

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ  
«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА ПРОГРАМУВАННЯ»

«Програмування ч.2»

*Звіт з лабораторної роботи №22*  
*Тема: «Регулярні вирази»*

Виконав:  
ст. гр. КІТ-120А  
Старовойтов Н.А.

Перевірив:  
Челак В.В.

Харків – 2021

**Мета:** Отримати навички роботи із регулярними виразами.

### **Індивідуальне завдання**

Робота на оцінку “відмінно”.

Поширити попередню лабораторну роботу таким чином:

1. При введенні інформації про “базовий тип” (нема різниці, чи з клавіатури, чи з файлу), організувати перевірку відповідності таким критеріям з використанням регулярних виразів:
  - Можна вводити тільки кириличні символи, латинські символи, цифри, пропуски, розділові знаки;
  - Не повинно бути пропусків та розділових знаків, які повторюються;
  - Перше слово не повинно починатися з маленького символу;
2. У клас-список додати метод, що виводить на екран список усіх об’єктів, які мають одне або більше полів з щонайменше двома словами (перевірку організувати за допомогою регулярних виразів).

### **Опис програми**

Функція `Regex_Check` проводить перевірку кожного елементу структури на відповідність заданому регулярному виразу. Приймає структуру, повертає результат перевірки. Текст функції:

```
bool Regex_Check(struct Bird *bird){
    bool result = true;
    regex_t regex;
    regcomp(&regex, "[0-1]$", 0);
    char boool[2];
    sprintf(boool, "%d", bird->LOTR);
    if (regexec(&regex, boool, 0, NULL, 0) != 0) result = false;

    regcomp(&regex, "^?[A-ZА-Я]([A-ZА-Яа-за-я]+ ?)+$", REG_EXTENDED);
    if (regexec(&regex, bird->name, 0, NULL, 0) != 0) result = false;

    char num[5];
    sprintf(num, "%d", bird->age);
    regcomp(&regex, "[0-9]{1,3}$", REG_EXTENDED);
    if (regexec(&regex, num, 0, NULL, 0) != 0) result = false;
```

```

    sprintf(num, "%d", bird->home.space);
    regcomp(&regex, "[0-9]{1,4}$", REG_EXTENDED);
    if (regexec(&regex, num, 0, NULL, 0) != 0) result = false;

    sprintf(num, "%d", bird->home.height);
    regcomp(&regex, "[0-9]{1,4}$", REG_EXTENDED);
    if (regexec(&regex, num, 0, NULL, 0) != 0) result = false;

    sprintf(boool, "%d", bird->home.count_of_feeders);
    regcomp(&regex, "[0-9]$", REG_EXTENDED);
    if (regexec(&regex, boool, 0, NULL, 0) != 0) result = false;

    regcomp(&regex, "^[0-1]$", 0);
    sprintf(boool, "%d", bird->home.if_nest);
    if (regexec(&regex, boool, 0, NULL, 0) != 0) result = false;

    regcomp(&regex, "^[0-1]$", 0);
    sprintf(boool, "%d", bird->enumSex);
    if (regexec(&regex, boool, 0, NULL, 0) != 0) result = false;

    return result;
}

```

Функція `Print_Regex_Two_Words` проводить перевірку елементу структури «name» (тобто імені/назви птаха) на відповідність заданому регулярному виразу, а саме: назва повинна складатись щонайменше з двох слів. Приймає структуру, якщо результат перевірки позитивний, виводить вміст структури на екран. Текст функції:

```

void Print_Regex_Two_Words(struct List * list) {
    regex_t regex;
    for (struct Bird *bird = list->head; bird != NULL; bird = bird->next) {
        regcomp(&regex, "[A-ZА-Яа-za-я]+ [A-ZА-Яа-za-я]+", REG_EXTENDED);
        if (regexec(&regex, bird->name, 0, NULL, 0) == 0) {
            Print_bird(bird);
        }
    }
}

