1. Цель работы

Целью работы является изучение структур данных «линейный список» и «циклический список», а также получение практических навыков их реализации.

2. Задание

Согласно варианту №20:

16	Дана последовательность чисел и число K. Необходимо суммировать элементы исходной последовательности, пока сумма не будет больше или равна K, после чего выводятся все просуммированные элементы.	Циклический двусвязный

3. Листинг программы

```
// Вариант 20.

#include <iostream>

using namespace std;

struct List {
    float num;
    List* prev;
    List* next;
};

float get_num_int() // Запрос и проверка числа на корректность
{
    float x;

    cin >> x;
    while (cin.fail() || (cin.peek() != '\n')) // Проверка на корректность
    {
        cin.clear(); // Очищение флага ошибки
        cin.ignore(numeric_limits<streamsize>::max(), '\n'); // Очистка буфера
запроса
        cout << "Повторите ввод: ";
        cin >> x;
    }

    return x;
}

bool IsEmpty(List* head) // Проверка пустой ли список
{
    return head == NULL;
}

void PrintList(List* head) // Вывод списка
{
    List* cur = head;

    do{
        cout << cur->num << endl;
        cur = cur->next;
    }while (cur != head);
```

```
}

bool FindElem(List* head, const float x) // Нахождение элемента
{
    List* cur = head;

    do {
        if (cur->num == x)
        {
            return true;
        }
        cur = cur->next;
    } while (cur != head);

    return false;
}

void AddElem(List** head, float x) // Добавление элемента
{
    List* cur = *head;
    List* p = new List;

    if (cur == NULL)
    {
        p->prev = p;
        p->next = p;
        p->num = x;
        *head = p;
    }
    else
    {
        do{
            cur = cur->next;
        } while (cur->next != *head);
        p->prev = cur;
        p->next = *head;
        p->num = x;
        cur->next = p;
        (*head)->prev = p;

    }

}

void RemoveElem(List** head, float x) // Удаление элемента
{
    List* cur = *head;

    do{
        if (cur->num == x)
        {
            break;
        }
        cur = cur->next;
    } while (cur != *head);

    if ((cur->next == *head) && (cur->prev == *head))
    {
        delete cur;
        *head = NULL;
    }
    else
    {
        cur->prev->next = cur->next;
        cur->next->prev = cur->prev;
```

```
        if (cur == *head)
        {
            *head = cur->next;
        }
        delete cur;
    }
}

void FreeList(List** head) // Очистка памяти
{
    List* cur = NULL;
    List* mainhead = *head;

    while ((*head)->next != mainhead)
    {
        cur = *head;
        *head = (*head)->next;
        delete cur;
    }

    delete* head;

    cout << "Удаление прошло успешно!" << endl;
}

float requestK() // Запрос числа K
{
    cout << "Введите число K: " << endl;
    return get_num_int();
}

void RequestElems(List** head, const float K) // Запрос элементов
{
    cout << "Начинайте вводить элементы." << endl;
    cout << "Число " << K << " останавливает ввод" << endl;
    while (true)
    {
        float x = get_num_int();
        AddElem(head, x);
        if (x == K)
        {
            break;
        }
    }
}

float FindSum(List* head, const float K) // Поиск и вывод последовательности по
заданию
{
    float S = 0;
    while (true)
    {
        float x = head->num;
        if (S + x >= K)
        {
            S += x;
            cout << head->num << endl;
            break;
        }
        else
        {
            S += x;
            cout << head->num << endl;
        }
```

```
        if (x == K)
        {
            break;
        }
        head = head->next;
    }

    return S;
}

int main()
{
    setlocale(LC_ALL, "RUS");
    List* list = NULL;
    int x;
    float K = NULL;
    cout << "Добро пожаловать!" << endl;

    while (true)
    {
        cout << "Выберите пункт меню." << endl;
        cout << "1. Добавить элемент в список" << endl;
        cout << "2. Удалить элемент из списка" << endl;
        cout << "3. Установить число K" << endl;
        cout << "4. Начать заполнение списка до числа K" << endl;
        cout << "5. Вывести список" << endl;
        cout << "6. Поиск суммы для K" << endl;
        cout << "7. Выход" << endl;
        cin >> x;
        if (x == 1) // Добавление
        {
            system("cls");
            cout << "Введите ваше значение для добавления: " << endl;
            float x = get_num_int();
            AddElem(&list, x);

        }
        else if (x == 2) // Удаление
        {
            system("cls");
            if (not(IsEmpty(list)))
            {
                cout << "Введите ваше значение для удаления: " << endl;
                float x = get_num_int();
                if (FindElem(list, x))
                {
                    RemoveElem(&list, x);
                    cout << "Число " << x << " успешно удалено!" << endl;
                }
                else
                {
                    cout << "Число " << x << " отсутствует." << endl;
                }
            }
            else
            {
                cout << "Ваш список пуст." << endl;
            }

        }
        else if (x == 3) // Число K
        {
            system("cls");
            K = requestK();
        }
```

```
        else if (x == 4) // Заполнение списка
        {
            system("cls");
            if (K)
            {
                cout << "Начинаем заполнять список..." << endl;
                RequestElems(&list, K);
            }
            else
            {
                cout << "Число K отсутствует." << endl;
            }
        }
        else if (x == 5) // Вывести список
        {
            system("cls");
            if (not(IsEmpty(list)))
            {
                cout << "Ваш список: " << endl;
                PrintList(list);
            }
            else
            {
                cout << "Ваш список пуст." << endl;
            }
        }
        else if (x == 6) // Поиск суммы
        {
            system("cls");
            if (K)
            {
                cout << "Суммируем пока сумма будет или больше или равна " << K << "
: " << endl;
                float S = FindSum(list, K);
                cout << "Итоговая сумма: " << S << endl;
            }
            else
            {
                cout << "Число K отсутствует." << endl;
            }

        }
        else if (x == 7) // Выход
        {
            cout << "Завершение работы..." << endl;
            break;
        }
        else // Неправильный ввод
        {
            system("cls");
        }

    }

    if (not(IsEmpty(list))) // Очистка памяти если список не пустой
    {
        cout << "Удаление списка..." << endl;
        FreeList(&list);
    }
}
```

4. Контрольные примеры

- Пример 1:

Список 3 3 2 5 10 3

Число K = 10

Найденная последовательность 3 3 2 5

Сумма 13

- Пример 2:

Список 1 4 1 1 2 9 4 10

Число K = 10

Найденная последовательность 1 4 1 1 2 9

Сумма 18

5. **Выводы**

В процессе лабораторной работы была изучена структура данных циклический список, а так же практические навыки реализации.