**Лабораторна робота №22.**

**Регулярні вирази.**

* **Вимоги**
* **Розробник**
* Гладков Костянтинт Сергiович
* Студент групи КІТ-320;
* 28-feb-2021.
* **Основне завдання**



* **Опис роботи**

Програма працює шляхом введення даних від користувача з файлу і їх подальшого аналізу з виведенням на екран та в файл результату. Також в кінці відбувається тест. Але якщо використовується debug, то також буде виведена інформацію про час роботи програми, дату, а також ім'я функції. Також програма має в собі регулярні вирази, які використовується для перевірки крітеріяем згодна завданням

* **Функціональне призначення**

Дана програма працює за допомогою використання динамічної пам'яті, а точніше маніпуляцією динами. пам'яті, також використовується файли. Для макросу використовується проста "функція" в якості макросу. А в якості перевірки заданим крітеріяем використовуються регулярні вирази.

* **Опис логічної структури**

Дана програма може бути використана для створення динамічного списку, двонаправленого, який буде зберігати собі дані і буде легкий для підправлені (видалення і введення даних). Також ця програма буде виводити ім'я функції і дату за певних умов і використовувати для перевірки критерієм регулярні вирази.

Функція main містить в собі виклик функцій, а також корисних властивостей для функцій

Функція append збільшує розмір списку

Функція push вставляє в кінець елемент

Функція insertAfter вставляє елемент після певного індексу

Функція printList показує наш список

Функція bubbleSort сортровка бульбашкою

Функція swap служить для сортування

Функція test\_append перевіряє функцію

Функція print\_node виводить конкретний елемент

Функція insertAtTheBegin вставляє елемент в початок

Функцiя printList\_reverse показує наш список в зворотному порядку

Приклад використання перевірки крітеріяем в програмі:



Ось сам макрос:





рис. 1 - insertAtTheBegin



рис. 2 - printList\_reverse



рис. 3 - print\_node



рис. 4 - printList

* **Важливі елементи програми**

struct Agency\* new\_node = malloc(sizeof(struct Agency));

struct Agency\* last = \*head\_ref;

strcpy(new\_node->agency\_boss.first\_name, "x");

strcpy(new\_node->agency\_boss.second\_name, "x");

strcpy(new\_node->agency\_boss.email, "x");

strcpy(new\_node->agency\_name, "x");

strcpy(new\_node->servicing\_city, "x");

new\_node->agency\_age = 5;

new\_node->have\_weekends = 0;

new\_node->next = NULL;

if (\*head\_ref == NULL) {

new\_node->prev = NULL;

\*head\_ref = new\_node;

return;

}

while (last->next != NULL)

last = last->next;

last->next = new\_node;

new\_node->prev = last;

struct Node \*temp = start;

printf("\n");

while (temp!=NULL)

{

printf("%d ", temp->data);

temp = temp->next;

}

#define TASK() time\_t time1 = time(NULL);\

printf("Function %s. Time %s ", \_\_func\_\_, ctime(&time1))

#define TIME\_CHECK(start,end) printf("Program worked for %d seconds\n",(end)-(start))

struct Node \*ptr1 = malloc(sizeof(struct Node));

ptr1->data = data;

ptr1->next = \*start\_ref;

