

Основы программирования

Лабораторная работа №13

Связанный список

Власенко О.Ф.

Задача 1 – Связанный, но не динамический! (код нужно копировать из презентации!)

```
#define _CRT_SECURE_NO_WARNINGS
```

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

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

```
struct Node {  
    int data;  
    struct Node * next;  
};
```

```
struct Node * first = NULL;
```

```
void printList() {  
    struct Node * ptr = first;  
    while (ptr != NULL) {  
        printf("(%d) -> ", ptr->data);  
        ptr = ptr->next;  
    }  
    printf("NULL\n");  
}
```

Задача 1 – Связанный, но не динамический! (2)

```
void main() {
    printList();

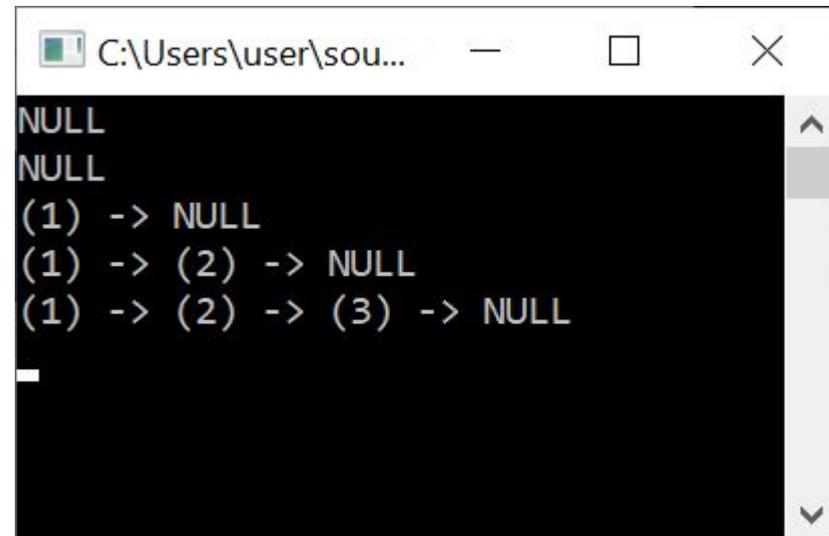
    struct Node node1 = { 1, NULL };
    struct Node node2 = { 2, NULL };
    struct Node node3 = { 3, NULL };
    printList();

    first = &node1;
    printList();

    node1.next = &node2;
    printList();

    node2.next = &node3;
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\sou...
NULL
NULL
(1) -> NULL
(1) -> (2) -> NULL
(1) -> (2) -> (3) -> NULL
_
```

Задача 2 – Связанный, и динамический!

```
void addToHead(int value) {
    struct Node * newNode = (struct Node*)malloc(sizeof(struct Node));

    newNode->next = first;
    newNode->data = value;

    first = newNode;
}

int deleteFromHead() {
    int value = first->data;
    struct Node * delNode = first;

    first = first->next;
    free(delNode);

    return value;
}
```

