*Федеральное государственное бюджетное образовательное учреждение* *высшего образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(национальный исследовательский университет)*** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА Компьютерные Системы и сети (ИУ6)

**Отчет**

**по лабораторной работе №9**

**варианта №7**

**Название лабораторной работы:**

Динамические структуры данных. Списки

**Дисциплина:**

Основы программирования

Студент гр. **ИУ6-12** **08/12/16\_\_\_\_\_\_\_\_\_\_ Векшин Роман**

(Подпись, дата) (И.О. Фамилия)

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Черноусова Татьяна Геннадьевна**

(Подпись, дата) (И.О. Фамилия)

**Задание:**

С клавиатуры вводятся записи, содержащие сведения о проживающих в гостинице (номер комнаты и фамилия). Сформировать и вывести список проживающих, упорядоченный по номерам комнат. Добавить новую запись, введенную с клавиатуры.

**Схема алгоритма:**

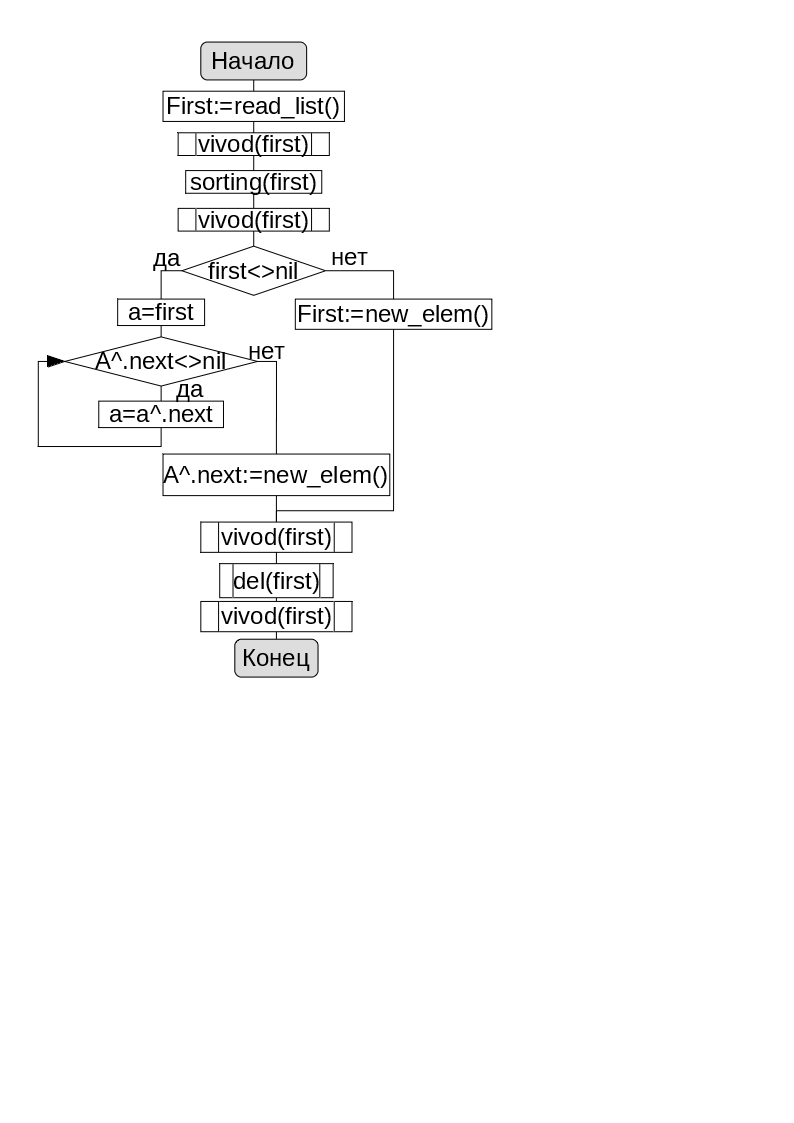


Рисунок -Схема алгоритма

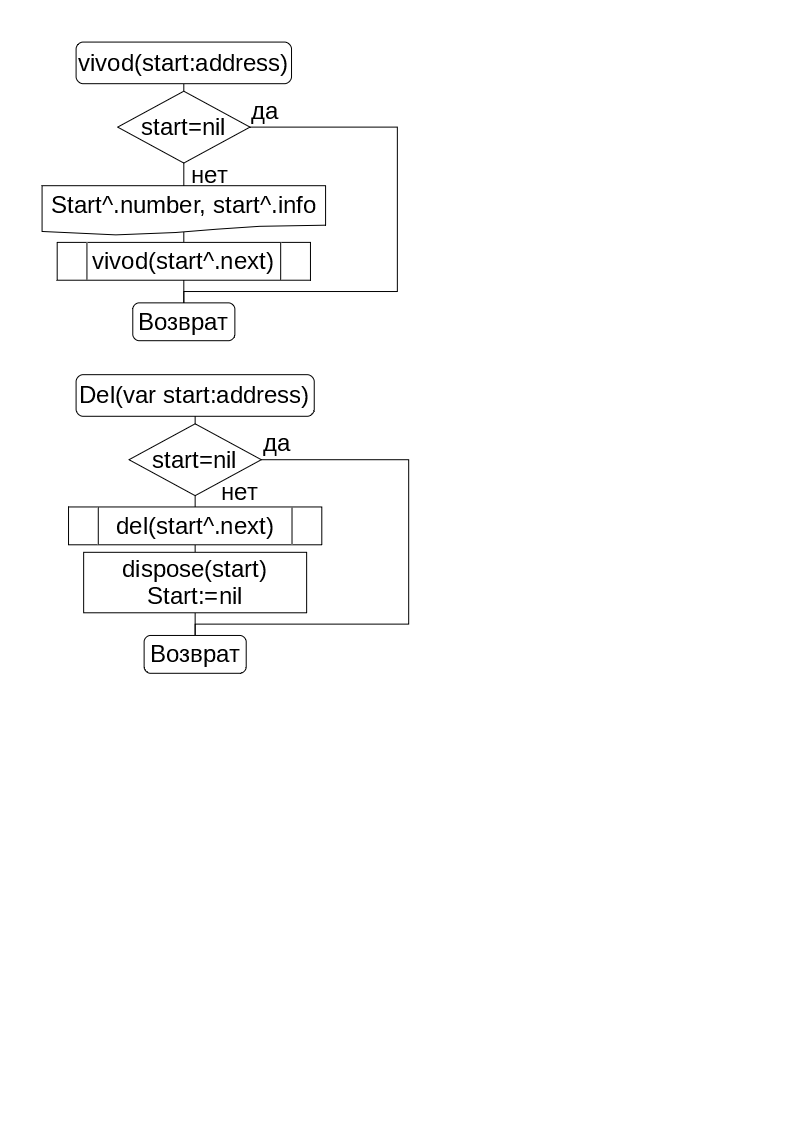


Рисунок -Схема алгоритма

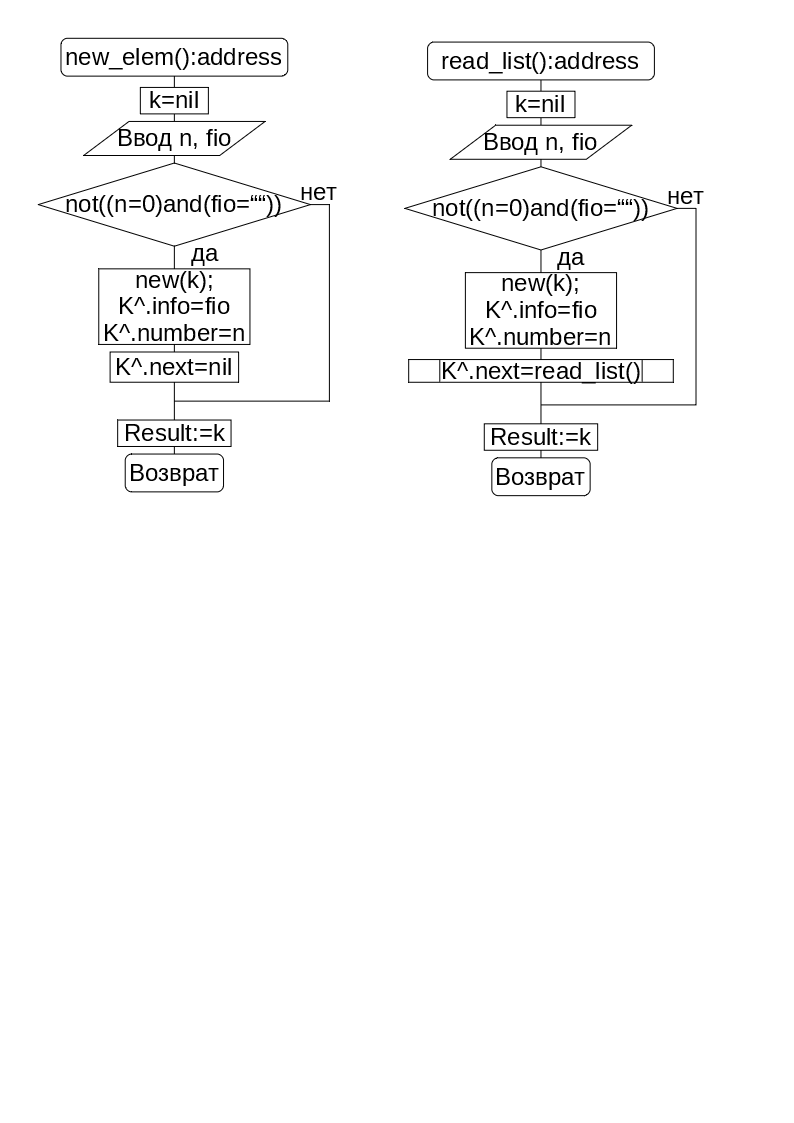


Рисунок -Схема алгоритма

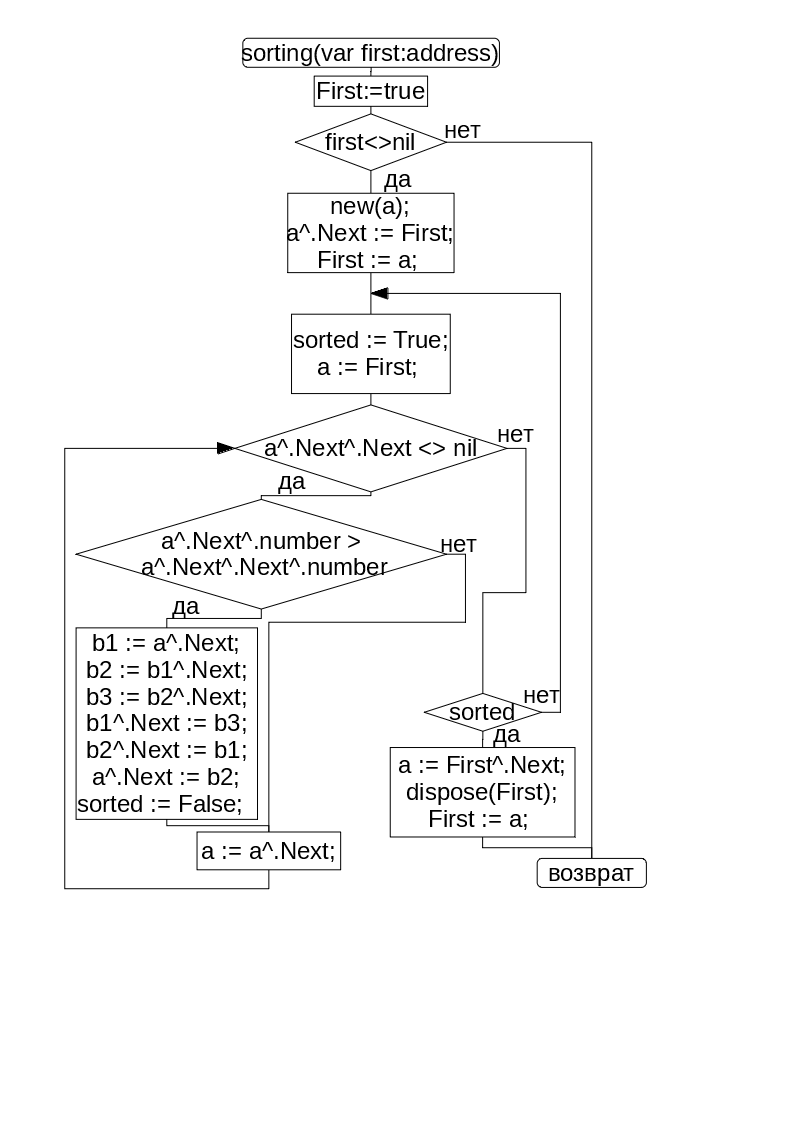


Рисунок -Схема алгоритма

**Описание используемых структур:**

Таблица -Структура данных

|  |  |
| --- | --- |
| **card** | |
| ИМЯ | ТИП |
| number | Integer |
| info | string |
| Next | address=^card |

**Код программы:**

program project1;

{$APPTYPE CONSOLE}

type

address = ^card;

card = record

number: integer;

info: string;

Next: address;

end;

**//Рекурсивный вывод списка, начиная с элемента на адресе start**

procedure vivod(const start: address);

begin

if not (start = nil) then

begin

writeln(start^.number, start^.info);

vivod(start^.Next);

end;

end;

