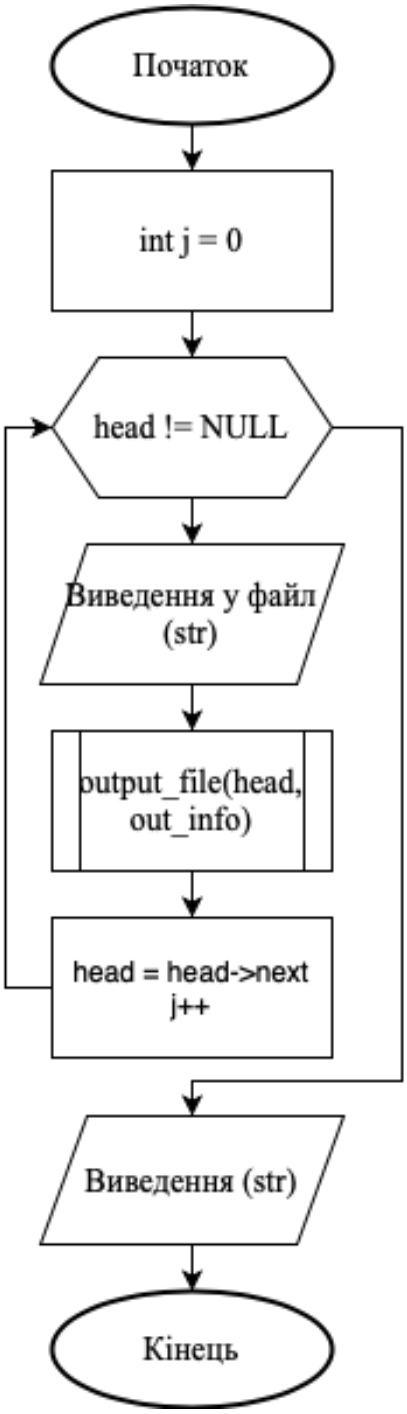

Рисунок 9 – смеха алгоритму функції output_file_element

2.2.10 Функція виводу на екран для списку

```
void output_monitor_element (Basic* head);
```

Призначення: виведення на екран динамічний список.

Схема алгоритму функції подана на рис. 10

Опис роботи: виведення на екран динамічного списки елементів структури.

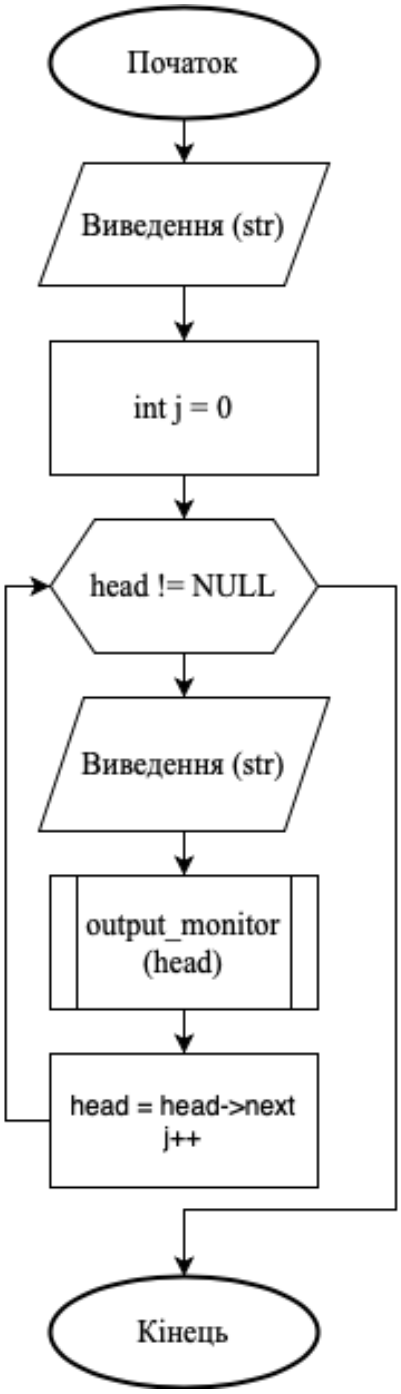

Рисунок 10 – смеха алгоритму функції output_monitor_element

2.2.11 Функція сортування для списку

```
void sort_by_age (Basic* head);
```

Призначення: сортування динамічного списку.

