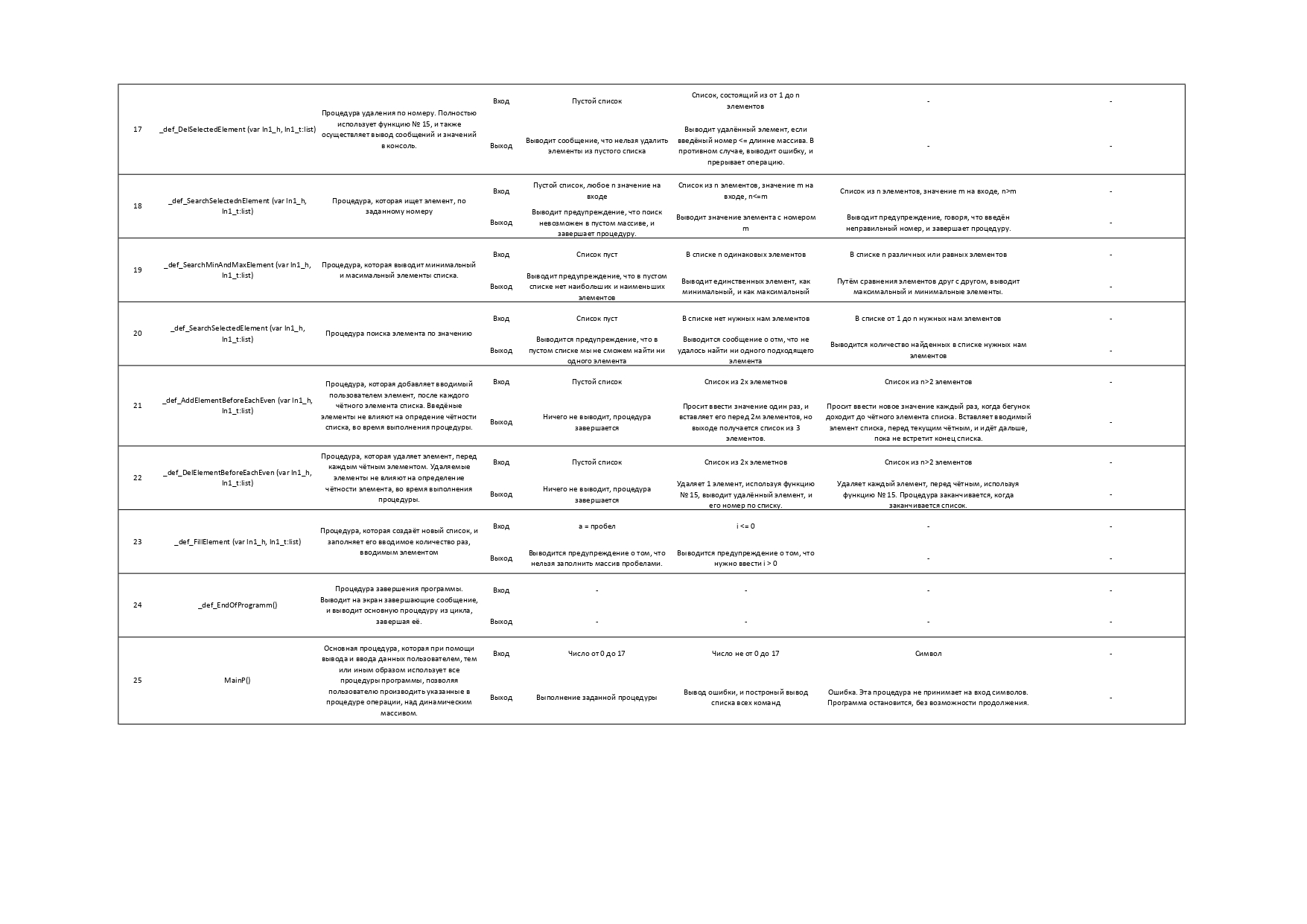
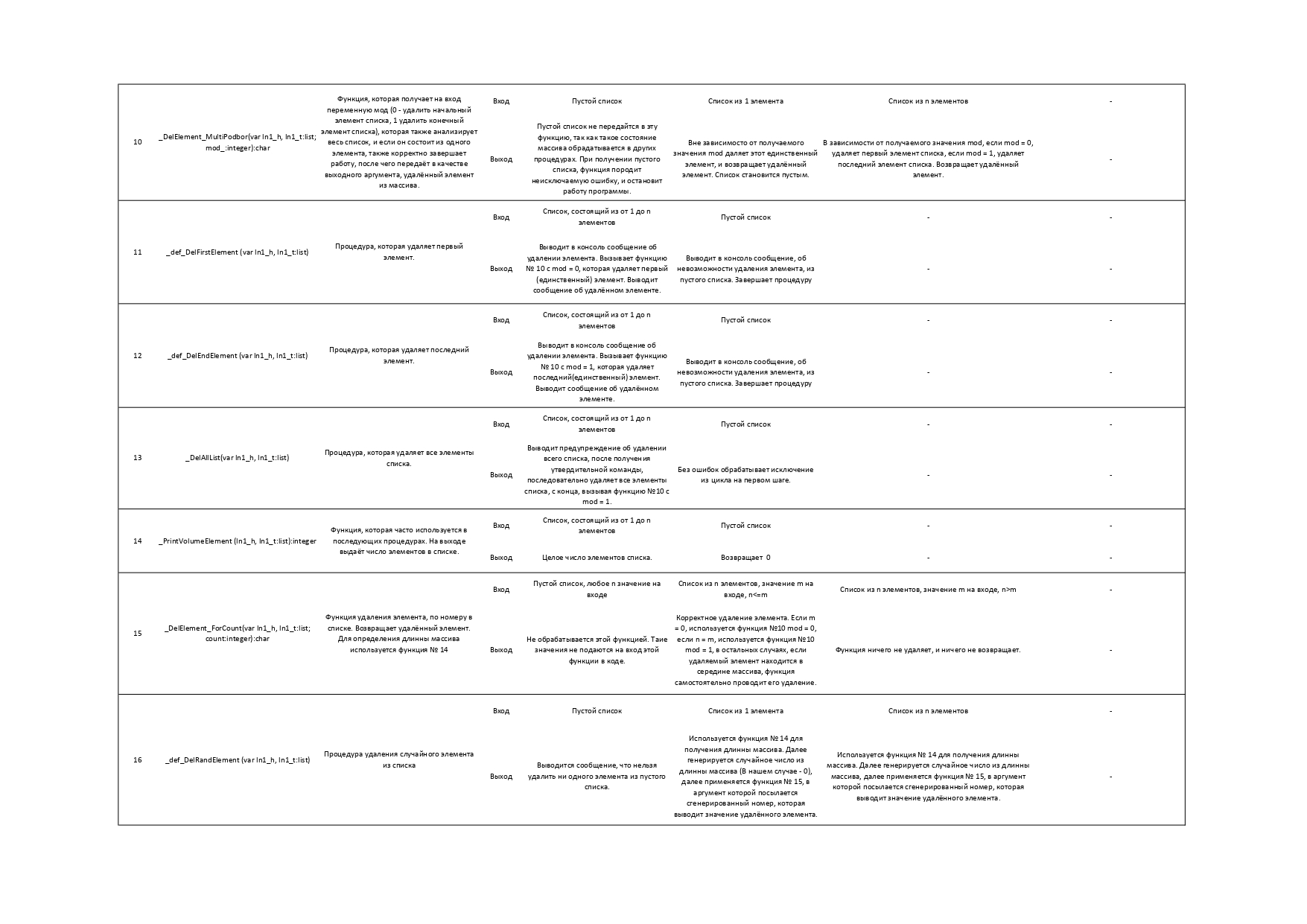
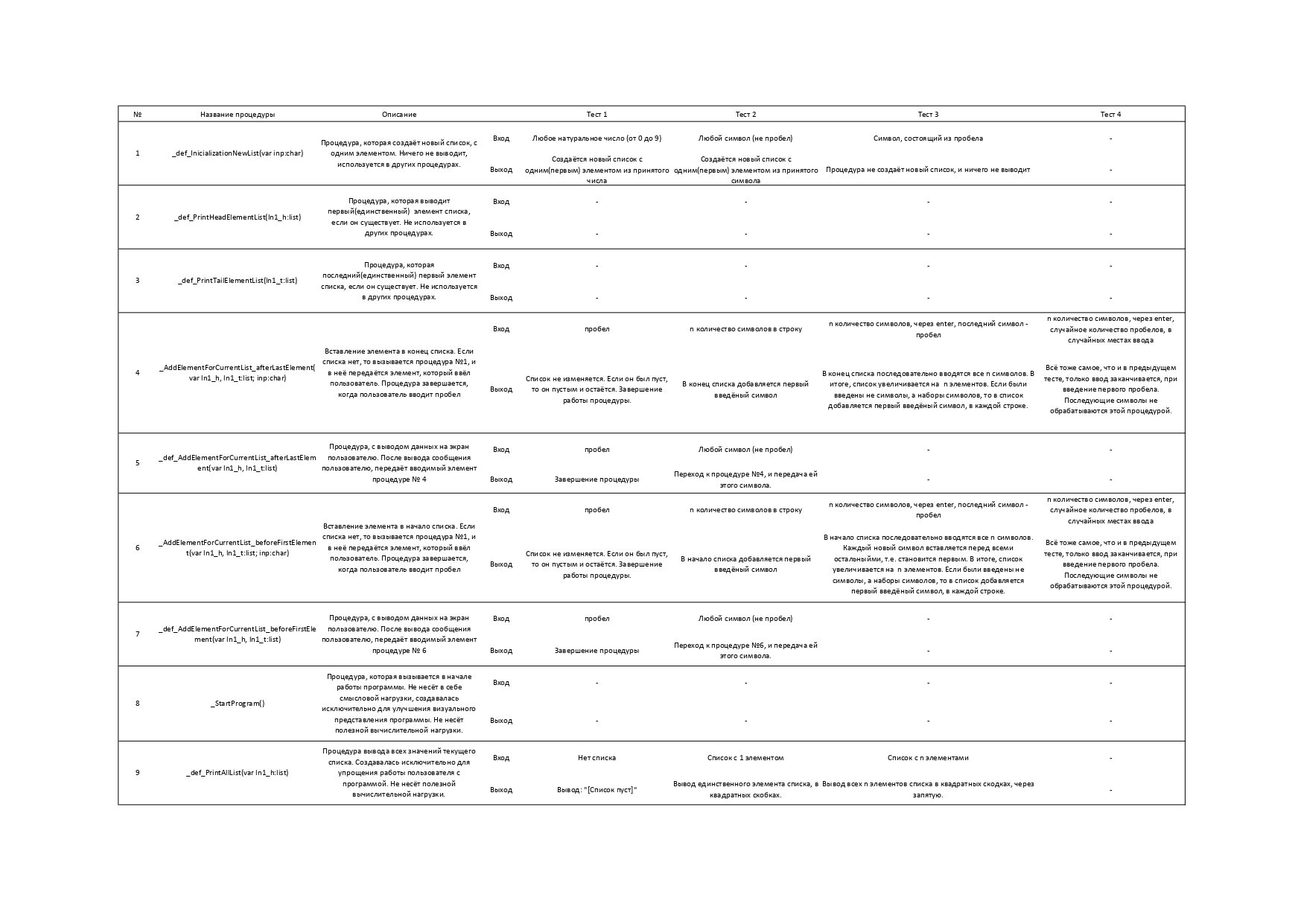
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ | | | | | | | | | | | | |
| Федеральное государственное автономное образовательное учреждение  высшего образования | | | | | | | | | | | | |
| **Дальневосточный федеральный университет** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **Кафедра прикладной математики, механики, управления и программного обеспечения** | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| **О Т Ч Е Т** | | | | | | | | | | | | |
| по лабораторной работе №1.1  дисциплина «Фундаментальные структуры данных и алгоритмы» | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | |  |  | Студент гр. Б9120-09.03.04прогин | | | |
|  |  |  | | О. Г. Александрович | |
|  | | | | | | |  |  | (подпись) | |  | |
|  | | | | | | | | | | | | |
|  | | | | | | |  |  | Руководитель | | |  |
|  | | | | | | |  |  | ст. преподаватель | | | |
|  | | | |  |  | |  |  |  |  | О.А. Крестникова | |
|  | | | |  |  | |  |  | (подпись) |  | (И.О. Фамилия) | |
|  |  |  |  | | |  |  |  |  | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
|  | | | | | | | | | | | | |
| г. Владивосток | | | | | | | | | | | | |
| 2021 | | | | | | | | | | | | |

