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»

Виконав:

Студент групи ІМ-42

Лобань Михайло Юрійович

номер у списку групи: 20

Перевірив:

Сергієнко А. М.

Київ 2025

Загальна постановка 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.

Завдання за варіантом

Варіант № 20

Ключами елементів списку є цілі числа. Кількість елементів списку повинна дорівнювати $\boldsymbol{2n}$. Перекомпонувати елементи списку так, розташування елементів було наступним: $a_1, a_{n+1}, a_2, a_{n+2}, a_3, \ldots, a_n, a_{2n}$, де $\boldsymbol{a_i}$ — $\boldsymbol{i}$-й компонент списку, не використовуючи додаткових структур даних, крім простих змінних (тобто «на тому ж місці»).

Текст програми

```
#include <stdio.h>

#include <stdlib.h>

#include <time.h>
```

```
struct element {
    int value;
    struct element* next;
};

struct element* create_element(int value) {
    struct element* new_element = (struct element*)malloc(sizeof(struct element));
    new_element->value = value;
    new_element->next = NULL;
    return new_element;
}

void insert(struct element** head, int value) {
    struct element* new_element = create_element(value);
    new_element->next = *head;
    *head = new_element;
}

void print_list(struct element* head) {
    struct element* temp = head;
    while (temp != NULL) {
        printf("%d -> ", temp->value);
        temp = temp->next;
    }
    printf("NULL\n");
}
```

```
void regroup_list(struct element** head) {

    if (!head || !(*head)) return;

    struct element* first = *head;
    struct element* second = (*head)->next;

    while(second && second->next) {
        first = first->next;
        second = second->next->next;
    }

    struct element* first_part = *head;
    struct element* second_part = first->next;

    first->next = NULL;

    while(second_part) {
        struct element* first_temp = first_part->next;
        struct element* second_temp = second_part->next;
        first_part->next = second_part;
        second_part->next = first_temp;
        first_part = first_temp;
        second_part = second_temp;
    }

}
```

```
void free_list(struct element* head) {
    struct element* temp;
    while (head != NULL) {
        temp = head;
        head = head->next;
        free(temp);
    }
}

int main() {
    struct element* head = NULL;
    int n;

    printf("Enter the number of elements: ");
    scanf("%d", &n);

    if(n % 2 != 0 || n < 0) {
        printf("Invalid number of elements");
        return 1;
    }

    printf("\n");

    srand(time(NULL));
    for (int i = 0; i < n; i++) {
        int value = (rand() % 21) - 10;
        insert(&head, value);
    }
```

```
    printf("Generated Linked List: \n\n");
    print_list(head);

    printf("\n");

    regroup_list(&head);
    printf("Regrouped Linked List: \n\n");
    print_list(head);

    free_list(head);
    return 0;
}
```

Результати тестування програми

N = 8

```
Enter the number of elements: 8

Generated Linked List:

-10 -> 8 -> -4 -> 9 -> 5 -> -5 -> -10 -> 3 -> NULL

Regrouped Linked List:

-10 -> 5 -> 8 -> -5 -> -4 -> -10 -> 9 -> 3 -> NULL

Process returned 0 (0x0)   execution time : 1.957 s
Press any key to continue.
```

N = 10

```
Enter the number of elements: 10

Generated Linked List:

-10 -> -10 -> -9 -> 5 -> -3 -> 1 -> 3 -> -4 -> -6 -> -4 -> NULL

Regrouped Linked List:

-10 -> 1 -> -10 -> 3 -> -9 -> -4 -> 5 -> -6 -> -3 -> -4 -> NULL

Process returned 0 (0x0)   execution time : 1.173 s
Press any key to continue.
```

N = 12

```
Enter the number of elements: 12

Generated Linked List:

-5 -> -8 -> 9 -> -6 -> -7 -> 5 -> -3 -> -3 -> -10 -> 4 -> 2 -> 6 -> NULL

Regrouped Linked List:

-5 -> -3 -> -8 -> -3 -> 9 -> -10 -> -6 -> 4 -> -7 -> 2 -> 5 -> 6 -> NULL

Process returned 0 (0x0)   execution time : 1.065 s
Press any key to continue.
```

N = 7 (error)

```
Enter the number of elements: 7
Invalid number of elements
Process returned 1 (0x1)   execution time : 0.977 s
Press any key to continue.
```

N = -14 (error)

```
Enter the number of elements: -14
Invalid number of elements
Process returned 1 (0x1)   execution time : 1.701 s
Press any key to continue.
```

Висновок

Створив програму, яка перекомпоновує однозв’язний список відповідно до варіанту, шляхом його розбиття на два підсписки, а потім їх злиття, не використовуючи додаткових структур даних, крім простих змінних.