Схема алгоритму функції подана на рис. 11

Опис роботи: сортування за відповідним критерієм (в цьому випадку за роками життя) динамічного списку елементів структури від меншого до більшого. Сортування виконується методом «бульбашки».

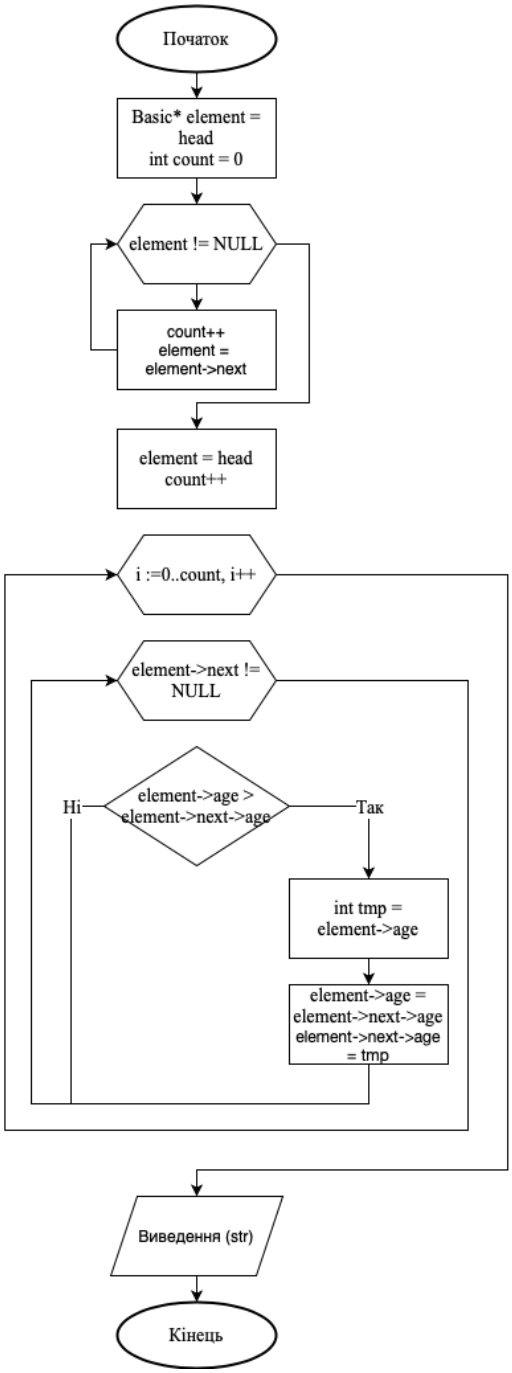

Рисунок 11 – смеха алгоритму функції sort_by_age

2.3 Структура проекту

```
.
├── 1
│   ├── src
│   │   ├── lib.c
│   │   ├── lib.h
│   │   └── main.c
│   └── test
│       └── test.c
├── 2
│   ├── src
│   │   ├── data.c
│   │   ├── data.h
│   │   ├── list.c
│   │   ├── list.h
│   │   └── main.c
│   └── test
│       └── test.c
├── Doxyfile
├── Makefile
├── README.md
└── doc
    ├── Radievych18_19.pdf
    └── assets
        ├── creative.png
        ├── creative_element.png
        ├── delete_once.png
        ├── flowchart_\ output_file.png
        ├── flowchart_check_file.png
        ├── flowchart_output_monitor.png
        ├── flowchart_scanning.png
        ├── main.png
        ├── output_file_element.png
        ├── output_monitor_element.png
        └── sort_by_age.png
```

3 ВАРІАНТИ ВИКОРИСТАННЯ

Цю програму можна використовувати за для перепису усіх зареєстрованих птахів в окремий файл на комп’ютері або для заповнення списку, заздалегідь давши про них певну інформацію.