**1 Неформальная постановка задачи**

Реализован двусвязный не кольцевой список, который содержит информацию, принадлежащую множеству символов. Эта программа выполняет такие операции:

* Выводит весь список
* Добавляет элемент в конец списка
* Добавляет элемент в начало списка
* Выводит начальный элемент
* Выводит последний элемент
* Удаляет первый элемент
* Удаляет последний элемент
* Удаляет весь список
* Удаляет случайный по номеру элемент списка
* Удаляет элемент списка по номеру
* Находит элемент по заданному номеру в списке
* Выводит минимальный и максимальный элементы
* Выводит элементы, которые равны введённому
* Добавляет элемент перед каждым чётным по номеру элементу
* Удаляет элемент, перед каждым чётным по номеру элементом
* Заполняет весь список указанным элементом указанное количество раз
* Выводит количество элементов в списке
* Завершает программу

**2 Описание процедур, и тесты**



\* Вход - это не входные данные и аргументы процедуры, а модуляция возможных последовательностей ввода данных пользователем, различные аргументы, передаваемые и обрабатываемы в других процедурах, а также любые возможные вариации состояния как отдельно взятого количества требуемых процедур, так и всей программы в целом.

**3 Текст программы**

**uses** crt;

**Type** list = ^TList;

TList = **record**

data:char; //Значение внутри ячейки списка

next:list;

prev:list;

**end**;

**var**

ln1\_h, ln1\_t:list; // ln1\_h - Голова списка // ln1\_t - Хвост списка // ln - это сокрашщение от "List", а h - Head(голова), t - Tail(хвост)

\_f,\_h:boolean;

**procedure** \_def\_InicializationNewList(**var** inp:char);

**var**

newData:list; // data - новая ячейка памяти

**begin**

**if**(inp <> ' ') **then**

**begin**

**new** (newData);

newData^.data := inp; // Или любое другое входящие значение

newData^.next := nil;

newData^.prev := nil;

ln1\_h := newData;

ln1\_t := newData;

**end**;

**end**;

**procedure** \_def\_PrintHeadElementList(ln1\_h:list); // Так как эти процедуры не используются в других процедурах, я решил их не изменять.

**begin**

writeln('Выводим первый элемент:');

**if**(ln1\_h <> nil) **then** writeln('list\_First = ', ln1\_h^.data);

**if**(ln1\_h = nil) **then** writeln('[Список пуст, нет первого элемента]');

**end**;

**procedure** \_def\_PrintTailElementList(ln1\_t:list); // Так как эти процедуры не используются в других процедурах, я решил их не изменять.

**begin**

writeln('Выводим последний элемент: ');

**if**(ln1\_h <> nil) **then** writeln('list\_End = ', ln1\_t^.data);

**if**(ln1\_h = nil) **then** writeln('[Список пуст, нет последнего элемента]');

**end**;

**procedure** \_AddElementForCurrentList\_afterLastElement(**var** ln1\_h, ln1\_t:list; inp:char); // Вставление элемента в текущий список, после последнего.

**var**

u,newData:list; // u - бегунок для прохождения списка

**begin**

u:=ln1\_t;

**if** (inp <> ' ') **then**

**begin**

**if**(ln1\_h = nil) **then**

**begin**

\_def\_InicializationNewList(inp);

u:=ln1\_t;

readln(inp);

**end**;

**while** (inp <> ' ') **do**

**begin**

**new**(newData);

newData^.data:=inp;

newData^.next:=nil;

newData^.prev:=u;

u^.next:=newData;

ln1\_t:=newData;

u:=ln1\_t;

readln(inp);

**end**;

**end**;

**end**;

**procedure** \_def\_AddElementForCurrentList\_afterLastElement(**var** ln1\_h, ln1\_t:list); // Вставление элемента в текущий список, после последнего.

**var**

inp:char;

**begin**

writeln('Добавление элементов в конец списка.');

writeln('Вводите элементы списка, через Enter. Конец ввода - пробел');

readln(inp);

\_AddElementForCurrentList\_afterLastElement (ln1\_h, ln1\_t, inp);

**end**;

**procedure** \_StartProgram(); // Процедура начала программы, с выводом "Loading %"

**var**

i:integer;

**begin**

writeln('Start Loading Program');

i:=0;

**while** (i<101) **do**

**begin**

writeln('Loading ', i, '%');

i:=i+1;

sleep(10);

**end**;

writeln('Loading complite');

writeln(' ');

**end**;

**procedure** \_def\_PrintAllList(**var** ln1\_h:list); // Эта процедура используется только для визуального представления списка пользователю. Внутри других процедур она не используется.

