

Министерство науки и высшего образования Российской Федерации  
Новосибирский Государственный технический университет  
Кафедра автоматизированных систем управления



**Отчет по лабораторной работе №1**  
**Вариант №7**  
**по дисциплине «Архитектура средств вычислительной техники»**  
**«Линейные структуры»**

Выполнили  
студенты группы АВТ-813:  
Кинчаров Данил  
Пайхаев Алексей  
Чернаков Кирилл  
Преподаватель:  
Ландовский Владимир Владимирович,  
к.т.н., доцент кафедры АСУ

г. Новосибирск  
2020 г.

## Содержание

Цель работы:.....	3
Задание: .....	3
Схемы структур данных:.....	4
Блок-схемы алгоритмов:.....	5
Текст программы:.....	13
Описание работы программы с интерфейсом: .....	15
Вывод: .....	16

## Цель работы:

Изучить линейные динамические и статические структуры данных и приобрести практические навыки программирования таких структур.

## Задание:

Разработать программную реализацию заданных согласно табл. 6.1 АД с использованием заданной структуры данных.

Т а б л и ц а 6.1

### Варианты заданий

№ п/п	АД	Операторы	Структура данных	Тип элемента
1	Стек	типичный набор (раздел 4.1)	массив	натуральное число
	Список	вставка с сохранением упорядоченности, удаление по значению, поиск позиции (номера элемента) по значению, очистка	односвязный список	символ русского алфавита

## Схемы структур данных:

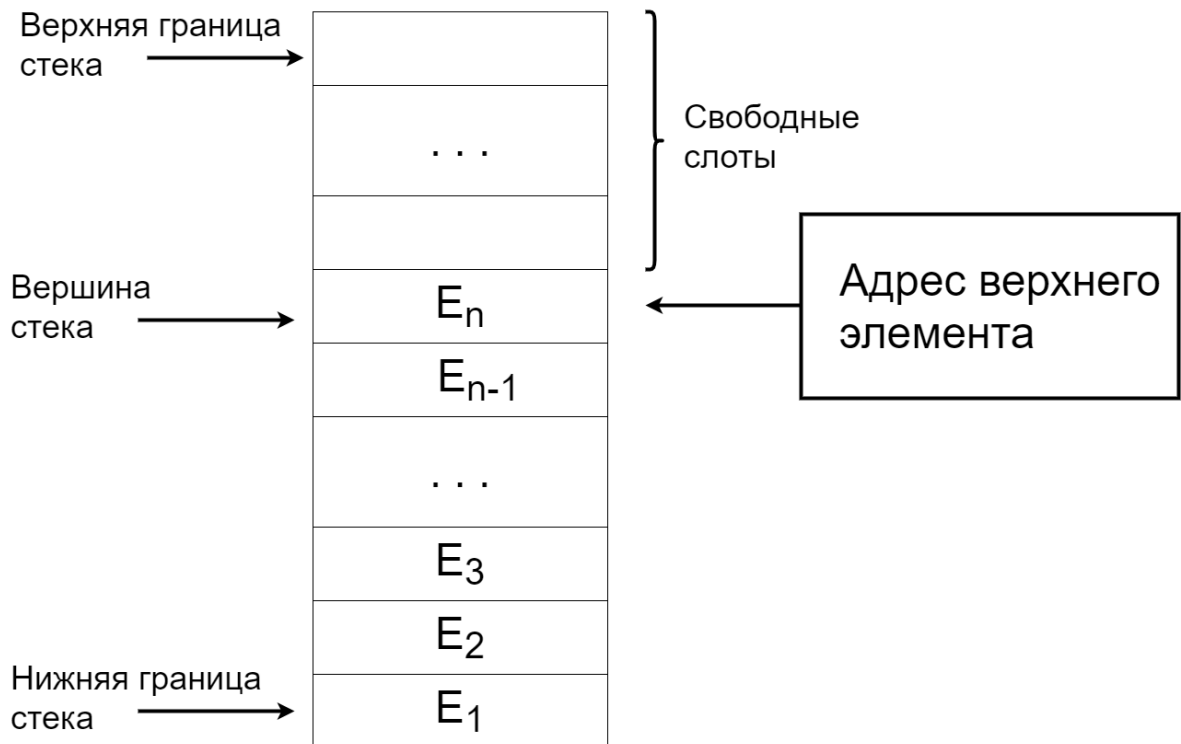


Рисунок 1 – Стек.