**//Рекурсивное удаление списка, начиная с элемента на адресе start**

procedure del(var start: address);

begin

if not (start = nil) then

begin

del(start^.Next);

dispose(start);

start := nil;

end;

end;

**//Сортировка списка**

procedure sorting(var First: address);

var

a, b1, b2, b3: address;

sorted: boolean = True;

begin

if First <> nil then

begin

new(a);

a^.Next := First;

First := a;

repeat

sorted := True;

a := First;

while (a^.Next^.Next <> nil) do

begin

if a^.Next^.number > a^.Next^.Next^.number then

begin

b1 := a^.Next;

b2 := b1^.Next;

b3 := b2^.Next;

b1^.Next := b3;

b2^.Next := b1;

a^.Next := b2;

sorted := False;

end;

a := a^.Next;

end;

until sorted;

a := First^.Next;

dispose(First);

First := a;

end;

end;

function new\_elem(): address;

var

k: address = nil;

n: integer;

fio: string;

begin

readln(n, fio);

if not ((n = 0) and (fio = '')) then

begin

new(k);

k^.info := fio;

k^.number := n;

k^.Next := nil;

end;

Result := k;

end;

**//Ввод списка, рекурсивный**

function read\_list(): address;

var

k: address = nil;

n: integer;

fio: string;

begin

readln(n, fio);

if not ((n = 0) and (fio = '')) then

begin

new(k);

k^.info := fio;

k^.number := n;

k^.Next := read\_list();

end;

Result := k;

end;

var

First, a: address;

begin

writeln('enter the list; end - 0');

First := read\_list();

writeln(('entered list'));

vivod(First);

sorting(First);

writeln('sorted list');

vivod(First);

if First <> nil then

begin

a := First;

while a^.Next <> nil do

a := a^.Next;

writeln('enter an element to add');

a^.Next := new\_elem;

end

else

First := new\_elem;

writeln('new list');

vivod(First);

writeln(('Removing list'));

del(First);

vivod(First);

writeln('End. Press any key');

readln();

end.

