# Лабораторна робота №14. Структуровані типи даних

## 1 ВИМОГИ

### 1.1 Розробник

* Хелемендик Дмитро Олегович;
* студент групи КІТ-121д;
* 20-січ-2022.

### 1.2 Загальне завдання

З розділу “Індивідуальні завдання комплексної роботи” взяти прикладну галузь стосовно номеру варіанту за попередньо-визначеною формулою. Створити структуру, що відображає “базовий клас”.

## 2 ОПИС ПРОГРАМИ

### 2.1 Функціональне призначення:

Програма призначена для знаходження згорівших лампочок за допомогою зчитування даних з файла та сортування за заданим критерієм. Програма працює за допомогою функцій, що задекларовані *lib.h, stdlib.h, string.h, stdio.h* та *сtype.h.*

Результат зберігається у змінній *bulbs.*

Демонстрація знайдених результатів передбачає як покрокове виконання програми в режимі налагодження, так і видача даних у вікні консолі.

### 2.2 Опис логічної структури

За допомогою ключового слова *struct* описуємо лампочку, що має 8 полів – чи ввімкнена лампочка, чи перегоріла лампочка, виробник, зворотній лічильник, ватти, температура колбору світіння, форма, тип цоколю.

Опис розроблених структур і функцій наводиться на базі результатів роботи системи автодокументування *Doxygen*.

Розроблено структуру, вміст якої подано нижче.

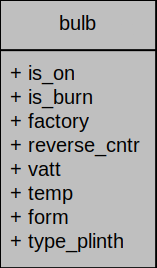


Рисунок 1 — поля структури bulb

#### Функція заповнення структури

int write\_to\_struct(char \*s, const char \*delim, struct bulb \*e);

*Призначення*: заповнить структуру даними.

*Опис роботи*: функція розбиває строку на частини та записує дані у структуру.

**Аргументи:**

- *s* — строка;

- *delim* — роздільник;

- *e* — показчик на структуру лампочки.

#### Функція підрахунку кількості строк

int lines\_count(char \*argv[]);

*Призначення*: отримання кількості строк для визначення кількості лампочок.

*Опис роботи*: функція рахує кількість строк у файлі.

**Аргументи:**

- *argv* - масив з аргументи(введеними користувачем).

#### Функція знаходження найдовшої строки

int longest\_line(char \*argv[]);

*Призначення*: знаходження найдовшої строки для визначення розміру буферу.

*Опис роботи*: функція знаходе кількість символів у найдовшій строці.

**Аргументи**:

- *argv* - масив з аргументи(введеними користувачем).

#### Функція заповнення структур та отримання критерія для сортування

int get\_struct\_and\_type(struct bulb \*bulbs, int count\_bulbs, char \*argv[], char \*type\_for\_sort);

*Призначення*: заповнення структур та отримання критерія для сортування.

*Опис роботи*: функція зчитує дані з файла, заповнює масив структур та записує у показчик критерій для сортування за його наявності.

**Аргументи**:

- *bulbs* - показчик на структуру;

- *count\_bulbs -* кількість лампочок;

- *argv* - масив з аргументи(введеними користувачем);

- type\_for\_sort- показчик для критерія сортування.

#### Функція знаходження згорівшої лампочки

int is\_burn\_bulbs(struct bulb \*bulbs, int count\_bulbs);

*Призначення*: визначити кількість згорівших лампочок.

*Опис роботи*: функція знаходе номера згорівших лампочок та записує їх у показчик.

**Аргументи**:

- bulbs- показчик на структуру;

- *count\_bulbs* - кількість лампочок.

#### Функція запису номера згорівшої лампочки

void find\_burn\_bulbs(struct bulb \*bulbs, int count\_bulbs, int \*burn\_bulbs);

*Призначення*: записати номер згорівшої лампочки у показчик.

*Опис роботи*: функція знаходе номера згорівших лампочок та записує їх у показчик.

**Аргументи**:

- *bulbs* - показчик на структуру;

- *count\_bulbs -* кількість лампочок;

- *burn\_bulbs* - показчик для зберігання номерів згорівших лампочок.

**Функція друку результату у консоль**

void print\_res\_screen(struct bulb \*bulbs, int count\_bulbs, int \*burn\_bulbs, int num\_burn\_bulbs, char \*type\_for\_sort);

*Призначення*: друк результату у вікно консолі.