Задача 2 – Связанный, и динамический! (2)

```
void main() {
    printList();

    addToHead(10);
    printList();

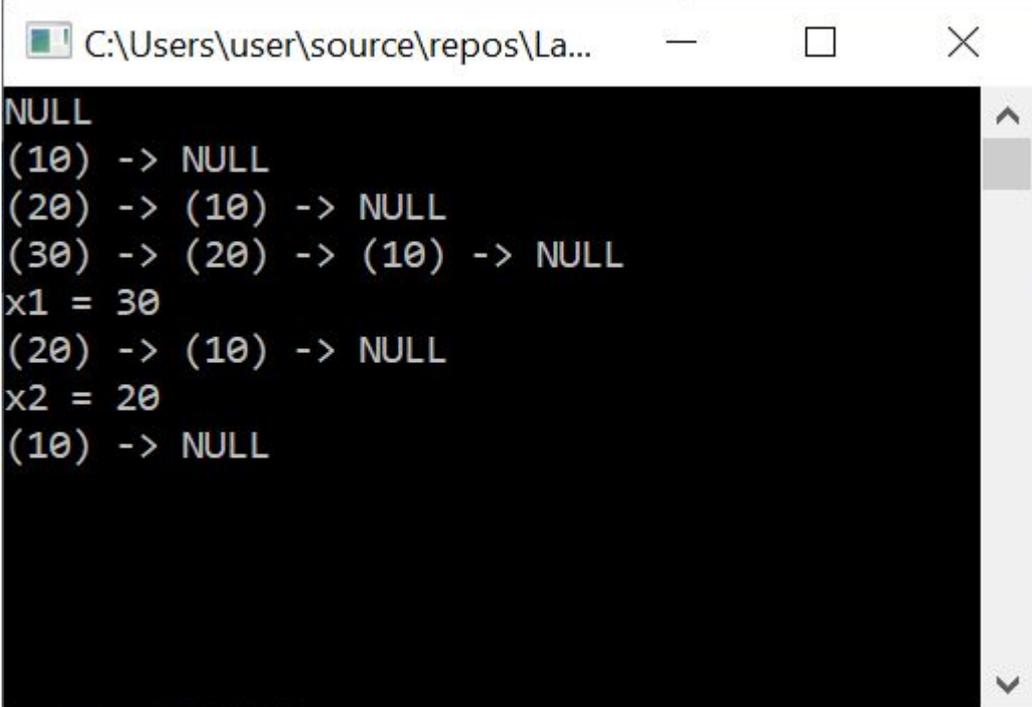
    addToHead(20);
    printList();

    addToHead(30);
    printList();

    int x1 = deleteFromHead();
    printf("x1 = %d\n", x1);
    printList();

    int x2 = deleteFromHead();
    printf("x2 = %d\n", x2);
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\repos\La...
NULL
(10) -> NULL
(20) -> (10) -> NULL
(30) -> (20) -> (10) -> NULL
x1 = 30
(20) -> (10) -> NULL
x2 = 20
(10) -> NULL
```

Задача 3 – Ищем по значению и очищаем!

```
int contains(int value) {
    struct Node * ptr = first;
    while (ptr != NULL) {
        if (ptr->data == value) {
            return 1;
        }
        ptr = ptr->next;
    }
    return 0;
}
```

```
void clearList() {
    while (first != NULL)
    {
        struct Node * delNode = first;
        first = first->next;
        free(delNode);
    }
}
```

Задача 3 – Ищем по значению и очищаем! (2)

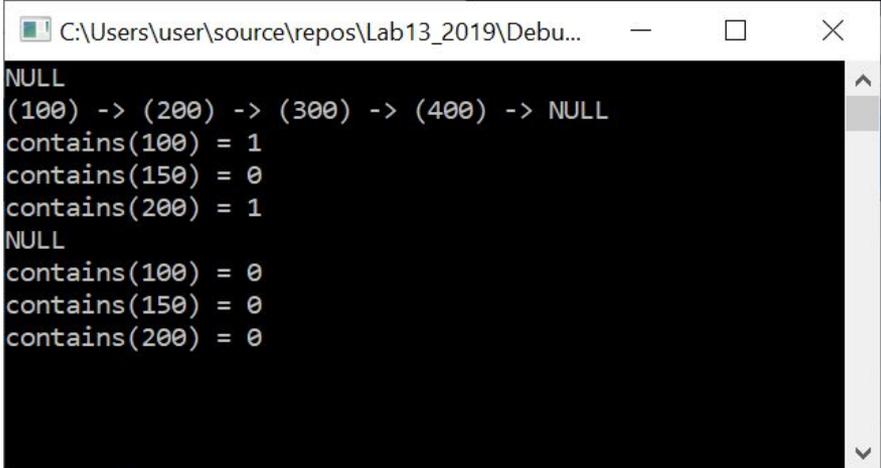
```
void main() {
    printList();

    addToHead(400);
    addToHead(300);
    addToHead(200);
    addToHead(100);
    printList();

    printf("contains(100) = %d\n", contains(100));
    printf("contains(150) = %d\n", contains(150));
    printf("contains(200) = %d\n", contains(200));

    clearList();
    printList();

    printf("contains(100) = %d\n", contains(100));
    printf("contains(150) = %d\n", contains(150));
    printf("contains(200) = %d\n", contains(200));
    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\repos\Lab13_2019\Debu...
NULL
(100) -> (200) -> (300) -> (400) -> NULL
contains(100) = 1
contains(150) = 0
contains(200) = 1
NULL
contains(100) = 0
contains(150) = 0
contains(200) = 0
```

Задача 4 – Подсчитать сумму всех элементов (код надо набирать руками!)

