

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ  
БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Отчёт по лабораторной работе №4

по дисциплине

«Программирование»

«Реализация элементарных структур данных на основе динамической памяти»

Выполнил студент гр. ИВТб-1303-06-00

\_\_\_\_\_ /Гортоломей И.К./

Проверил преподаватель кафедры ЭВМ

\_\_\_\_\_ /Баташев П.А./

Киров

2025

## **Цель**

Цель работы: освоение элементарных структур данных, закрепление навыков работы с динамической памятью.

## **Задание**

1. Реализовать программу взаимодействия со структурой данных. Язык программирования и вид структуры указан в варианте.
2. Структура данных должна быть реализована на основе динамической памяти.
3. Взаимодействие со структурой данных (создание и очистка структуры; вставка, чтение и удаление элемента; демонстрация всех элементов структуры с отображением их количества) должно происходить с помощью меню, навигация в котором происходит с помощью стрелок.
4. Так же меню должно содержать одну дополнительную возможность, указанную в варианте. При реализации дополнительной возможности прямой доступ к элементам структуры недопустим, разрешено взаимодействие со структурой только с использованием реализованных подпрограмм в пункте 3

## **Дано:**

Структура: Дек

Дополнительная возможность: Заполнить случайными элементами (количество и диапазон генерируемых значений указывает пользователь)

Язык: Pascal

# Решение

Схема алгоритма:

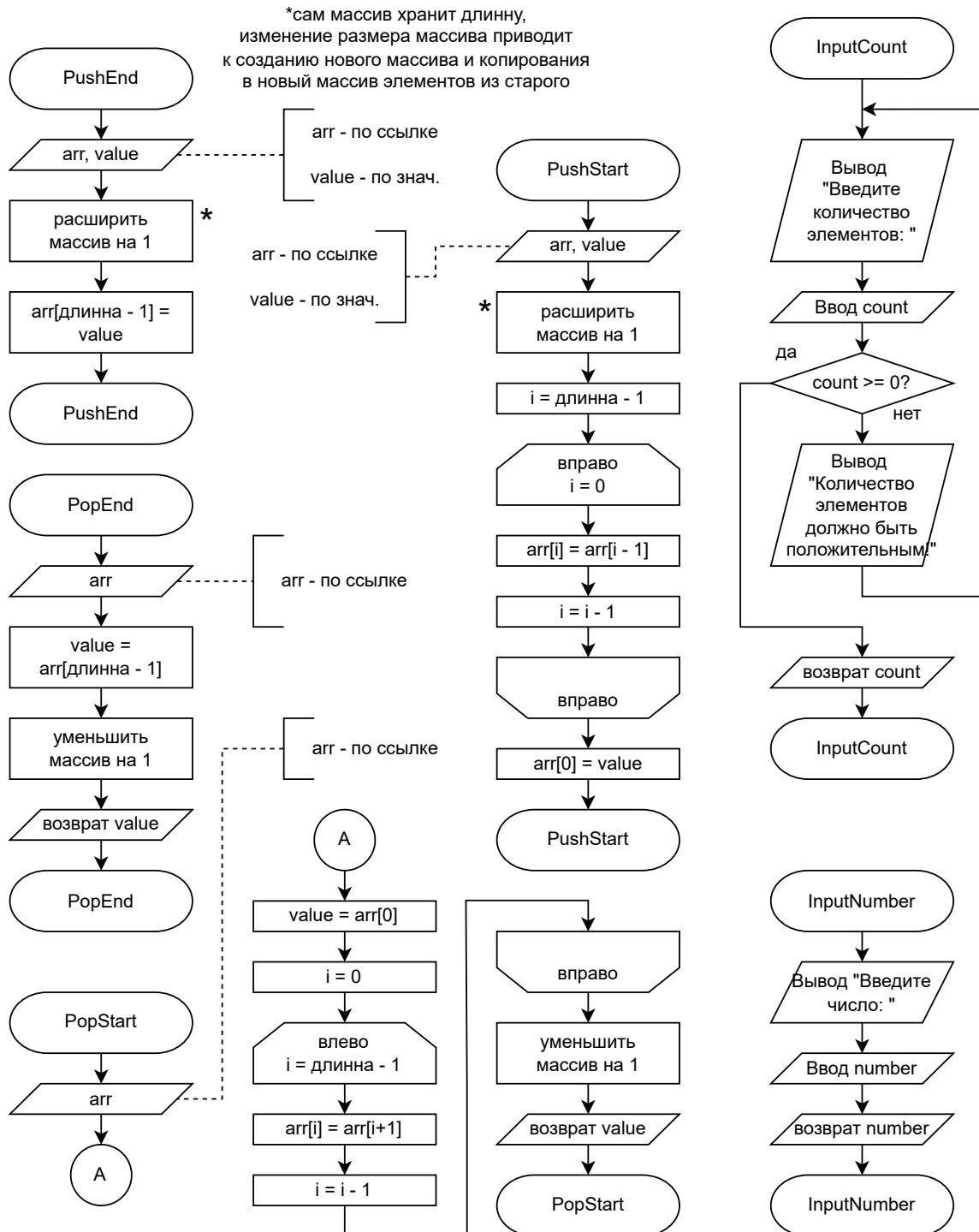


Рис. 1: функции и процедуры