*Опис роботи*: функція друкує результат у вікно консолі.

**Аргументи**:

- *bulbs* - показчик на структуру;

- *count\_bulbs -* кількість лампочок;

- *burn\_bulbs* - показчик для зберігання номерів згорівших лампочок;

- *num\_burn\_bulbs* - кількість згорівших лампочок;

- type\_for\_sort- показчик для критерія сортування.

**Функція друку результату у файл**

void print\_res\_file(struct bulb \*bulbs, int count\_bulbs, int \*burn\_bulbs, int num\_burn\_bulbs, char \*type\_for\_sort, char \*argv[]);

*Призначення*: друк результату у файл.

*Опис роботи*: функція друкує результат у файл.

**Аргументи**:

- *bulbs* - показчик на структуру;

- *count\_bulbs -* кількість лампочок;

- burn\_bulbs *-* показчик для зберігання номерів згорівших лампочок;

- num\_burn\_bulbs *-* кількість згорівших лампочок;

- type\_for\_sort- показчик для критерія сортування;

- *argv* - масив з аргументи(введеними користувачем).

#### Основна функція

int main()

*Призначення*: головна функція.

*Опис роботи*:

- знаходю кількість лампочок за допомогою функції lines\_count та зберігаю дані у змінній count\_bulbs;

- виділяю пам'ять для масиву символів, у якому буде зберігаться критерій для сортування;

- виділяю пам'ять для структури розміром count\_bulbs;

- далі зчитую дані з файлу, записую критерій(якщо він є), заповнюю структуру шляхом виклику функції get\_struct\_and\_type;

- знаходю кількість згорівших лампочок функцією is\_burn\_bulbs та зберігаю у змінній num\_burn\_bulbs;

- виділяю пам'ять для масиву з номерами згорівших лампочок;

- знаходю номера згорівших лампочок та зберігаю їх в показчику burn\_bulbs;

- друкую результат у вікно консолі за допомогою функції print\_res\_screen;

- якщо введений третий аргумент - друкую результат у файл шляхом виклику функції print\_res\_file;

- звільнюю пам'ять;

- успішний код повернення з програми (0).

#### Структура проекту:

└── lab14

├── assets

│ └── input.txt

├── doc

│ ├── assets

│ │ └── bulb\_fields.png

│ ├── lab14.docx

│ └── lab14.md

│ └── lab14.pdf

├── Doxyfile

├── Makefile

├── README.md

├── src

│ ├── lib.c

│ ├── lib.h

│ └── main.c

└── test

└── test.c

#### 2.3 Важливі фрагменти програми

#### Запис до структури

char \*p = strtok(s, delim); // розбиваю строку на частини

// виконую запис в структуру

if (!p || !strncpy(e->is\_on, p, sizeof(e->is\_on) - 1))

return 1;

if (!(p = strtok(NULL, delim)) || !strncpy(e->is\_burn, p, sizeof(e->is\_burn) - 1))

return 1;

if (!(p = strtok(NULL, delim)) || !strncpy(e->factory, p, sizeof(e->factory) - 1))

return 1;

if (!(p = strtok(NULL, delim)) || sscanf(p, "%d", &(e->reverse\_cntr)) != 1)

return 1;

if (!(p = strtok(NULL, delim)) || sscanf(p, "%d", &(e->vatt)) != 1)

return 1;

if (!(p = strtok(NULL, delim)) || sscanf(p, "%d", &(e->temp)) != 1)

return 1;

if (!(p = strtok(NULL, delim)) || !strncpy(e->form, p, sizeof(e->form) - 1))

return 1;

if (!(p = strtok(NULL, delim)) || !strncpy(e->type\_plinth, p, sizeof(e->type\_plinth) - 1))

return 1;

return 0;

#### Заповнення структур та отримання критерію сортування

fgets(type\_for\_sort, 50, f); // знаходю критерій для сортування якщо він є

char \*buff = (char \*)malloc((unsigned int)size\_buff + 1);

for (int i = 0; i < count\_bulbs; i++) {

fgets(buff, size\_buff + 1, f);

if (write\_to\_struct(buff, ",", &bulbs[i])) // заповнюю масив структур

fprintf(stderr, "Error!\n");

}

#### 3. Варіанти використання

Для демонстрації результатів кожної задачі використовується:

- покрокове виконання програми в утиліті lldb;

- виконання програми у вікні консолі.