```

## Схеми алгоритмів функцій

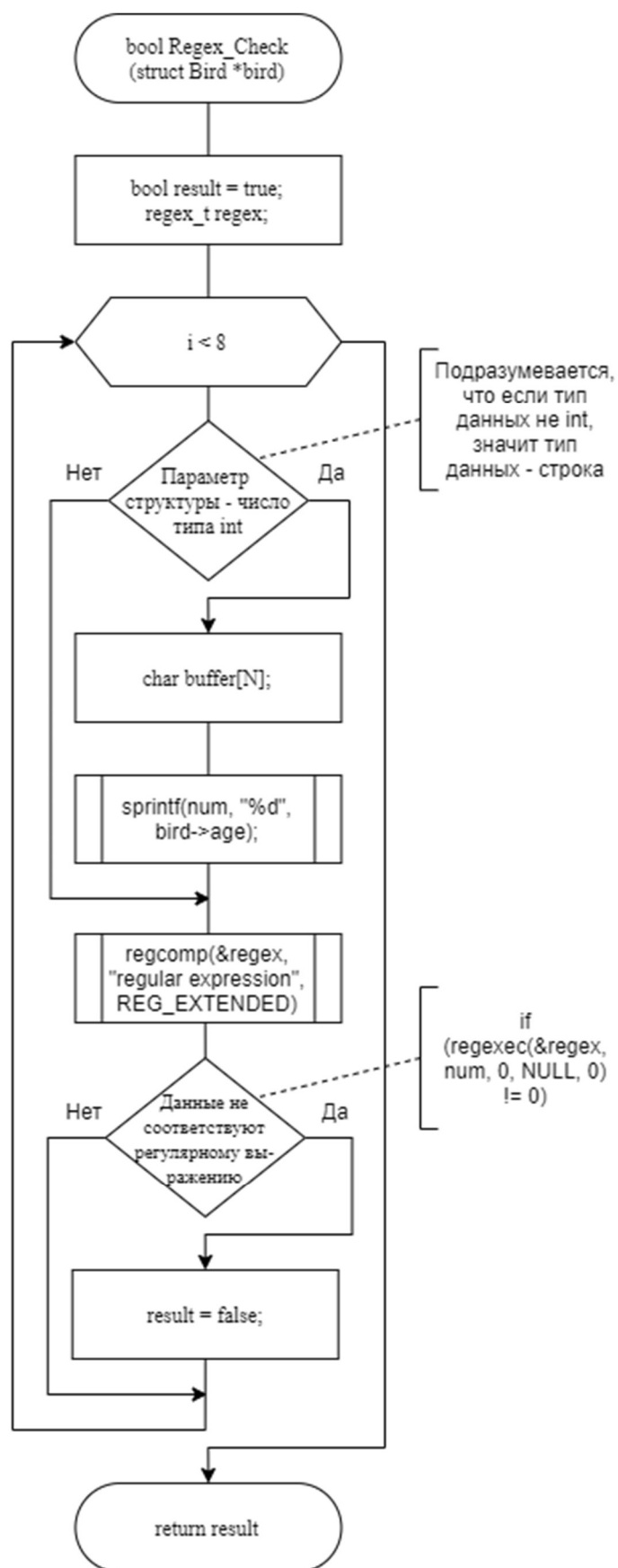


Рисунок 1 — Блок-схема функції Regex\_Check

## Текст программы

```
#include "../Library/library.h"

#define DEBUG

int main() {
    system("chcp 65001");

    #ifdef DEBUG
        Print_Date();
    #endif

    printf("\n");
    struct List *list = New_list();

    struct Bird struct_paste_bird = {0, "SOME_BIRD", 27, 250, 40, 2, 0, 1};
    struct Bird *paste_bird = malloc(sizeof(struct Bird));
    *paste_bird = struct_paste_bird;

    Read_list_from_file(list);
    int start1, finish1;

    while (true) {
        int action = 0;
        printf("\n|-----|\n");
        printf(" Выберите действие: \n");
        printf("\t1. Вывести список на экран\n");
        printf("\t2. Вывести список в файл\n");
        printf("\t3. Вставить новый элемент в список\n");
        printf("\t4. Удалить элемент из списка\n");
        printf("\t5. Найти в списке элемент\n");
        printf("\t6. Сортировать список по возрасту птицы\n");
        printf("\t7. Вывести на экран птиц с именем, содержащим два или больше слова\n");
        printf("\t8. Выход\n");
        printf("|-----|\n\n");
        printf(" --> ");
        scanf("%d", &action);
        switch (action) {
            case 1:
                #ifdef DEBUG
                    printf("\nВыполняется функция Show_list");
                    printf("\nСписок:\n\n");
                    start1 = clock();
                    Show_list(list);
                    finish1 = clock();
                    finish1 -= start1;
                    printf("Время выполнения функции: %dms%s", finish1, "\n");
                #else
                    printf("\nСписок:\n\n");
                    Show_list(list);
                #endif
                break;
            case 2:
                #ifdef DEBUG
                    printf("\nВыполняется функция Print_list_in_file");
                    start1 = clock();
                    Print_list_in_file(list);
                    finish1 = clock();
                    finish1 -= start1;
                    printf("\n Список занесен в файл 'result.txt'\n");
                #endif
            default:
                break;
        }
    }
}
```

```

printf("Время выполнения функции: %dms%s", finish1, "\n\n");
#else
Print_list_in_file(list);
printf("\n Список занесен в файл 'result.txt'\n\n");
#endif
break;
case 3:
printf("\nЭлемент, который будет вставлен: \n\n");
Print_bird(paste_bird);
printf("\nВведите позицию, на которую нужно вставить элемент: \n");
printf(" --> ");
int pos;
scanf("%d", &pos);
#ifdef DEBUG
printf("\nВыполняется функция Insert_element");
start1 = clock();
Insert_element(list, pos, paste_bird);
finish1 = clock();
