Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2
з дисципліни
«Алгоритми і структури даних»

Виконала:
студентка групи ІМ-43:
Козаченко Софія Олександрівна
номер у списку групи: 14

Перевірив:
Сергієнко А. М.

Київ 2025

Завдання :

1. Створити список з n (n > 0) елементів (n вводиться з клавіатури), якщо інша кількість елементів не вказана у конкретному завданні за варіантом.
2. Тип ключів (інформаційних полів) задано за варіантом.
3. Вид списку (черга, стек, дек, прямий однозв'язний лінійний список, обернений однозв'язний лінійний список, двозв'язний лінійний список, однозв'язний кільцевий список, двозв'язний кільцевий список) вибрати самостійно з метою найбільш доцільного розв'язку поставленої за варіантом задачі.
4. Створити функції (або процедури) для роботи зі списком (для створення, обробки, додавання чи видалення елементів, виводу даних зі списку в консоль, звільнення пам'яті тощо).
5. Значення елементів списку взяти самостійно такими, щоб можна було продемонструвати коректність роботи алгоритму програми. Введення значень елементів списку можна виконати довільним способом (випадкові числа, формування значень за формулою, введення з файлу чи з клавіатури).
6. Виконати над створеним списком дії, вказані за варіантом, та коректне звільнення пам'яті списку.
7. При виконанні заданих дій, виводі значень елементів та звільненні пам'яті списку вважати, що довжина списку (кількість елементів) невідома на момент виконання цих дій. Тобто, не дозволяється зберігати довжину списку як константу, змінну чи додаткове поле

Варіант 14 :

Ключами елементів списку є рядки довжиною не більше 5-ти символів. Перекомпонувати список так, щоб елементи списку були розташовані в оберненому порядку (виконати «дзеркальне відображення» списку), не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Текст програми :

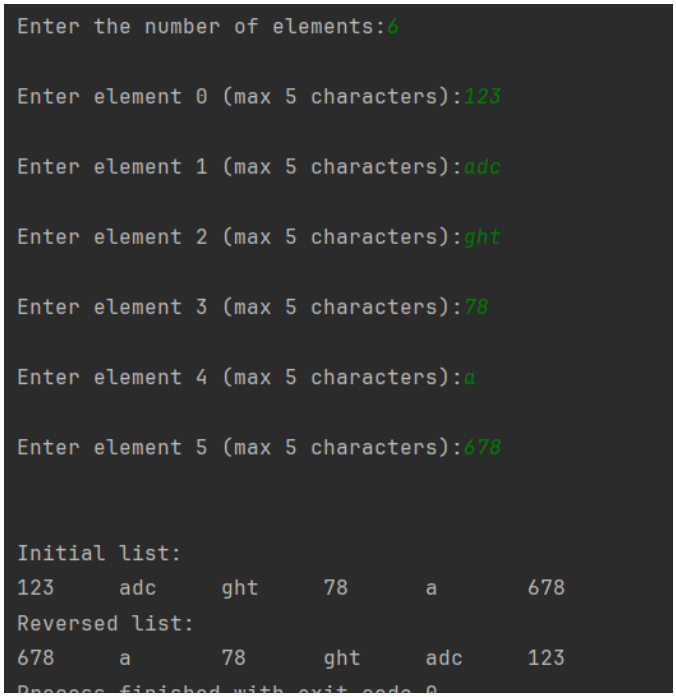
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