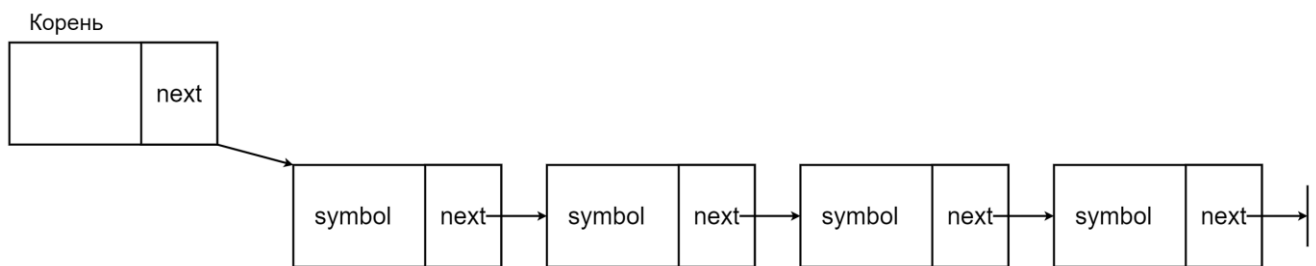


Рисунок 2.1 – Односвязный список.

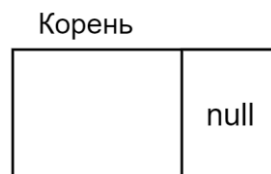
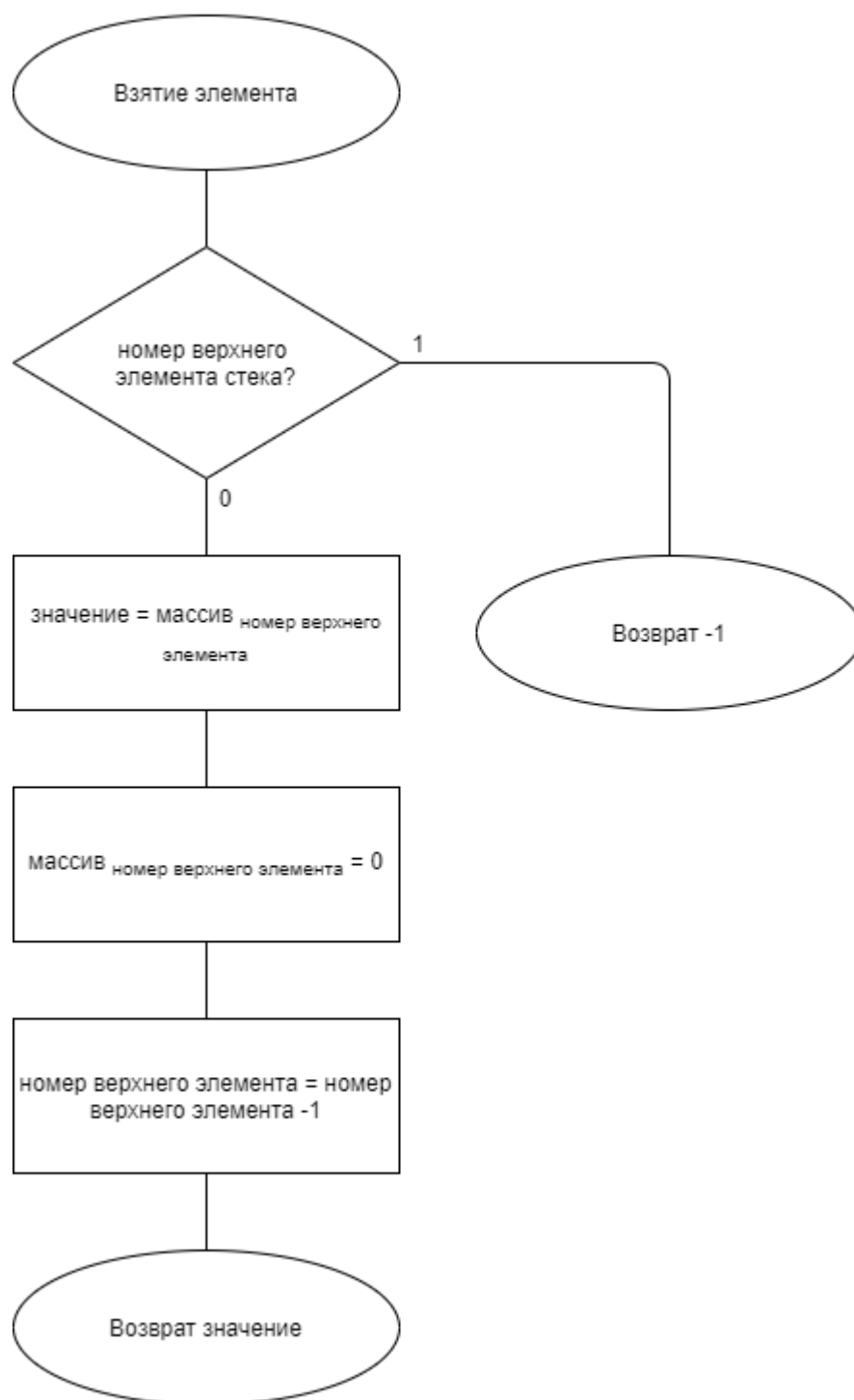