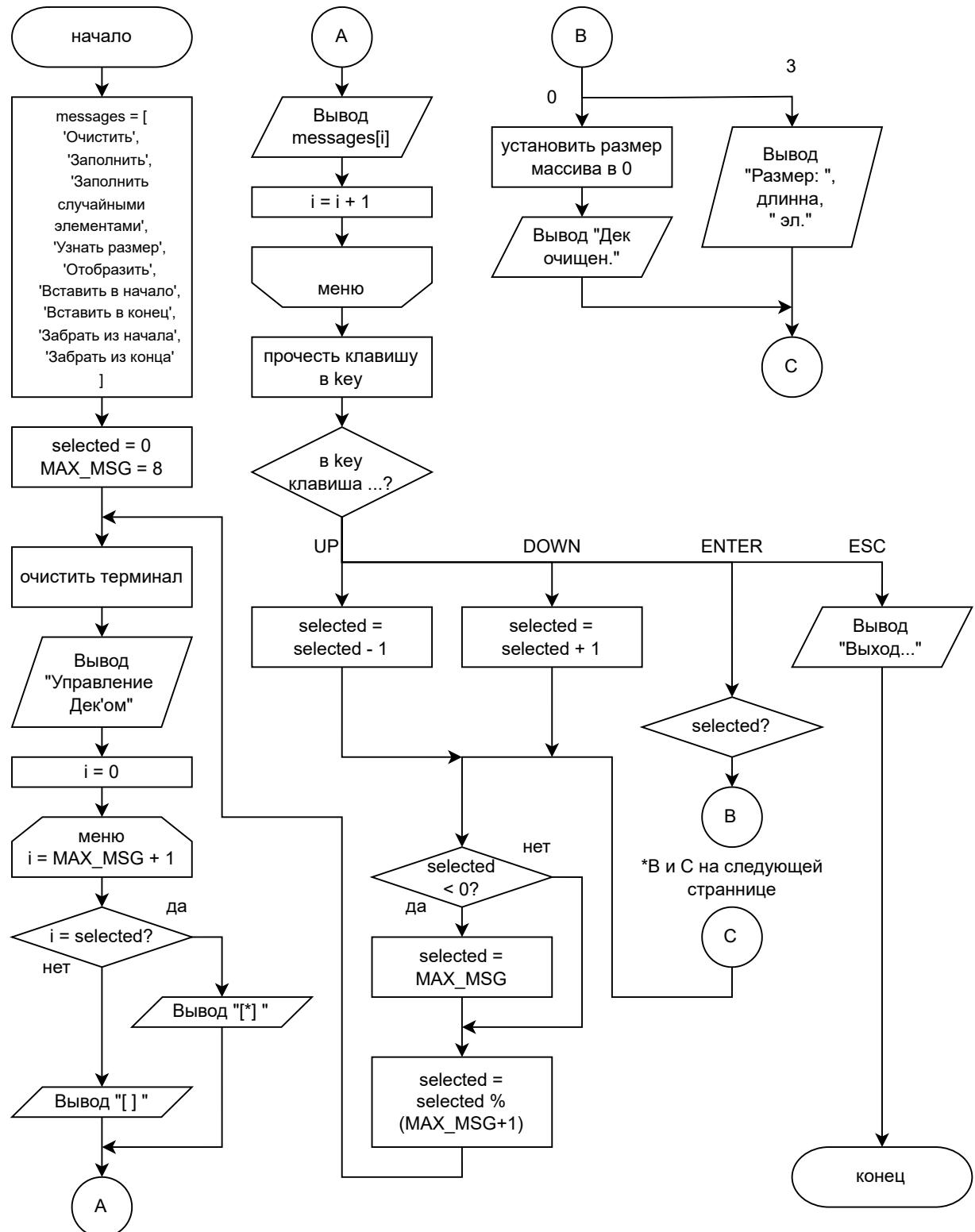
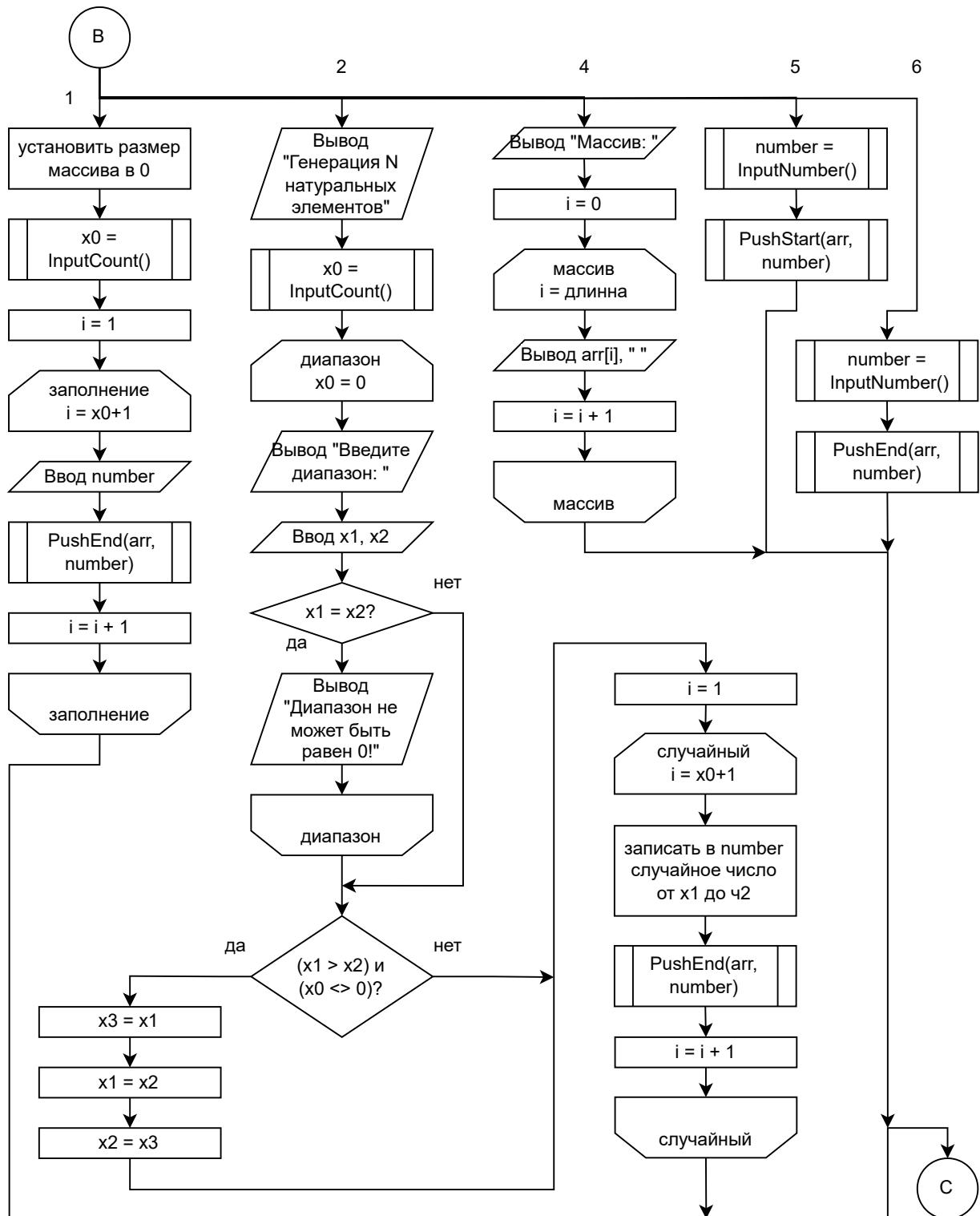