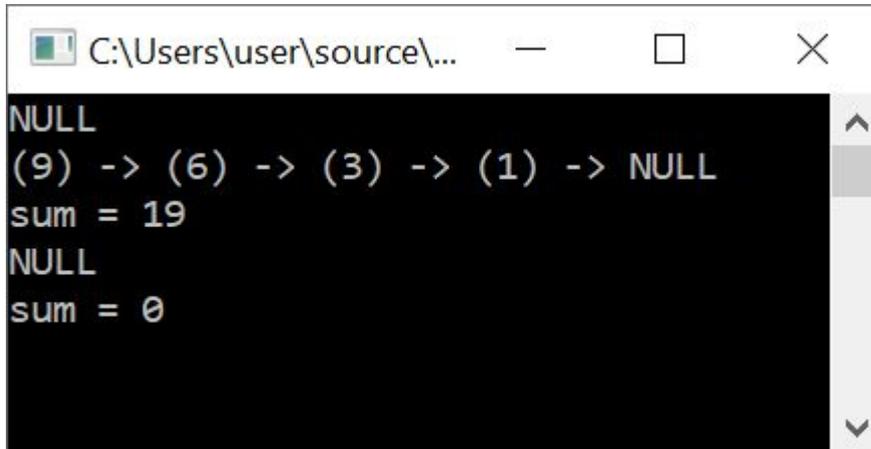
```
int sum()
{
    struct Node * ptr = first;
    int s = 0;
    while (ptr != NULL) {
        s += ptr->data;
        ptr = ptr->next;
    }
    return s;
}
```

```
void main() {
    printList();

    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(9);
    printList();

    printf("sum = %d\n", sum());

    clearList();
    printList();
    printf("sum = %d\n", sum());
    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\...
NULL
(9) -> (6) -> (3) -> (1) -> NULL
sum = 19
NULL
sum = 0
```

Задача 5 – Подсчитать количество четных элементов

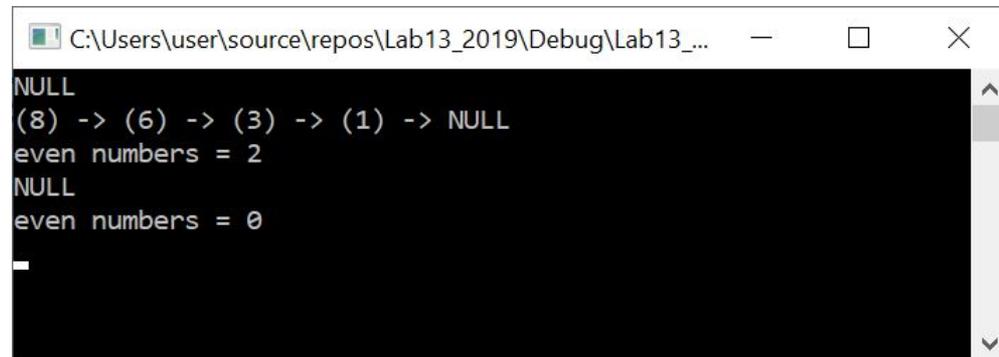
(код надо придумать!)

```
void main() {
    printList();

    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(8);
    printList();

    printf("even numbers = %d\n", evenCount());

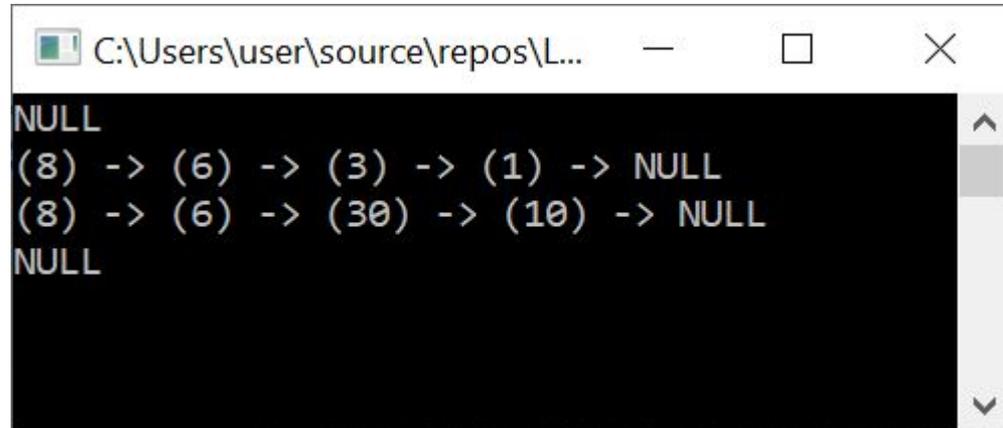
    clearList();
    printList();
    printf("even numbers = %d\n", evenCount());
    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\repos\Lab13_2019\Debug\Lab13_...
NULL
(8) -> (6) -> (3) -> (1) -> NULL
even numbers = 2
NULL
even numbers = 0
```