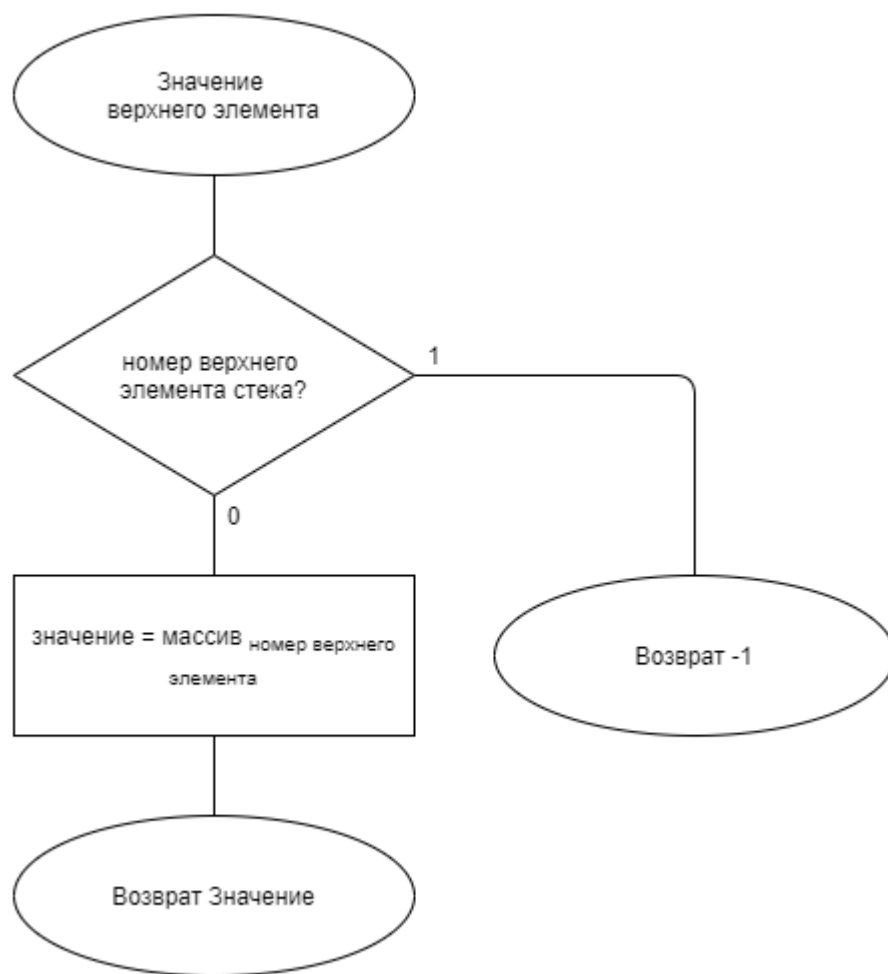
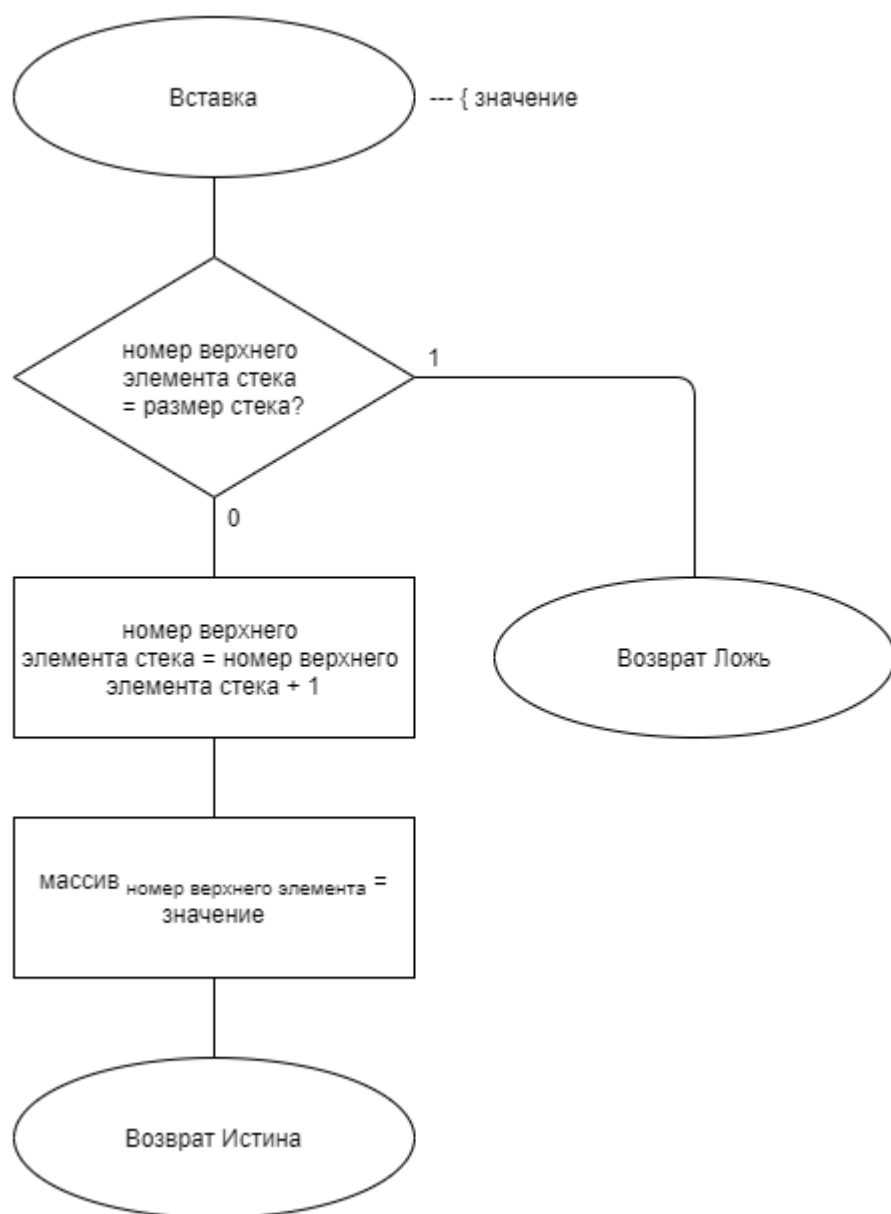