**var**

u:list;

**begin**

writeln('Выводим весь список:');

u:=ln1\_h;

write('list = [');

**if**(u<>nil) **then**

**begin**

write(u^.data);

u:=u^.next

**end**;

**while**(u<>nil) **do**

**begin**

write(', ', u^.data);

u:=u^.next;

**end**;

writeln('];');

**if**(ln1\_h = nil) **then** writeln('[Список пуст]');

**end**;

**procedure** \_AddElementForCurrentList\_beforeFirstElement(**var** ln1\_h, ln1\_t:list; inp:char); // Вставление элемента в текущий список, перед первым.

**var**

u,newData:list; // u - бегунок для прохождения списка

**begin**

u:=ln1\_h;

**if** (inp <> ' ') **then**

**begin**

**if**(ln1\_h = nil) **then**

**begin**

\_def\_InicializationNewList(inp);

u:=ln1\_h;

readln(inp);

**end**;

**while** (inp <> ' ') **do**

**begin**

**new**(newData);

newData^.data:=inp;

newData^.next:=u;

newData^.prev:=nil;

u^.prev:=newData;

ln1\_h:=newData;

u:=ln1\_h;

readln(inp);

**end**;

**end**;

**end**;

**procedure** \_def\_AddElementForCurrentList\_beforeFirstElement(**var** ln1\_h, ln1\_t:list); // Вставление элемента в текущий список, перед первым.

**var**

inp:char;

**begin**

writeln('Добавление элементов в начало списка.');

writeln('Вводите элементы списка, через Enter. Конец ввода - пробел');

readln(inp);

\_AddElementForCurrentList\_beforeFirstElement(ln1\_h, ln1\_t, inp);

**end**;

**function** \_DelElement\_MultiPodbor(**var** ln1\_h, ln1\_t:list; mod\_:integer):char; // Модифицированная мультиоперация удаления элемента, которая возвращает удалённый элемент

**var**

u,f:list;

**begin**

**if**(ln1\_h <> ln1\_t) **then**

**begin**

**if**(mod\_ = 0) **then** // Удаление элемента с начала списка

**begin**

u:=ln1\_h;

f:=ln1\_h;

u:=u^.next;

ln1\_h:=u;

u^.prev:=nil;

f^.next:=nil;

\_DelElement\_MultiPodbor := f^.data;

dispose(f);

**end**

**else if**(mod\_ = 1) **then** // Удаление элемента с конца списка

**begin**

u:=ln1\_t;

f:=ln1\_t;

u:=u^.prev;

ln1\_t:=u;

u^.next:=nil;

f^.prev:=nil;

\_DelElement\_MultiPodbor := f^.data;

dispose(f);

**end**;

**end**

**else begin** // Удаление последнего элемента, если список состоит только из одного элемента

u:=ln1\_h;

ln1\_h:=nil;

ln1\_t:=nil;

\_DelElement\_MultiPodbor := u^.data;

dispose(u);

**end**;

**end**;

**procedure** \_def\_DelFirstElement (**var** ln1\_h, ln1\_t:list); // Удаление первого элемента

**begin**

writeln('Операция удаления первого элемента.');

**if**(ln1\_h <> nil) **then**

**begin**

writeln('Мы удаляем элемент со значением "', \_DelElement\_MultiPodbor(ln1\_h, ln1\_t, 0),'"');

**end**

**else begin**

writeln('Список пуст, а значит удаление элемента весьма затруднительно, и скорее даже невозможно(');

**end**;

**end**;

**procedure** \_def\_DelEndElement (**var** ln1\_h, ln1\_t:list); // Удаление последнего элемента

**var**

u,f:list;

**begin**

writeln('Операция удаления последнего элемента.');

**if**(ln1\_t <> nil) **then**

**begin**

writeln('Мы удаляем элемент со значением "', \_DelElement\_MultiPodbor(ln1\_h, ln1\_t, 1),'"');

**end**

**else begin**

writeln('Список пуст, а значит удаление элемента весьма затруднительно, и скорее даже невозможно(');

**end**;

**end**;

**procedure** \_DelAllList(**var** ln1\_h, ln1\_t:list); // Очистка всего списка, последовательное удаление всех элементов с конца

**begin**

**while** (ln1\_t <> nil) **do**

**begin**

\_DelElement\_MultiPodbor(ln1\_h, ln1\_t, 1);

**end**;

**end**;

**procedure** \_def\_DelAllList (**var** ln1\_h, ln1\_t:list); // Удаление всего списка

**var**

y:char;

**begin**

writeln('Удаление всего списка.');

writeln('Вы действительно хотите удалить весь список? Подтвердите это, введя букву "y" y/n');

readln(y);