finish1 -= start1;
printf("\nЭлемент вставлен\n");
printf("Время выполнения функции: %dms%s", finish1, "\n\n\n");
#else
Insert_element(list, pos, paste_bird);
printf("\nЭлемент вставлен\n\n\n");
#endif
break;
case 4:
printf("\nВведите позицию элемента, который будет удалён: \n");
printf(" --> ");
int num;
scanf("%d", &num);
#ifdef DEBUG
printf("\nВыполняется функция Delete_element");
start1 = clock();
Delete_element(list, num);
finish1 = clock();
finish1 -= start1;
printf("\nЭлемент удалён\n");
printf("Время выполнения функции: %dms%s", finish1, "\n\n\n");
#else
Delete_element(list, num);
printf("\nЭлемент удалён\n\n\n");
#endif
break;
case 5:
printf("\nВведите критерий, по которому искать птицу: \n");
printf("\t1. С кольцом или без\n");
printf("\t2. Имя\n");
printf("\t3. Возраст\n");
printf("\t4. Площадь скворечника\n");
printf("\t5. Высота скворечника\n");
printf("\t6. Кол-во кормушек скворечника\n");
printf("\t7. Является ли скворечник гнездом\n");
printf("\t8. Пол птицы\n");
printf(" --> ");
int number;
scanf("%d", &number);
printf("\n");
#ifdef DEBUG
printf("\nВыполняется функция Find_bird");
start1 = clock();
Find_bird(list, number);

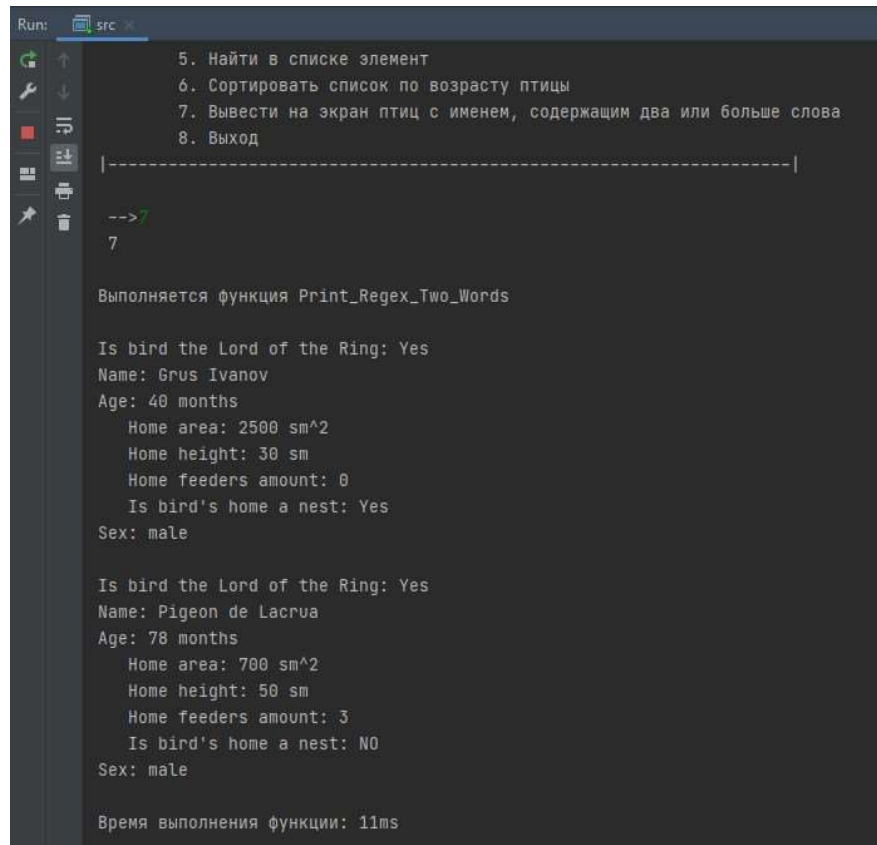
```

```

        finish1 = clock();
        finish1 -= start1;
        printf("Время выполнения функции: %dms%s", finish1, "\n\n");
    #else
        Find_bird(list, number);
    #endif
    break;
case 6:
    #ifdef DEBUG
        printf("\nВыполняется функция Sort_birds_by_age");
        start1 = clock();
        Sort_birds_by_age(list);
        finish1 = clock();
        finish1 -= start1;
        printf("\nСписок отсортирован \n");
        printf("Время выполнения функции: %dms%s", finish1, "\n\n");
    #else
        Sort_birds_by_age(list);
        printf("\nСписок отсортирован \n");
    #endif
    break;
case 7:
    #ifdef DEBUG
        printf("\nВыполняется функция Print_Regex_Two_Words\n\n");
        start1 = clock();
        Print_Regex_Two_Words(list);
        finish1 = clock();
        finish1 -= start1;
        printf("Время выполнения функции: %dms%s", finish1, "\n\n");
    #else
        Print_Regex_Two_Words(list);
    #endif
    break;
case 8:
    #ifdef DEBUG
        printf("\nВыполняется функция Free_list");
        start1 = clock();
        Free_list(list);
        finish1 = clock();
        finish1 -= start1;
        free(list);
        printf("\nВремя выполнения функции: %dms%s", finish1, "\n");
        printf("\nЗавершение работы программы...\n");
        return 0;
    #else
        Free_list(list);
        free(list);
        printf("\nЗавершение работы программы...\n");
        return 0;
    #endif
default:
    printf("\nERROR: Вы ввели некорректный вариант\n");
    break;
}
}
}