Рисунок 2.2 – Пустой односвязный список, где null – специальное значение обозначающее отсутствие указателя.

## Блок-схемы алгоритмов:







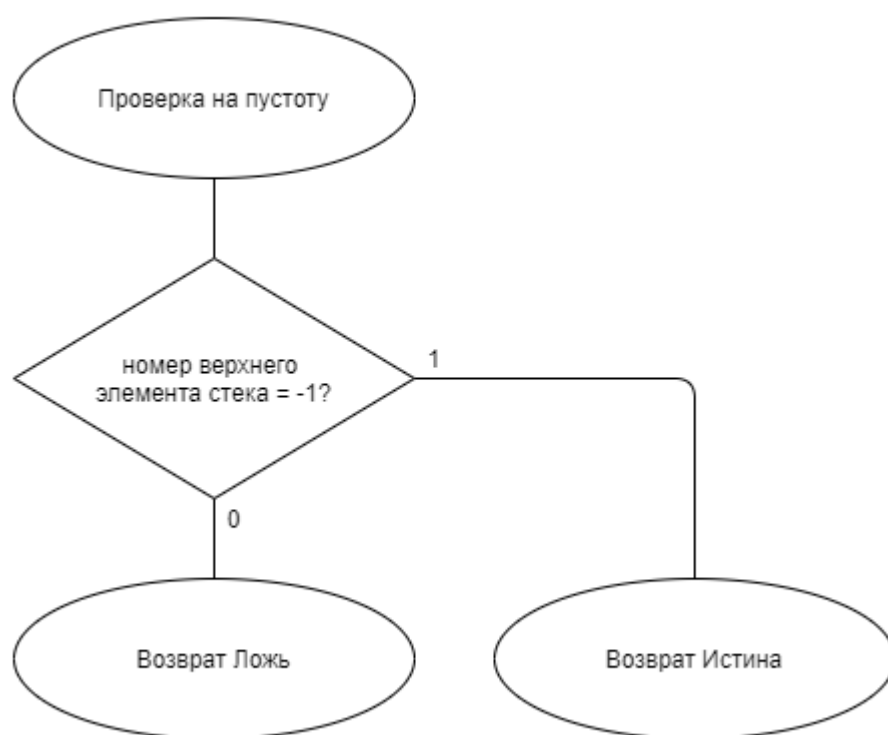
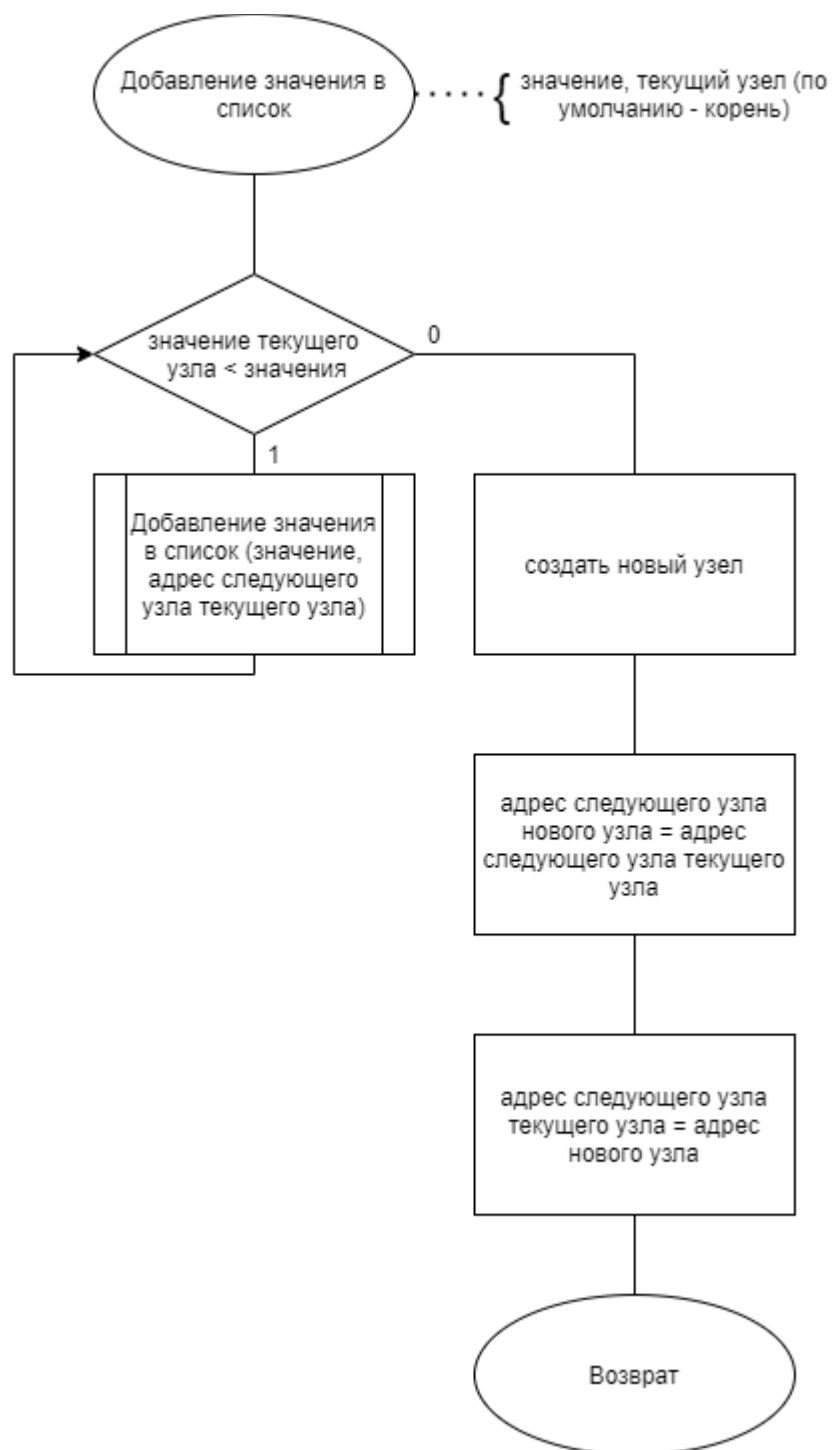