Задача 6 – Все нечетные увеличить в 10 раз

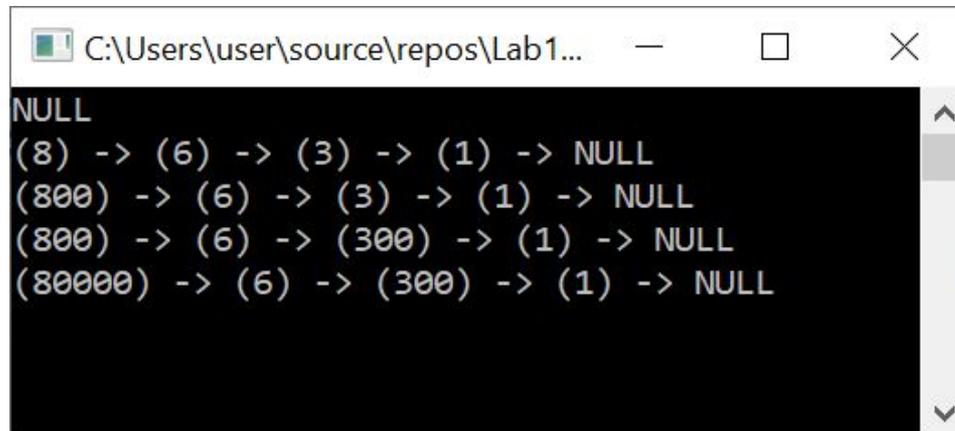
```
void main() {  
    printList();  
  
    addToHead(1);  
    addToHead(3);  
    addToHead(6);  
    addToHead(8);  
    printList();  
  
    oddsX10();  
    printList();  
  
    clearList();  
    printList();  
  
    {  
        int x;  
        scanf("%d", &x);  
    }  
}
```



```
C:\Users\user\source\repos\L...  
NULL  
(8) -> (6) -> (3) -> (1) -> NULL  
(8) -> (6) -> (30) -> (10) -> NULL  
NULL
```

Задача 7 – i-ый элемент увеличить в 100 раз

```
void elementIx100(int i)
{
    struct Node * ptr = first;
    int index = 0;
    while (ptr != NULL) {
        if (index == i) {
            ptr->data = ptr->data * 100;
            return;
        }
        ptr = ptr->next;
        index++;
    }
}
```



```
C:\Users\user\source\repos\Lab1...
NULL
(8) -> (6) -> (3) -> (1) -> NULL
(800) -> (6) -> (3) -> (1) -> NULL
(800) -> (6) -> (300) -> (1) -> NULL
(8000) -> (6) -> (300) -> (1) -> NULL
```

```
void main() {
    printList();

    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(8);
    printList();

    elementIx100(0);
    printList();

    elementIx100(2);
    printList();

    elementIx100(0);
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```

Задача 8 – все элементы левее i-го увеличить в 10 раз

```
void main() {
    printList();