**if**(y = 'y') **then**

**begin**

\_DelAllList(ln1\_h, ln1\_t);

ln1\_h:=nil;

ln1\_t:=nil;

**end**

**else begin**

writeln('Введена не та буква. Операция отменена, но вы можете вызвать её повторно');

**end**;

**end**;

**function** \_PrintVolumeElement (ln1\_h, ln1\_t:list):integer; // Печать количества элементов в списке

**var**

i:integer;

u:list;

**begin**

u:=ln1\_h;

i:=0;

**while** (u<>nil) **do**

**begin**

i:=i+1;

u:=u^.next;

**end**;

\_PrintVolumeElement:=i;

**end**;

**procedure** \_def\_PrintVolumeElement (ln1\_h, ln1\_t:list); // Печать количества элементов в списке

**begin**

writeln('В списке ', \_PrintVolumeElement(ln1\_h, ln1\_t) ,' элементов');

**end**;

**function** \_DelElement\_ForCount(**var** ln1\_h, ln1\_t:list; count:integer):char; // Удаление элемента по номеру

**var**

u,f,d:list;

i,j:integer;

**begin**

i := \_PrintVolumeElement(ln1\_h, ln1\_t);

**if** count <= i **then**

**begin**

**if** count = 0 **then**

**begin**

\_DelElement\_ForCount:=\_DelElement\_MultiPodbor(ln1\_h, ln1\_t, 0);

**end**

**else if** count = i-1 **then**

**begin**

\_DelElement\_ForCount:=\_DelElement\_MultiPodbor(ln1\_h, ln1\_t, 1);

**end**

**else begin**

u:=ln1\_h;

j:=0;

**while**(j <> count) **do**

**begin**

j:=j+1;

u:=u^.next;

**end**;

f:=u^.prev;

d:=u;

u:=u^.next;

d^.next:=nil;

d^.prev:=nil;

\_DelElement\_ForCount:=d^.data;

dispose(d);

f^.next:=u;

u^.prev:=f;

**end**;

**end**;

**end**;

**procedure** \_def\_DelRandElement (**var** ln1\_h, ln1\_t:list); // Удаление рандомного элемента

**var**

i,SelNmb:integer;

**begin**

i := \_PrintVolumeElement(ln1\_h, ln1\_t); // i - Количество элементов в списке

**if**(i<>0) **then**

**begin**

SelNmb:=random(i-1);

writeln('Случайное число от 0, до ', i-1, ' = ',SelNmb);

writeln('Мы удаляем элемент со значением "', \_DelElement\_ForCount(ln1\_h, ln1\_t, SelNmb),'"');

**end**

**else begin**

writeln('Список пуст, и из него нельзя удалить ни одного элемента.');

**end**;

**end**;

**procedure** \_def\_DelSelectedElement (**var** ln1\_h, ln1\_t:list); // Удаление элемента по номеру

**var**

i,j,r, SelNmb:integer;

u,f,d:list;

**begin**

i := \_PrintVolumeElement(ln1\_h, ln1\_t); // i - Количество элементов в списке

**if**(i<>0) **then**

**begin**

writeln('Введите любой номер от 0 до ',i-1);

r:=-1;

readln(SelNmb);

**if**(SelNmb<0) **or** (SelNmb>i-1) **then**

**begin**

writeln('Введён неверный номер. Операция будет прервана.');

**end**

**else begin**

writeln('Мы удаляем элемент со значением "', \_DelElement\_ForCount(ln1\_h, ln1\_t, SelNmb),'"');

**end**;

**end**

**else begin**

writeln('Список пуст, и из него нельзя удалить ни одного элемента.');

**end**;

**end**;

**procedure** \_def\_SearchSelectednElement (**var** ln1\_h, ln1\_t:list); // Эта процедура нужна только для визуального представления данных пользователю, и не используется больше нигде в коде.