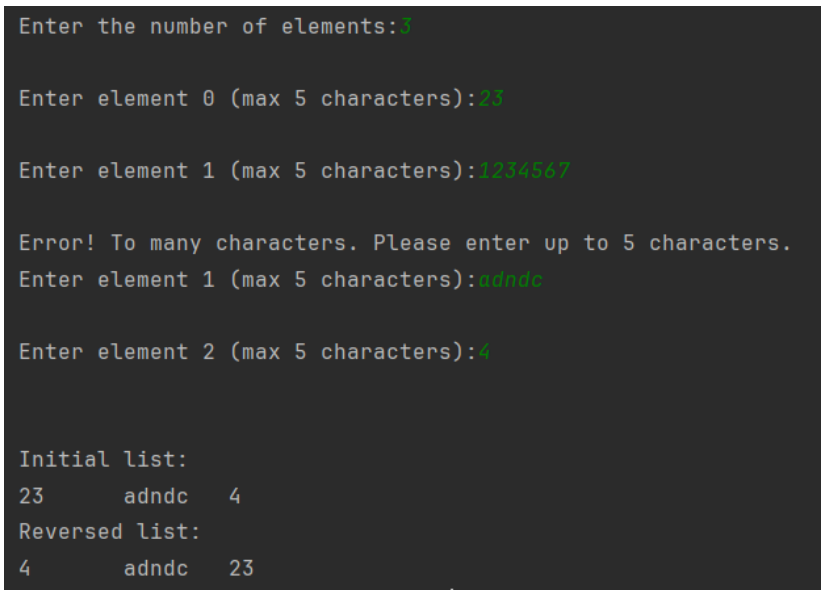
```
typedef struct Node {
    char key[6];
    struct Node* prev;
    struct Node* next;
} Node;

void clearMemory(Node* head);

Node* createList (int n) {
    if (n <= 0) return NULL;
    Node* head = NULL;
    Node* tail = NULL;
    for (int i = 0; i < n; i++) {
        Node* newNode = (Node*)malloc(sizeof(Node));
        if (newNode == NULL) {
            printf("Not enough memory for new node\n");
            clearMemory(head);
            return NULL;
        };

        char input[20];
        while (1) {
            printf("Enter element %d (max 5 characters): ", i);
            fgets(input, sizeof(input), stdin);
            input[strcspn(input, "\n")] = '\0';

            if (strlen(input) <= 5) {
                strcpy(newNode->key, input);
                break;
            } else {
                printf("Error! Too many characters. Please enter up to 5 characters.\n");
            }
        }

        newNode->next = NULL;
        newNode->prev = tail;

        if (tail != NULL) {
            tail->next = newNode;
```

```
        } else {
            head = newNode;
        }
        tail = newNode;
    }
    return head;
}

Node* reverseList(Node* head) {
    if (head == NULL) return NULL;
    Node* temp = NULL;
    Node* current = head;

    while (current != NULL) {
        temp = current->prev;
        current->prev = current->next;
        current->next = temp;
        if (current->prev == NULL)
            return current;
        current = current->prev;
    }
    return head;
}

void printList (Node* head) {
    while (head != NULL) {
        printf("%s\t", head->key);
        head = head->next;
    }
}

void clearMemory (Node* head) {
    while (head != NULL) {
        Node* temp = head;
        head = head->next;
        free(temp);
    }
}
```

```
int main() {
    int n;
    printf("Enter the number of elements: ");
    if (scanf("%d", &n) != 1 || n <= 0) {
        printf("Invalid input!\n");
        return 1;
    }
    while (getchar() != '\n');

    Node* head = createList(n);

    printf("\nInitial list:\n");
    printList(head);

    head = reverseList(head);

    printf("\nReversed list:\n");
    printList(head);

    clearMemory(head);

    return 0;
}
```

Скріншоти тестування :





Висновок :

Виконуючи лабораторну роботу №2, я засвоїла теоретичний матеріл з теми «Зв’язані динамічні структури даних. Списки» та набула практичного досвіду використання зв’язних динамічних структур даних у вигляді двозв’язного лінійного списку та його обробки. Також навчилась принципам організації пам’яті для динамічних структур. І додатково панувала нові концепції та техніки програмування, до прикладу : як працює fgets(), яка забезпечує перевірку введення на відповідність вимогам (не більше 5 символів) та використала її у лабораторній роботі.