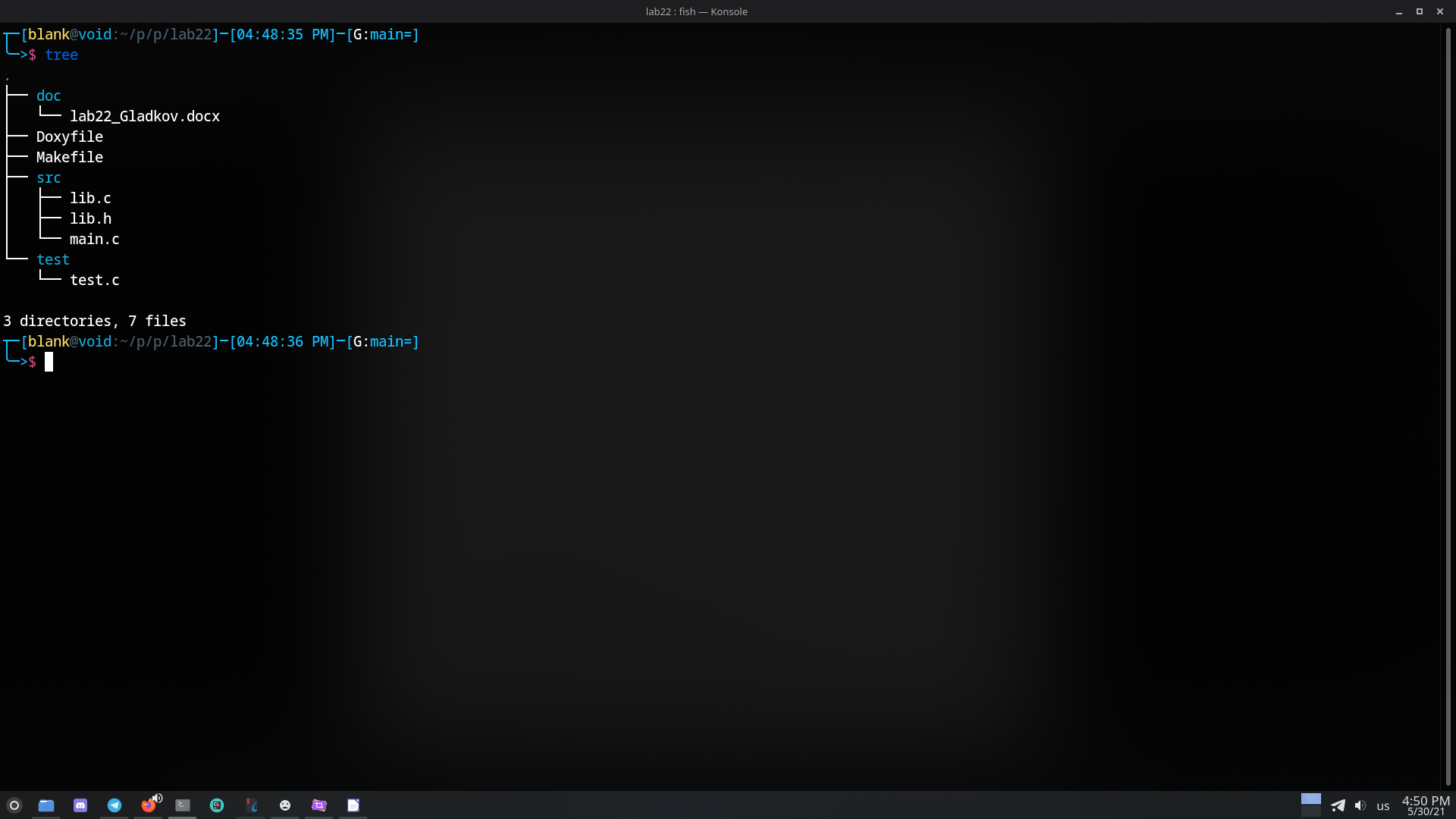
if (\*start\_ref != NULL){

(\*start\_ref)->prev = ptr1;