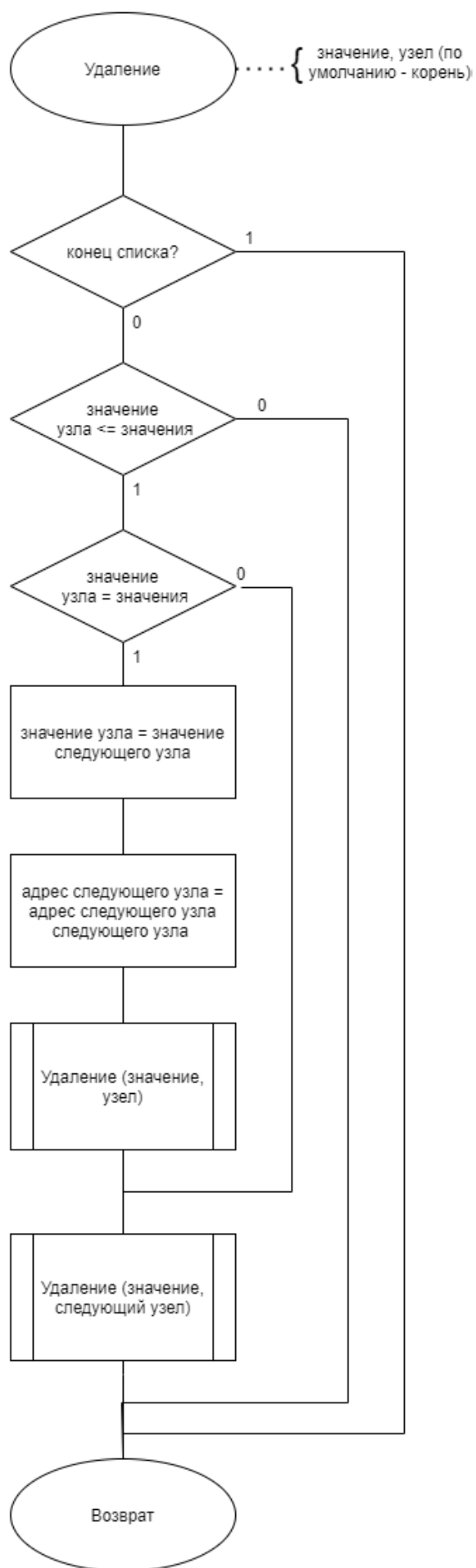
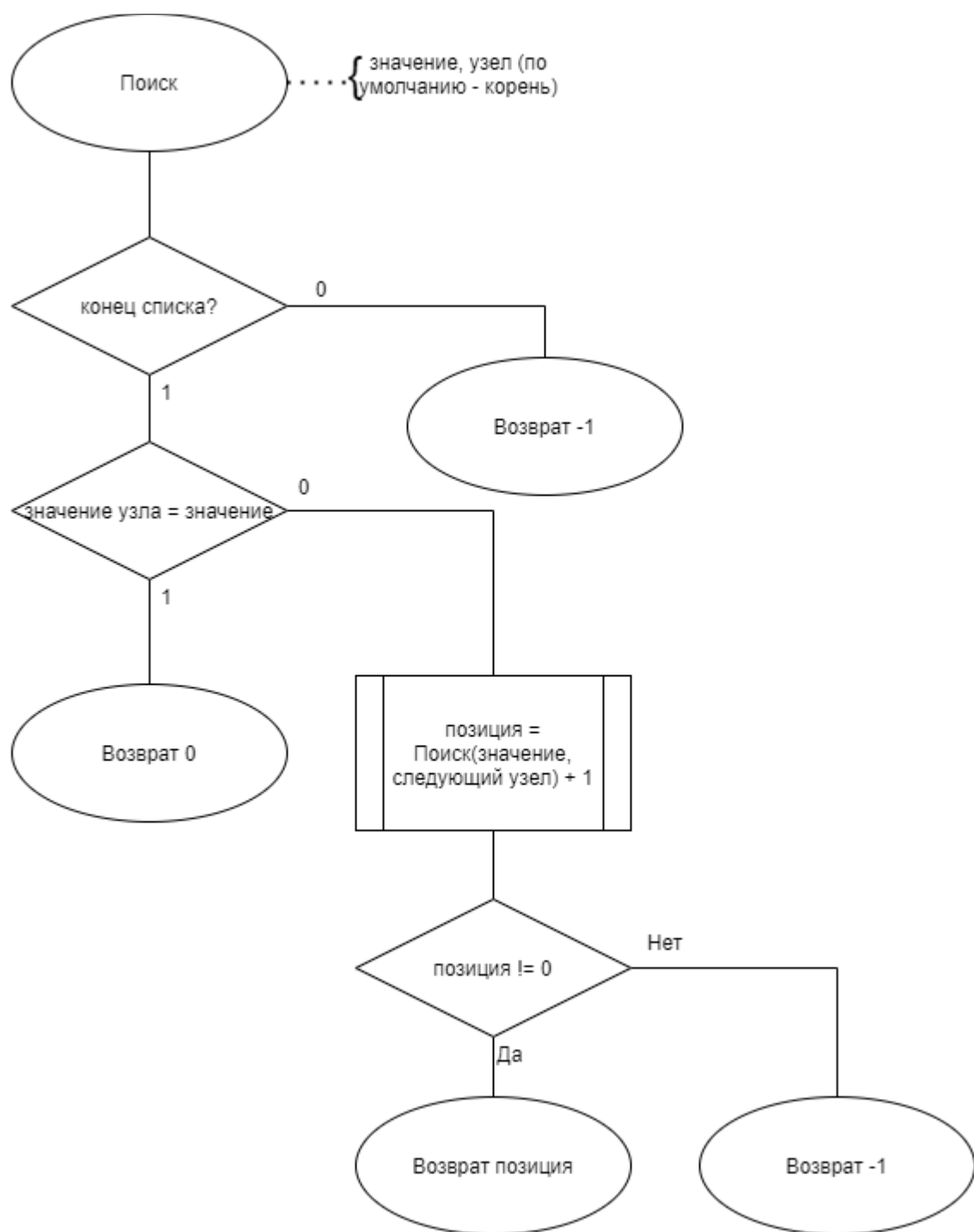