**var**

i,inp, l:integer;

u:list;

**begin**

u:=ln1\_h;

i:=0;

**while** (u<>nil) **do**

**begin**

i:=i+1;

u:=u^.next;

**end**;

writeln('Ищем элемент по заданному номеру:');

writeln('Всего в списке ', i, ' элементов.');

**if**(i = 0) **then**

**begin**

writeln('Поиск элементов невозможен, в пустом списке(');

**end**

**else**

**begin**

write('Введите номер элемента: ');

readln(inp);

**if**(inp<1)**or** (inp>i) **then**

**begin**

writeln('Введено неправльное число. Операция будет прервана.');

**end**

**else begin**

l:=1;

u:=ln1\_h;

**while**(l<>inp) **do**

**begin**

l:=l+1;

u:=u^.next;

**end**;

writeln('Ура! Нашли тот самый элемент. Вот он: ', u^.data);

**end**;

**end**;

**end**;

**procedure** \_def\_SearchMinAndMaxElement (**var** ln1\_h, ln1\_t:list); // Эта процедура нужна только для визуального представления данных пользователю, и не используется больше нигде в коде.

**var**

i:integer;

min,max,f:char;

u:list;

**begin**

writeln('Выводим минимальный и максимальный элементы списка:');

u:=ln1\_h;

min := '0';

max:='0';

**while**(u<>nil) **do**

**begin**

**if**(u^.data < min) **then** min:=u^.data;

**if**(u^.data > max) **then** max:=u^.data;

u:=u^.next;

**end**;

u:=ln1\_h;

f:=max;

max:=min;

min:=f;

**while**(u<>nil) **do**

**begin**

**if**(u^.data < min) **then** min:=u^.data;

**if**(u^.data > max) **then** max:=u^.data;

u:=u^.next;

**end**;

**if**(ln1\_h = nil) **then**

**begin**

writeln('В пустом списке нет наименьших и наибольших элементов');

**end**

**else**

**begin**

writeln('min = ', min);

writeln('max = ', max);

**end**;

**end**;

**procedure** \_def\_SearchSelectedElement (**var** ln1\_h, ln1\_t:list); // Эта процедура нужна только для визуального представления данных пользователю, и не используется больше нигде в коде.

**var**

count:integer;

s:char;

u:list;

**begin**

writeln('Вас приветствует подпрограмма поиска!) ');

write('Введите элемент, который нужно найти: ');

readln(s);

**if**(ln1\_h = nil) **then**

**begin**

writeln('Этот список пуст. Я думаю, в нём не получится найти ни одного элемента(');

**end**

**else begin**

count:=0;

u:=ln1\_h;

**while**(u<> nil) **do**

**begin**

**if**(u^.data = s) **then** count:=count+1;

u:=u^.next;

**end**;

**if**(count = 0) **then**

**begin**

writeln('Ой, кажется не получилось найти ни одного такого элемента(');

**end**

**else begin**

writeln('Получилось найти ', count, 'шт. таких элементов');

**end**;

**end**;

**end**;

**procedure** \_def\_AddElementBeforeEachEven (**var** ln1\_h, ln1\_t:list);

**var**

u,f,n:list;

i:integer;

a:char;

**begin**

writeln('Добавление элементов перед каждым чётным');

u:=ln1\_h;

i:=1;

**while**(u<>nil)**do**

**begin**

**if**((i **mod** 2) = 0) **then**

**begin**

write('Введите добавляемый элемент: ');

readln(a);

writeln('Добавляем элемент ', a, ' на место под номером ', i);

**new**(n);

n^.data:=a;

f:=u^.prev;

n^.prev:=f;

n^.next:=u;

f^.next:=n;

u^.prev:=n;

**end**;

i:=i+1;

u:=u^.next;

**end**;

**end**;

**procedure** \_def\_DelElementBeforeEachEven (**var** ln1\_h, ln1\_t:list); // Удаление элементов перед каждым чётным

**var**

u:list;

i:integer;

**begin**

writeln('Удаление элементов перед каждым чётным');

u:=ln1\_h;

i:=0;

**while**(u<>nil)**do**

**begin**

**if**((i **mod** 2) = 1) **then**

**begin**

writeln('Мы удаляем элемент №',i-1,' со значением ', \_DelElement\_ForCount(ln1\_h, ln1\_t, i-1),' ');

**end**;

i:=i+1;

u:=u^.next;

**end**;

**end**;

**procedure** \_def\_FillElement (**var** ln1\_h, ln1\_t:list);

**var**

u,t:list;

i,j:integer;

a:char;

**begin**

writeln('В результате этой операции создатстя пустой новый список.');

write('Введите элемент, которым мы будем заполнять: '); readln(a);

**if**(a = ' ') **then**

**begin**

writeln('Нельзя заполнять список пробелами');

**end**

**else begin**

write('Сколько раз нужно заполнить этим элементом? ');readln(i);