**Тестирование:**

Таблица -Результаты тестирования программы 1

|  |  |  |
| --- | --- | --- |
| Исходные данные | Ожидаемый результат | Полученный результат |
| 1 unit1  2 unit2  3 unit3  4 unit4  0  0 unit 5 | 1 unit1  2 unit2  3 unit3  4 unit4  1 unit1  2 unit2  3 unit3  4 unit4  0 unit5 | 1 unit1  2 unit2  3 unit3  4 unit4  1 unit1  2 unit2  3 unit3  4 unit4  0 unit5 |
| 5 unit5  4 unit4  3 unit3  0  9 unit2 | 3 unit3  4 unit4  5 unit5  3 unit3  4 unit4  5 unit5  9 unit2 | 3 unit3  4 unit4  5 unit5  3 unit3  4 unit4  5 unit5  9 unit2 |
| 3 unit3  5 unit5  4 unit4  9 unit8  0  0 unit0 | 3 unit3  4 unit4  5 unit5  9 unit8  0 unit0 | 3 unit3  4 unit4  5 unit5  9 unit8  3 unit3  4 unit4  5 unit5  9 unit8  0 unit0 |
| 0  0 unit0 | 0 unit0 | 0 unit0 |

**Пример работы программы:**

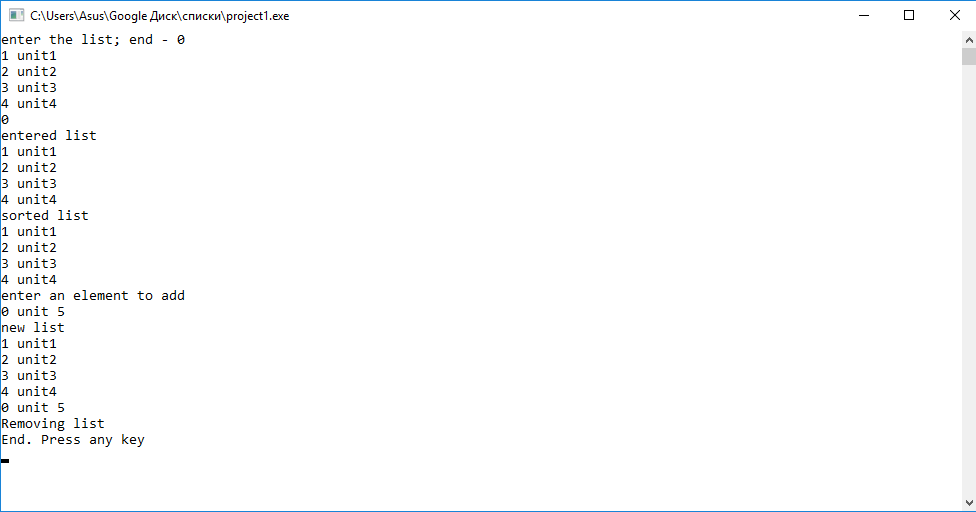


Рисунок -Пример работы программы

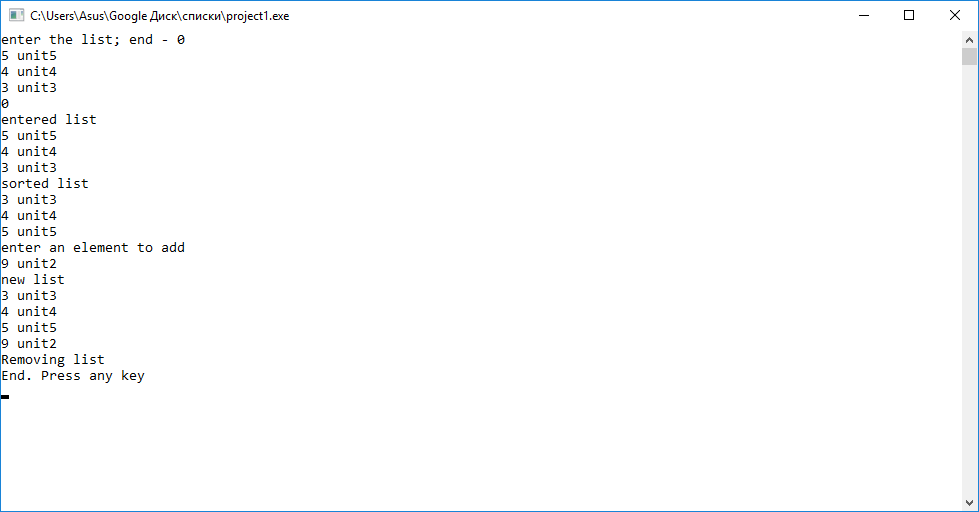


Рисунок -Пример работы программы

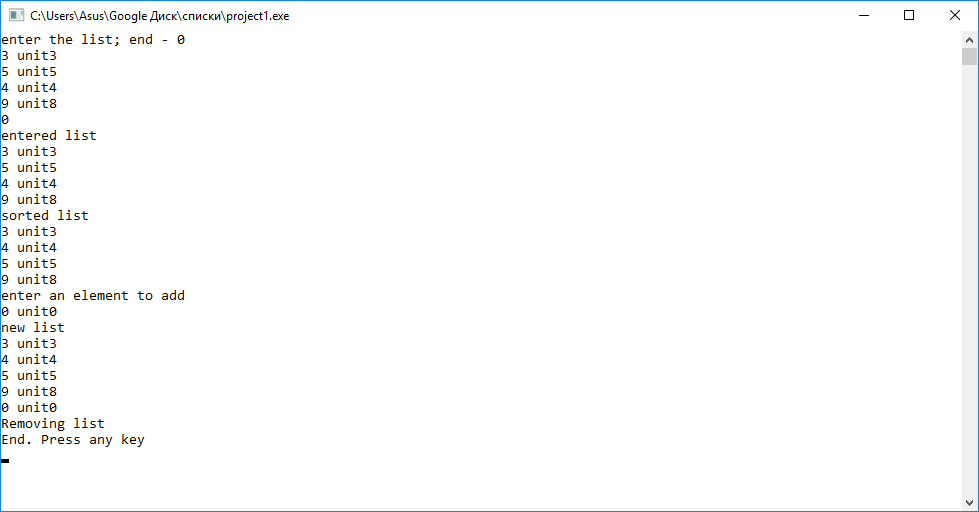


Рисунок -Пример работы программы

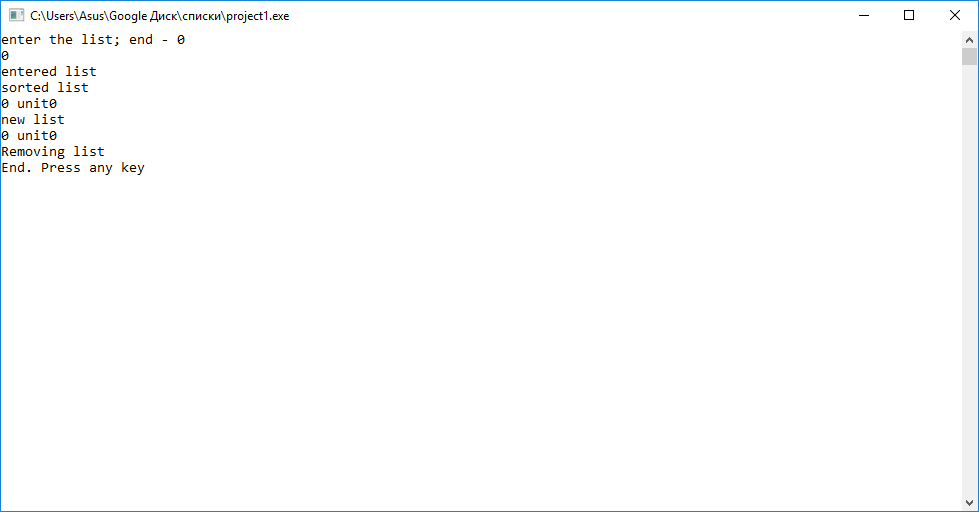


Рисунок -Пример работы программы

**Вывод:**

1. Разработан алгоритм программы, которая создает, сортирует, дополняет и удаляет список сведений о проживающих в гостинице, и составлена его схема в среде LibreOffice Draw(см. рис. 1-4).
2. Создан код программы по схеме алгоритма в среде Lazarus
3. Проведено тестирование (см. табл. 2).
4. Тестирование показало корректность работы программы (см. рис. 5-8)