Рисунок 3 – Блок-схемы стека.









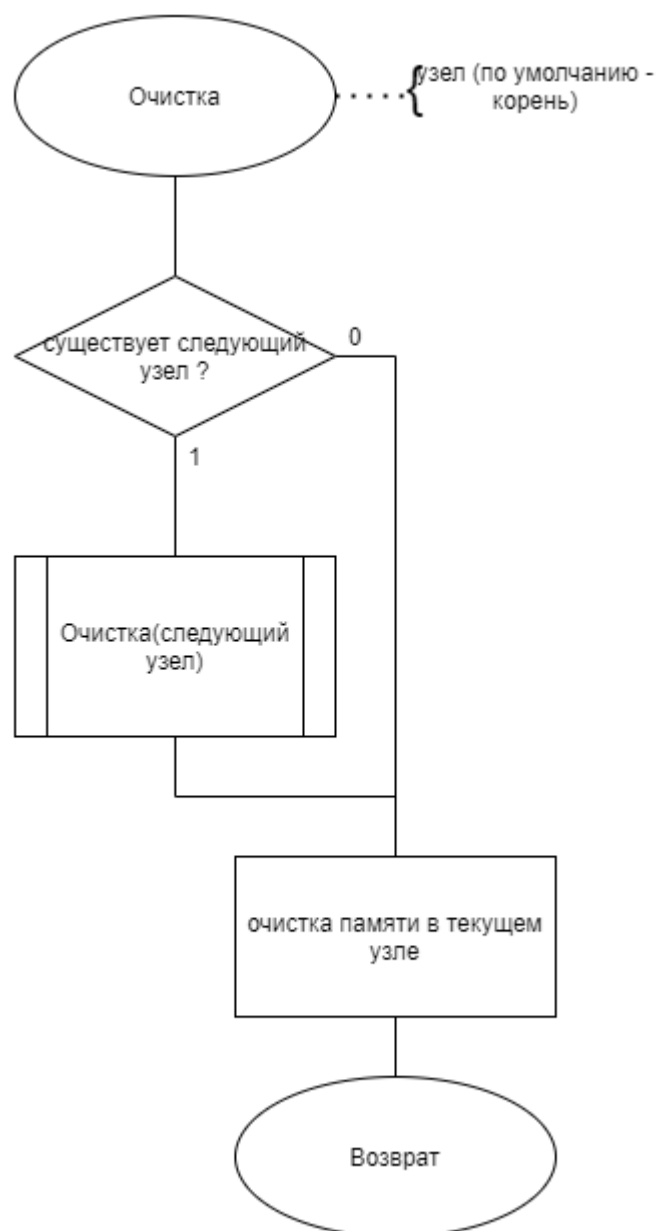


Рисунок 4 – Блок-схемы списка.

## Текст программы:

### List.js

```
class List {
  constructor() {
    this.root = {
      symbol: '',
      next: null
    }
  }

  insert(x, currentNode = this.root) {
    if (currentNode.next && currentNode.symbol < x ) {
      this.insert( x, currentNode.next )
    } else {
      let node = {
        symbol: x,
        next: currentNode.next
      }
      currentNode.next = node
    }
  }

  delete(x, currentNode = this.root) {
    if ( currentNode ) {
      if ( currentNode && currentNode.symbol <= x ) {
        if ( currentNode.symbol === x ) {
          currentNode.symbol = currentNode.next.symbol
          currentNode.next = currentNode.next.next
          this.delete( x, currentNode )
        }
        this.delete( x, currentNode.next )
      }
    }
  }

  find( x, currentNode = this.root ) {
    if ( currentNode ) {
      if ( currentNode.symbol === x ) {
        return 0;
      }
      let position = this.find(x, currentNode.next) + 1
      return position ? position : -1
    }
  }

  clear() {
    this.root.next = null
  }
}

module.exports = List
```

## Stack.js

```
class Stack {
  constructor() {
    this.size = 10;
    this.array = new Array(this.size);
    this.current = -1;
  }

  push(x) {
    if (this.current === this.size) {
      return false;
    }
    this.current++;
    this.array[this.current] = x;
    return true;
  }

  pop() {
    if (this.current < 0) {
      return -1;
    }

    this.array[this.current] = 0;
    this.current--;
  }

  top() {
    if (this.current < 0) {
      return -1;
    }

    return this.array[this.current];
  }


  clear() {
    for (; this.current > -1; this.current--) this.array[this.current] = 0;
  }

  isEmpty() {
    if (this.current === -1) return true;
    return false;
  }
}

module.exports = Stack;
```

## Описание работы программы с интерфейсом:

Верхняя часть интерфейса предназначена для работы со стеком, нижняя — со списком.

 Document — □ ×

Работа со стеком

Стек: [ "6", "5", "1", "4", "8" ].

Результат: false.

Добавить элемент

Удалить последний элемент

Стек пустой?

Очистить стек

Вывести стек

Показать верхний элемент

Работа со списком

Список: { "symbol": "", "next": { "symbol": "A", "next": { "symbol": "B", "next": { "symbol": "C", "next": null } } } }.

Результат: 2.

Добавить элемент

Найти номер позиции

Удалить со значением

Очистить список

Вывести список

## **Вывод:**

В ходе лабораторной работы были изучены следующие структуры данных: стек (абстрактный тип данных, используемый в большинстве языков программирования, особенностью стека является модель, при которой взаимодействия происходят с последним элементом) и линейный односвязный список (динамическая структура данных, в которой каждый элемент связан со следующим элементом с помощью специального указателя). Также на практике были закреплены навыки программирования данных структур.