**if**(i>0) **then**

**begin**

**new**(t);

t^.data:=a;

t^.prev:=nil;

t^.next:=nil;

ln1\_h:=t;

ln1\_t:=t;

j:=1;

u:=ln1\_h;

**while**(j<i) **do**

**begin**

**new**(t);

t^.data:=a;

t^.prev:=u;

u^.next:=t;

t^.next:=nil;

ln1\_t:=t;

j:=j+1;

u:=t;

**end**;

**end**

**else begin**

writeln('Введено некорректное количество заполнений. Попробуйте ввести число > 0', i);

**end**;

**end**;

**end**;

**procedure** \_def\_EndOfProgramm();

**begin**

writeln('Спасибо, что использовали программу по работе с динамическими массивами.');

writeln('Ждём вас снова)');

\_h:= true;

**end**;

**procedure** MainP();

**var**

i:integer;

**begin**

**if**(\_h = false) **then**

**begin**

**if**(\_f = false) **then**

**begin**

\_f:=true;

writeln('Вас приветствует программа "Обработки динамических массивов"!');

writeln('Созданная великим, и черезвычайно талантливым программистом с ником Gogik Ortey');

writeln('Конкретно эта программа работает только с двухсторонними некольцевыми списками.');

writeln('Также программа поддерживает все цифры, буквы и символы, вводимые с клавиатуры)');

**end**;

writeln(' ');

writeln('Для того, что бы совершить какую-то операцию, введи нужное число:');

writeln('0 - Вывести весь список');

writeln('1 - Добавить элемент в конец списка');

writeln('2 - Добавить элемент в начало списка');

writeln('3 - Вывесли начальный элемент списка');

writeln('4 - Вывести последний элемент списка');

writeln('5 - Удалить первый элемент списка');

writeln('6 - Удалить последний элемент списка');

writeln('7 - Удалить весь список');

writeln('8 - Удалить случайный по номеру элемент списка (и вывести его)');

writeln('9 - Удаление заданного по номеру элемента списка');

writeln('10 - Найти элемент по заданному номеру в списке');

writeln('11 - Вывести минимальный и максимальный элементы списка');

writeln('12 - Вывести элементы, которые равны введёному элементу (выведется их колличество в списке)');

writeln('13 - Добавление элемента, перед каждым чётным по номеру');

writeln('14 - Удаление элемента, перед каждым чётным по номеру');

writeln('15 - Заполнить новый список указанным элементом указанное колличество раз');

writeln('16 - Вывести количество элементов в списке');

writeln('17 - Завершить программу');

**while**(\_h = false) **do**

**begin**

readln(i);

**if**((i<0) **or** (i>17)) **then**

**begin**

writeln('Введено неправильное число. Пожалуйста, будь немного аккуратнее. Попробуй ещё раз, у тебя точно всё получится!)');

MainP();

**end**;

**case** i **of**

0: \_def\_PrintAllList(ln1\_h);

1: \_def\_AddElementForCurrentList\_afterLastElement(ln1\_h, ln1\_t);

2: \_def\_AddElementForCurrentList\_beforeFirstElement(ln1\_h, ln1\_t);

3: \_def\_PrintHeadElementList(ln1\_h);

4: \_def\_PrintTailElementList(ln1\_t);

5: \_def\_DelFirstElement (ln1\_h, ln1\_t);

6: \_def\_DelEndElement (ln1\_h, ln1\_t);

7: \_def\_DelAllList (ln1\_h, ln1\_t);

8: \_def\_DelRandElement (ln1\_h, ln1\_t);

9: \_def\_DelSelectedElement (ln1\_h, ln1\_t);

10: \_def\_SearchSelectednElement (ln1\_h, ln1\_t);

11: \_def\_SearchMinAndMaxElement (ln1\_h, ln1\_t);

12: \_def\_SearchSelectedElement (ln1\_h, ln1\_t);

13: \_def\_AddElementBeforeEachEven (ln1\_h, ln1\_t);

14: \_def\_DelElementBeforeEachEven (ln1\_h, ln1\_t);

15: \_def\_FillElement (ln1\_h, ln1\_t);

16: \_def\_PrintVolumeElement (ln1\_h, ln1\_t);

17: \_def\_EndOfProgramm();

**end**;

writeln('Операция завершена');

writeln('------------------');

**end**;

**end**;

**end**;

**begin**

\_h:=false;

\_StartProgram();

MainP();

**end**.

// [30.09.2021]

// https://vk.com/gog.ortey