Результат роботи з doxygen продемонстровано на рисунку 12, рисунку 13 та рисунку 14, виконання модульних тестів на рисунку 15 та демонстрація відсутності витоків пам’яті на рисунку 16..

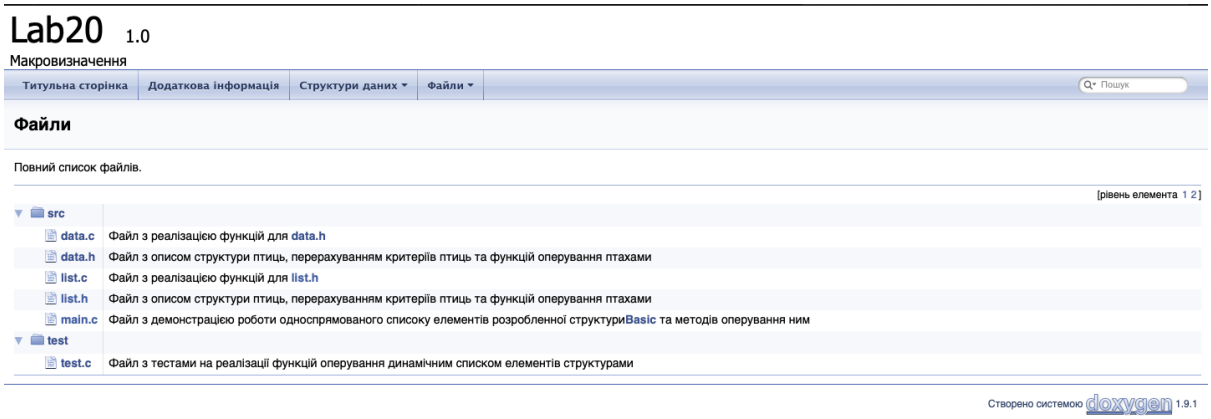

Рисунок 12 – робота з doxygen

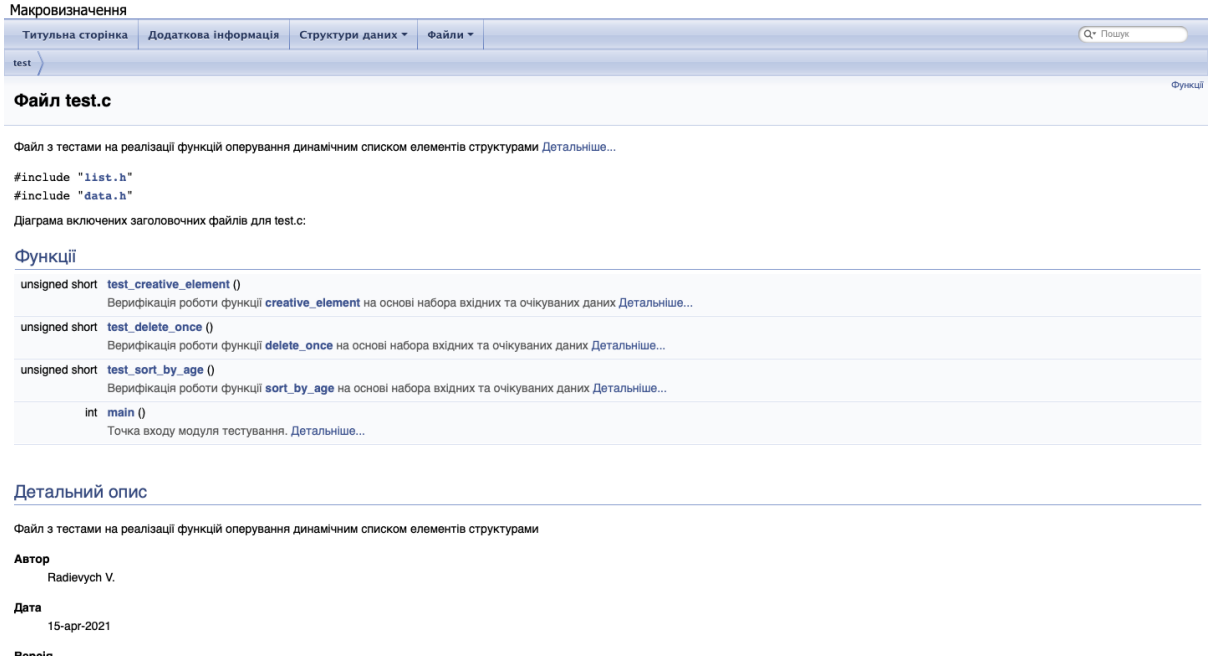

Рисунок 13 – робота з doxygen

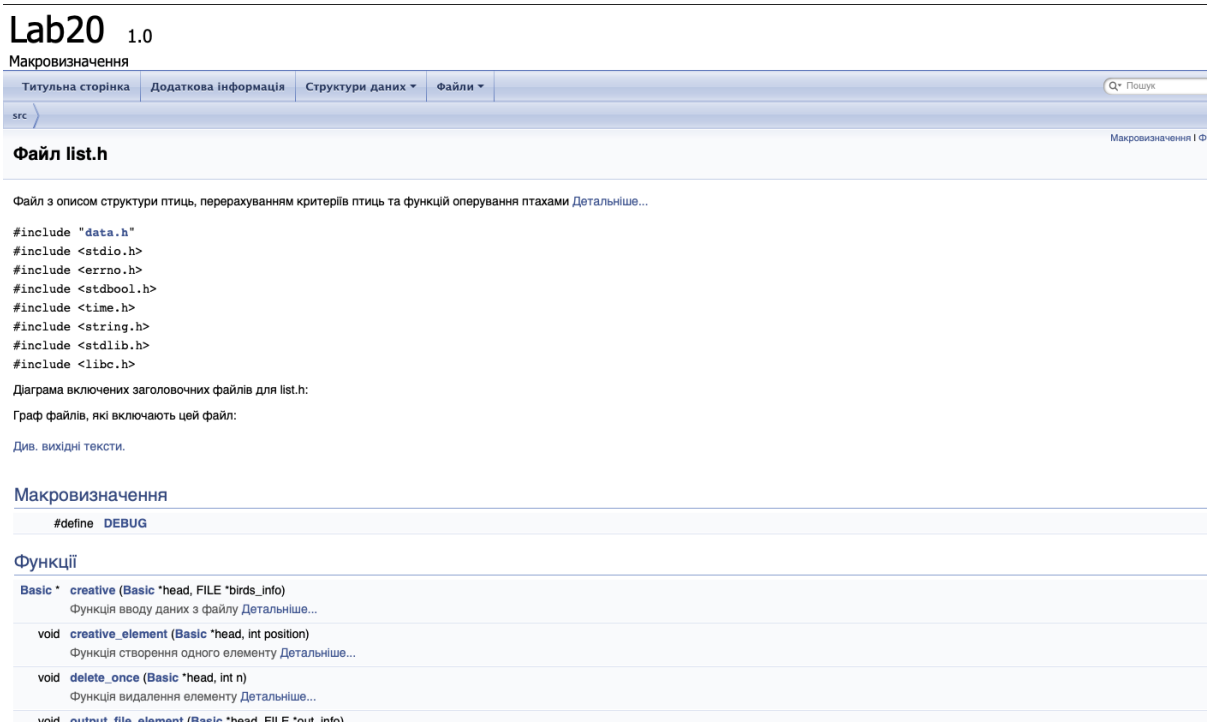

Рисунок 15 – робота з doxygen

```
whatislove@MacBook-Air-Vladislav dist % ./test1.bin
Функція: unsigned short test_creative_element()
Запуск тесту test_creative_element ...

Apr 26 2021 23:30:18
Функція: void creative_element(Basic *, int)
Елемент додано до списку!

Загальний час виконання тесту test_creative_element: 0 sec

Функція: unsigned short test_delete_once()
Запуск тесту test_delete_once ...

Apr 26 2021 23:30:18
Функція: void delete_once(Basic *, int)
Елемент видалено!

Загальний час виконання тесту test_delete_once: 0 sec

Функція: unsigned short test_sort_by_age()
Запуск тесту test_sort_by_age ...

Apr 26 2021 23:30:18
Функція: void sort_by_age(Basic *)
Дані відсортовано!

Загальний час виконання тесту test_sort_by_age: 0 sec
Модульні тести пройдено успішно!
Загальний час роботи тестів: 0.000000 sec
whatislove@MacBook-Air-Vladislav dist %
```