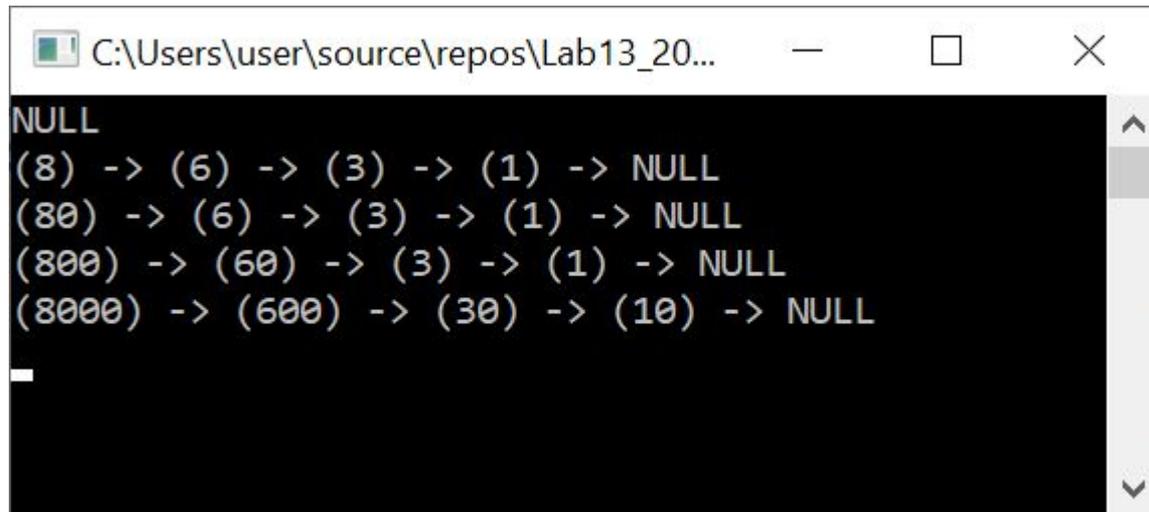
    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(8);
    printList();

    elementLeftIx10(1);
    printList();

    elementLeftIx10(2);
    printList();

    elementLeftIx10(4);
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\repos\Lab13_20...
NULL
(8) -> (6) -> (3) -> (1) -> NULL
(80) -> (6) -> (3) -> (1) -> NULL
(800) -> (60) -> (3) -> (1) -> NULL
(8000) -> (600) -> (30) -> (10) -> NULL
```

Задача 9 – все элементы правее i-го увеличить в 10 раз

```
void main() {
    printList();

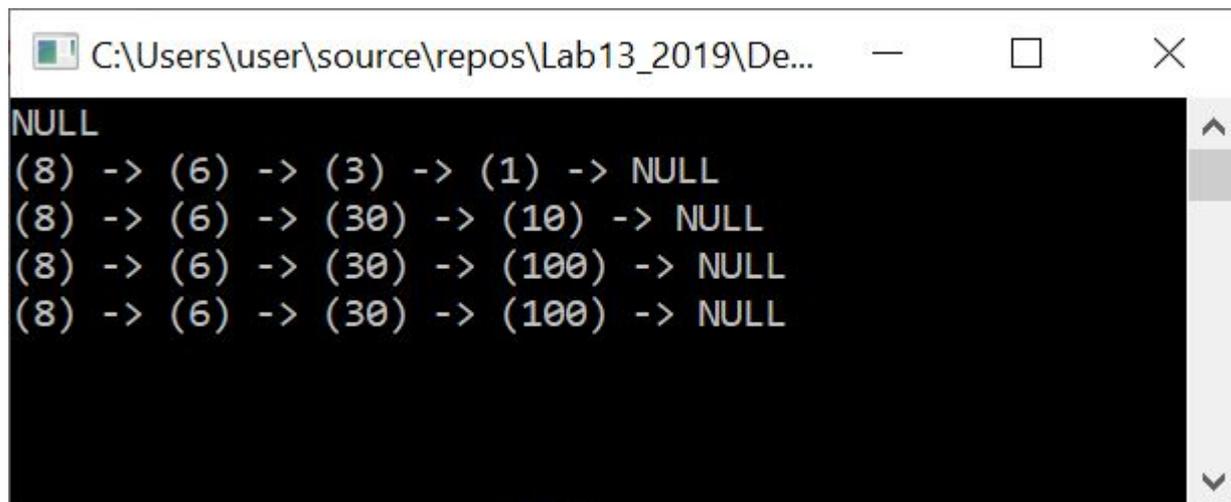
    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(8);
    printList();