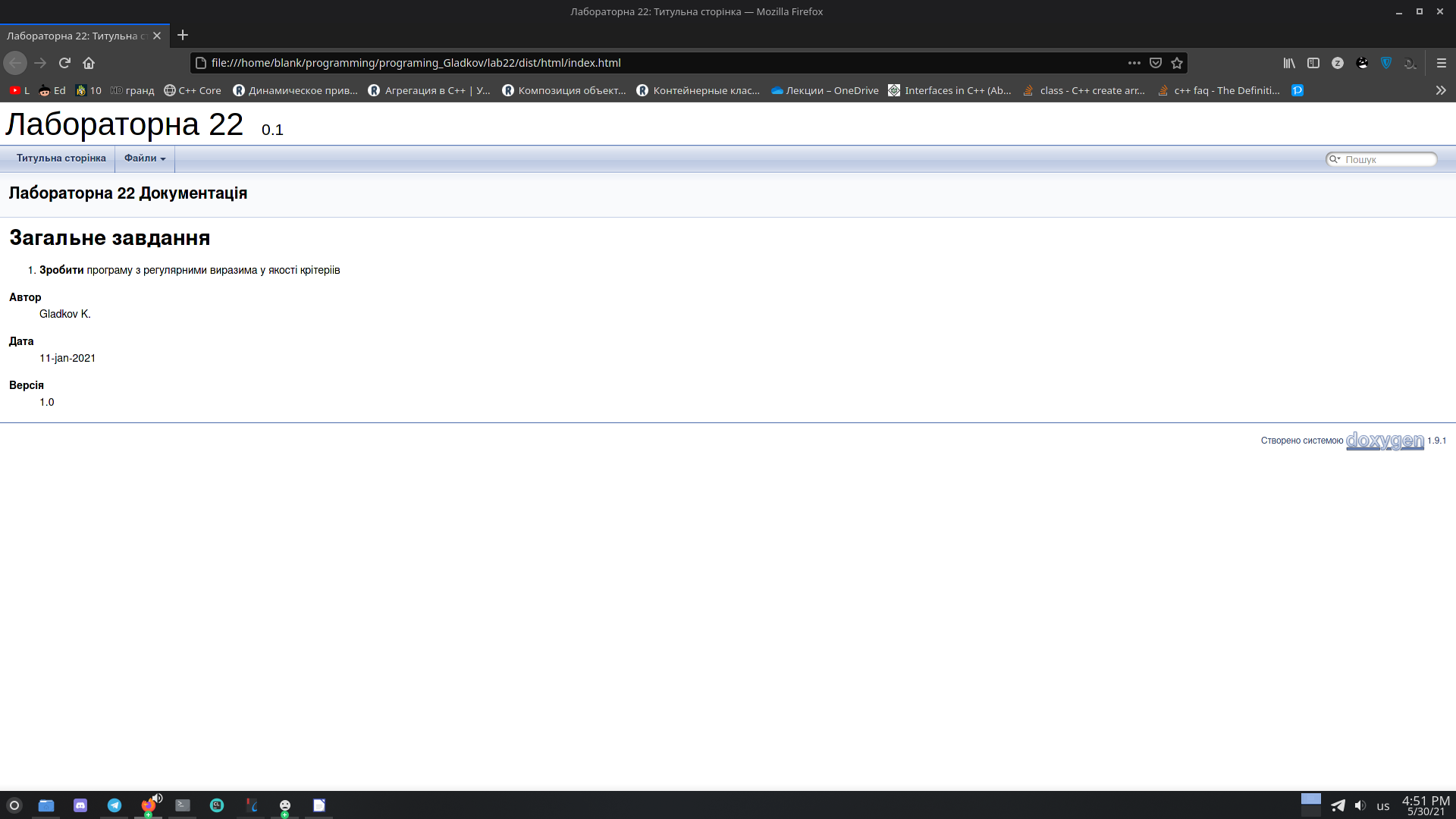
}

\*start\_ref = ptr1;

* **Структура**

****

* **Doxygen**

****

* **Варіанти використання**

Дана програма може бути використана для роботи з динамічними списками. Також ця програма буде виводити ім'я функції і дату за певних умов. І вона прозводит перевірку крітеріяем.

**Висновки**

У даній лабораторній роботі був пріобрітен досвід роботи з динамічними списками i роботою з макросами. Також був придбаний досвід роботи з регулярними виразом і складанням з їх допомогою критерій, за яким працює програма.