```

## Результати роботи програми



```
Run: src x
5. Найти в списке элемент
6. Сортировать список по возрасту птицы
7. Вывести на экран птиц с именем, содержащим два или больше слова
8. Выход

-----|
-->7
7

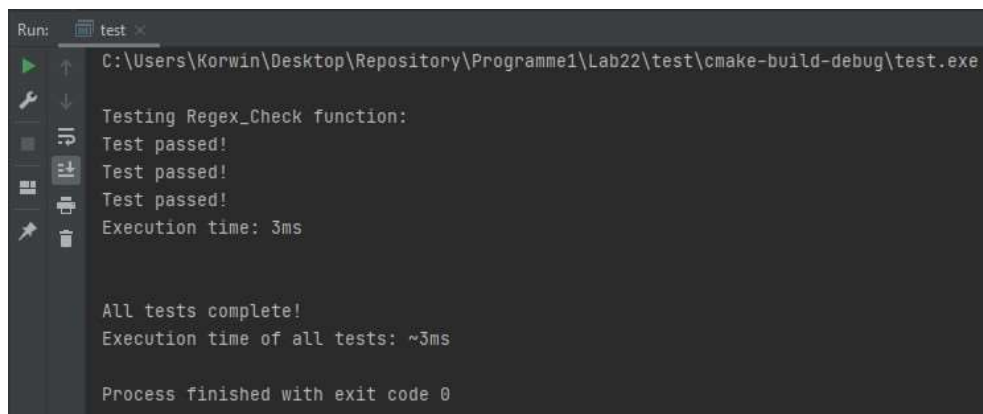
Выполняется функция Print_Regex_Two_Words

Is bird the Lord of the Ring: Yes
Name: Grus Ivanov
Age: 40 months
  Home area: 2500 sm^2
  Home height: 30 sm
  Home feeders amount: 0
  Is bird's home a nest: Yes
Sex: male

Is bird the Lord of the Ring: Yes
Name: Pigeon de Lacrua
Age: 78 months
  Home area: 700 sm^2
  Home height: 50 sm
  Home feeders amount: 3
  Is bird's home a nest: NO
Sex: male

Время выполнения функции: 11ms
```

Рисунок 2 — Результат успішного виконання програми



```
Run: test x
C:\Users\Korwin\Desktop\Repository\Programme1\Lab22\test\cmake-build-debug\test.exe

Testing Regex_Check function:
Test passed!
Test passed!
Test passed!
Execution time: 3ms

All tests complete!
Execution time of all tests: ~3ms

Process finished with exit code 0
```

Рисунок 3 — Результат успішного виконання тестів

## Висновки

Під час виконання даної лабораторної роботи було отримано навички роботи із регулярними виразами.