Рис. 2: основное тело программы ч.1



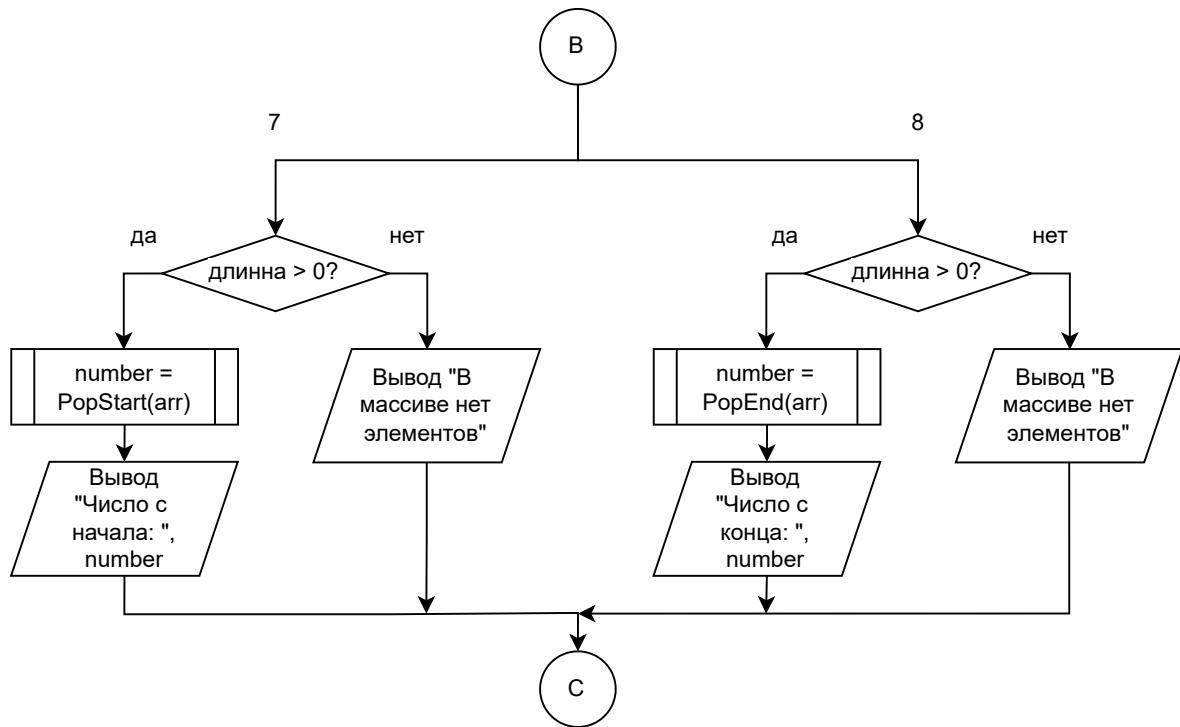


Рис. 4: основное тело программы ч.3

### Код программы на Pascal:

```

program Main;
uses Keyboard, SysUtils;

const
  MAX_MSG = 8;

type
  Element = record
    data: integer;
    prev: ^Element;
    next: ^Element;
  end;

// ##### ФУНКЦИИ И ПРОЦЕДУРЫ

```

```

// procedure PushEnd(var arr: dArray; value: integer);
// begin
//   SetLength(arr, Length(arr)+1);
//   arr[Length(arr)-1] := value;
// end;

// function PopEnd(var arr: dArray): integer;
// var
//   value: integer;
// begin
//   value := arr[Length(arr)-1];
//   SetLength(arr, Length(arr)-1);
//   PopEnd := value;
// end;

// procedure PushStart(var arr: dArray; value: integer);
// var
//   i: integer;
// begin
//   SetLength(arr, Length(arr)+1);
//   for i := Length(arr)-1 downto 1 do arr[i] := arr[i-1];
//   arr[0] := value;
// end;

// function PopStart(var arr: dArray): integer;
// var
//   value, i: integer;
// begin
//   value := arr[0];
//   for i := 0 to Length(arr)-2 do arr[i] := arr[i+1];
//   SetLength(arr, Length(arr)-1);
//   PopStart := value;
// end;

function InputCount(): integer;
var
  count: integer;

```

```

begin
repeat
  write('Введите количество элементов: ');
  readln(count);
  if count >= 0 then break;
  writeln('Количество элементов должно быть положительным!');
until false;
InputCount := count;
end;

function InputNumber(): integer;
var
  number: integer;
begin
  write('Введите число: ');
  readln(number);
  InputNumber := number;
end;
// #####



var
// #####
_key_event: TKeyEvent;
// #####



str: Element;
el: Element;
ptr1: ^Element;
ptr2: ^Element;

key: Word;
messages: array[0..MAX_MSG] of string = (
  'Очистить',
  'Заполнить',
  'Заполнить случайными элементами',
  'Узнать размер',
  'Отобразить',

```

```

'Вставить в начало',
'Вставить в конец',
'Забрать из начала',
'Забрать из конца'
);

selected, number: integer;
i, x0, x1, x2, x3: integer;

begin
// #####
Randomize;
InitKeyboard;
// #####
// #####
str.data := 0;
str.prev := nil;
str.next := nil;

selected := 0;
repeat
// #####
// очистить экран
Write(#27'[2J');
Write(#27'[1;1H');
// #####
// #####
writeln('Управление Дек''ом');
for i := 0 to MAX_MSG do begin
  if i = selected then
    write('* ')
  else
    write(' [ ] ');
  writeln(messages[i]);
end;

// #####
// прочесть клавишу в key
_key_event := GetKeyEvent;