Рисунок 15 – робота з модульними тестами

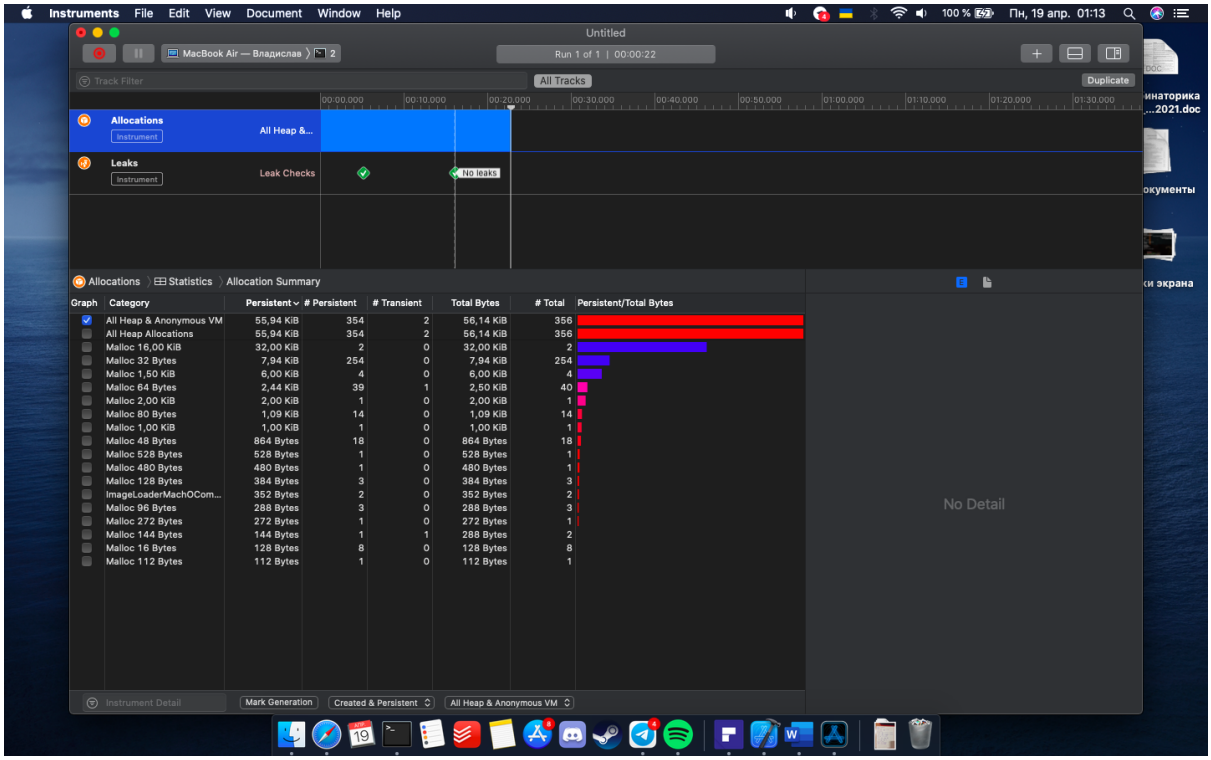

Рисунок 16 – демонстрація відсутності витоків пам’яті

4 ВИСНОВОК

При виконанні даної лабораторної роботи я закріпив набуті мною навички, створення та взаємодію з динамічними масивами та динамічними структурами.

Посилання на GitHub, де знаходяться усі програми: https://github.com/KotKHPI/Programming_Radievych