    elementRightIx2(1);
    printList();

    elementRightIx2(2);
    printList();

    elementRightIx2(4);
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```



```
C:\Users\user\source\repos\Lab13_2019\De...
NULL
(8) -> (6) -> (3) -> (1) -> NULL
(8) -> (6) -> (30) -> (10) -> NULL
(8) -> (6) -> (30) -> (100) -> NULL
(8) -> (6) -> (30) -> (100) -> NULL
```

Задача 10* – удалить i-ый элемент

```
void deleteElementI(int i)
{
    if (first == NULL) {
        return;
    }

    if (i == 0) {
        deleteFromHead();
        return;
    }

    struct Node * prev_ptr = first;
    struct Node * ptr = first->next;
    int index = 1;

    while (ptr != NULL) {
        if (index == i) { ... }
        prev_ptr = ptr;
        ptr = ptr->next;
        index++;
    }
}
```

```
void main() {
    printList();

    addToHead(1);
    addToHead(3);
    addToHead(6);
    addToHead(8);
    addToHead(12);

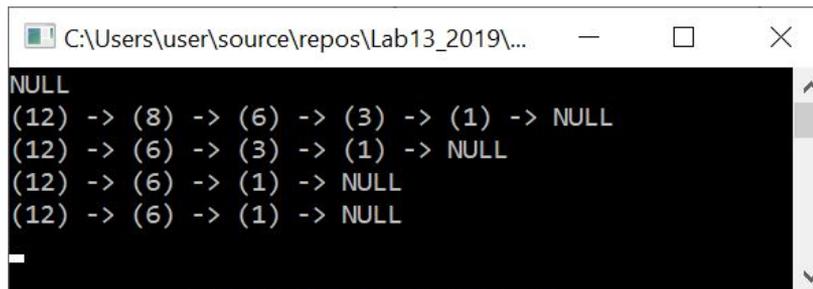
    printList();

    deleteElementI(1);
    printList();

    deleteElementI(2);
    printList();

    deleteElementI(4);
    printList();

    {
        int x;
        scanf("%d", &x);
    }
}
```

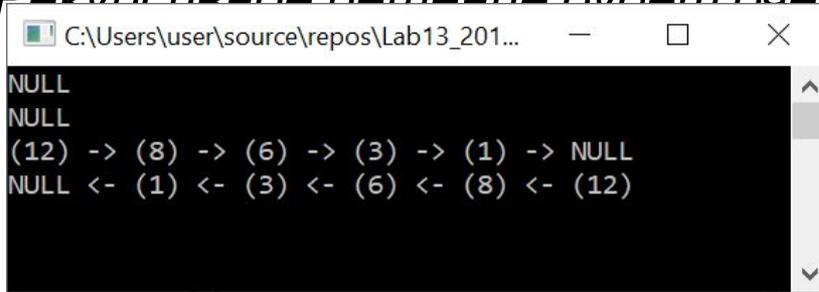


```
C:\Users\user\source\repos\Lab13_2019\...
NULL
(12) -> (8) -> (6) -> (3) -> (1) -> NULL
(12) -> (6) -> (3) -> (1) -> NULL
(12) -> (6) -> (1) -> NULL
(12) -> (6) -> (1) -> NULL
```

Задачи 11+ ***

Во всех задачах нужно изменять поля NEXT, но нельзя трогать поля DATA!!!

- Вывести односвязанный список в обратную сторону
(Рекурсивное решение будет очень элегантно!)



```
C:\Users\user\source\repos\Lab13_201...
NULL
NULL
(12) -> (8) -> (6) -> (3) -> (1) -> NULL
NULL <- (1) <- (3) <- (6) <- (8) <- (12)
```

- Проверить есть ли в списке повторяющиеся элементы
- Поменять местами два соседних элемента
- Поменять местами элементы с индексами i и j
- Отсортировать список по возрастанию

Домашнее задание

Если какие-то из домашних работ не доделаны – доделывайте и сдавайте!