**Варіант використання 1**: послідовність дій для запуску програми у режимі відлагодження:

- запустити програму у відлагоднику lldb з трьома аргументами(перший - бінарний файл, другий - путь до файла, третій - файл, у який буде записаний результат);

- поставити точку зупинки на функції main (строка з `return 0;`);

- запустити програму;

- подивитися результат виконання програми.

Також результат зберігається у заданому файлі(зараз це output.txt).

dima@dima-VirtualBox:~/dev/programing-khelemendyk/lab14/dist$ lldb main.bin "/home/dima/dev/programing-khelemendyk/lab14/assets/input.txt" "output.txt"

(lldb) target create "main.bin"

Current executable set to '/home/dima/dev/programing-khelemendyk/lab14/dist/main.bin' (x86\_64).

(lldb) settings set -- target.run-args "/home/dima/dev/programing-khelemendyk/lab14/assets/input.txt" "output.txt"

(lldb) b 69

Breakpoint 1: where = main.bin`main + 233 at main.c:69:2, address = 0x0000000000402a49

(lldb) r

Process 5252 launched: '/home/dima/dev/programing-khelemendyk/lab14/dist/main.bin' (x86\_64)

Ваші лампочки:

Лампочка 1: yes, no, TOV Roga ta koputa, 20, 15, 1800, Globe, E40

Лампочка 2: no, yes, TOV Roga ta oleni, 40, 30, 3600, Circle, E20

Лампочка 3: no, no, TOV Koputa, 30, 25, 1900, Pear, E27

Лампочка 4: yes, yes, TOV Romashka, 25, 150, 2500, Candle, E50

Лампочка 5: yes, yes, TOV Kapysta, 19, 10, 1400, Ogive, E30

Перегорівші лампочки:

Лампочка 2: no, yes, TOV Roga ta oleni, 40, 30, 3600, Circle, E20

Лампочка 4: yes, yes, TOV Romashka, 25, 150, 2500, Candle, E50

Лампочка 5: yes, yes, TOV Kapysta, 19, 10, 1400, Ogive, E30

Заданий критерій для сортування: Виробник лампочки

Лампочка 1: TOV Roga ta koputa

Лампочка 2: TOV Roga ta oleni

Лампочка 3: TOV Koputa

Лампочка 4: TOV Romashka

Лампочка 5: TOV Kapysta

Process 5252 stopped

\* thread #1, name = 'main.bin', stop reason = breakpoint 1.1

frame #0: 0x0000000000402a49 main.bin`main(argc=3, argv=0x00007fffffffe048) at main.c:69:2

66 free(type\_for\_sort);

67 free(burn\_bulbs);

68 }

-> 69 return 0;

70 }

**Варіант використання 2**: запуск програми у вікні консолі:

- запустити програму у консолі з трьома аргументами;

- подивитись результат програми.

Також результат зберігається у заданому файлі(зараз це output.txt).

dima@dima-VirtualBox:~/dev/programing-khelemendyk/lab14/dist$ ./main.bin "/home/dima/dev/programing-khelemendyk/lab14/assets/input.txt" "output.txt"

Ваші лампочки:

Лампочка 1: yes, no, TOV Roga ta koputa, 20, 15, 1800, Globe, E40

Лампочка 2: no, yes, TOV Roga ta oleni, 40, 30, 3600, Circle, E20

Лампочка 3: no, no, TOV Koputa, 30, 25, 1900, Pear, E27

Лампочка 4: yes, yes, TOV Romashka, 25, 150, 2500, Candle, E50

Лампочка 5: yes, yes, TOV Kapysta, 19, 10, 1400, Ogive, E30

Перегорівші лампочки:

Лампочка 2: no, yes, TOV Roga ta oleni, 40, 30, 3600, Circle, E20

Лампочка 4: yes, yes, TOV Romashka, 25, 150, 2500, Candle, E50

Лампочка 5: yes, yes, TOV Kapysta, 19, 10, 1400, Ogive, E30

Заданий критерій для сортування: Виробник лампочки

Лампочка 1: TOV Roga ta koputa

Лампочка 2: TOV Roga ta oleni

Лампочка 3: TOV Koputa

Лампочка 4: TOV Romashka

Лампочка 5: TOV Kapysta

## ВИСНОВКИ

При виконанні даної лабораторної роботи було набуто практичного досвіду у взаємодії з структуровані типами даних.