```

```

_key_event := TranslateKeyEvent(_key_event);
key := GetKeyCode(_key_event);
// ##########

// е key клавиша ...?
case key of
  // ESC?
  283: begin writeln('Выход...'); break; end;
  // UP?
  65313: selected := selected - 1;
  // DOWN?
  65319: selected := selected + 1;
  // ENTER?
  7181: begin
    writeln;
    // selected?
    case selected of

      // 0: begin
      //   SetLength(arr, 0);
      //   writeln('Дек очищен.');
      // end;

      // 1: begin
      //   SetLength(arr, 0);
      //   x0 := InputCount();
      //   for i := 1 to x0 do begin
      //     read(number);
      //     PushEnd(arr, number);
      //   end;
      // end;

      // 2: begin
      //   writeln('Генерация N натуральных элементов');
      //   // write('Введите 0 для записи в конец, иначе в начало: ');
      //   // readln(x);

```

```

//  x0 := InputCount();

//  while x0 <> 0 do begin
//    write('Введите диапазон: ');
//    readln(x1, x2);
//    if x1 <> x2 then break;
//    writeln('Диапазон не может быть равен 0!');
//  end;

//  if (x1 > x2) and (x0 <> 0) then begin
//    x3 := x1;
//    x1 := x2;
//    x2 := x3;
//  end;

//  for i := 1 to x0 do begin
//    // записать в number случайное число от x1 до x2
//    number := Random(x2 - x1 + 1) + x1;
//    // if x = 0 then
//    //   PushEnd(arr, number)
//    // else
//    //   PushStart(arr, number);
//    PushEnd(arr, number);
//  end;
// end;

3: begin
  writeln('Размер: ', str.data, ' эл.');
end;

4: begin
  write('Массив: ');
  ptr1 := str.prev;
  while ptr1 <> nil do begin
    el := ptr1^;
    write(el.data, ' ');
    ptr1 := el.next;
  end;

```

```

    end;
end;

5: begin
    number := InputNumber();           // ввод
    new(ptr1);                      // выделение памяти
    (ptr1^).data := number;          // запись значения в A
    (ptr1^).prev := nil;             // в A ссылка на следующий
    (ptr1^).next := nil;             // в A ссылка на предыдущий
    if str.prev = nil then begin
        str.prev := ptr1;            // если дек пуст
        str.next := ptr1;            // в дек ссылку на начало
    end else begin
        (ptr1^).next := str.prev;     // в A ссылка на следующий B
        (str.prev^).prev := ptr1;     // в B ссылка на предыдущий A
        str.prev := ptr1;             // в начало дек ссылку на A
    end;
    str.data := str.data + 1;         // счётчик +1
end;

6: begin
    number := InputNumber();           // ввод
    new(ptr1);                      // выделение памяти
    (ptr1^).data := number;          // запись значения в A
    (ptr1^).prev := nil;             // в A ссылка на следующий
    (ptr1^).next := nil;             // в A ссылка на предыдущий
    if str.prev = nil then begin
        str.prev := ptr1;            // если дек пуст
        str.next := ptr1;            // в дек ссылку на начало
    end else begin
        (ptr1^).prev := str.next;     // в A ссылка на предыдущий B
        (str.next^).next := ptr1;     // в B ссылка на следующий A
        str.next := ptr1;             // в конец дек ссылку на A
    end;
    str.data := str.data + 1;         // счётчик +1
end;

```

```

// 7: begin
//   if Length(arr) > 0 then begin
//     number := PopStart(arr);
//     writeln('Число с начала: ', number);
//   end else writeln('В массиве нет элементов');
// end;

// 8: begin
//   if Length(arr) > 0 then begin
//     number := PopEnd(arr);
//     writeln('Число с конца: ', number);
//   end else writeln('В массиве нет элементов');
// end;
end;

readln;
end;

if selected < 0 then selected := MAX_MSG;
selected := selected mod (MAX_MSG + 1);
// Sleep(50);

until false;
DoneKeyboard;
end.

```

## **Выводы**

В ходе работы бала реализованна структура данных "Дек". Дек можно использовать как Стек или как Очередь. Память выделяется динамически при добавлении или удалении элемента Дек'а. Также были повторены функции, процедуры, case-меню и была реализованна навигация в меню при помощи стрелочек.