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:

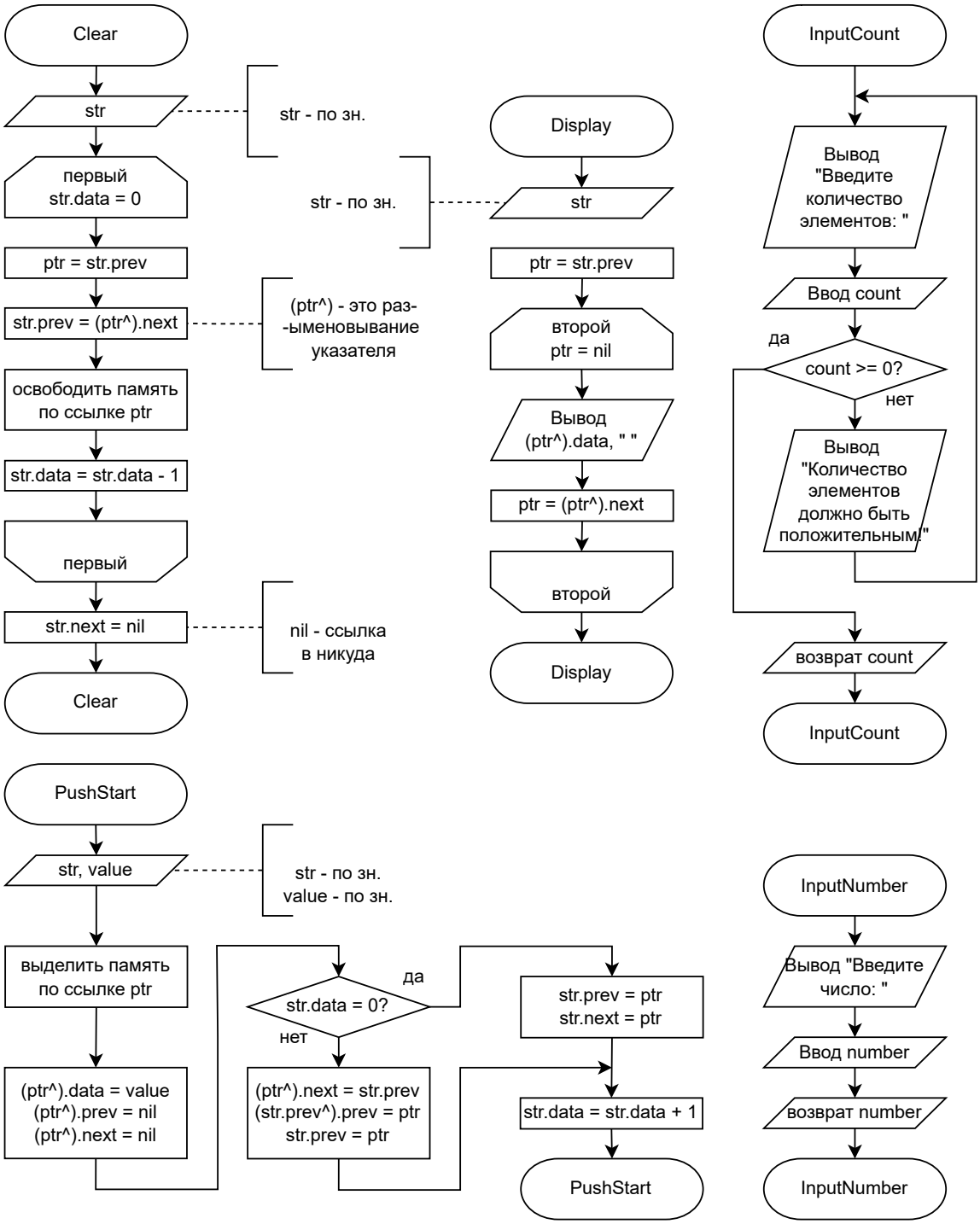

Рис. 1: функции и процедуры ч.1

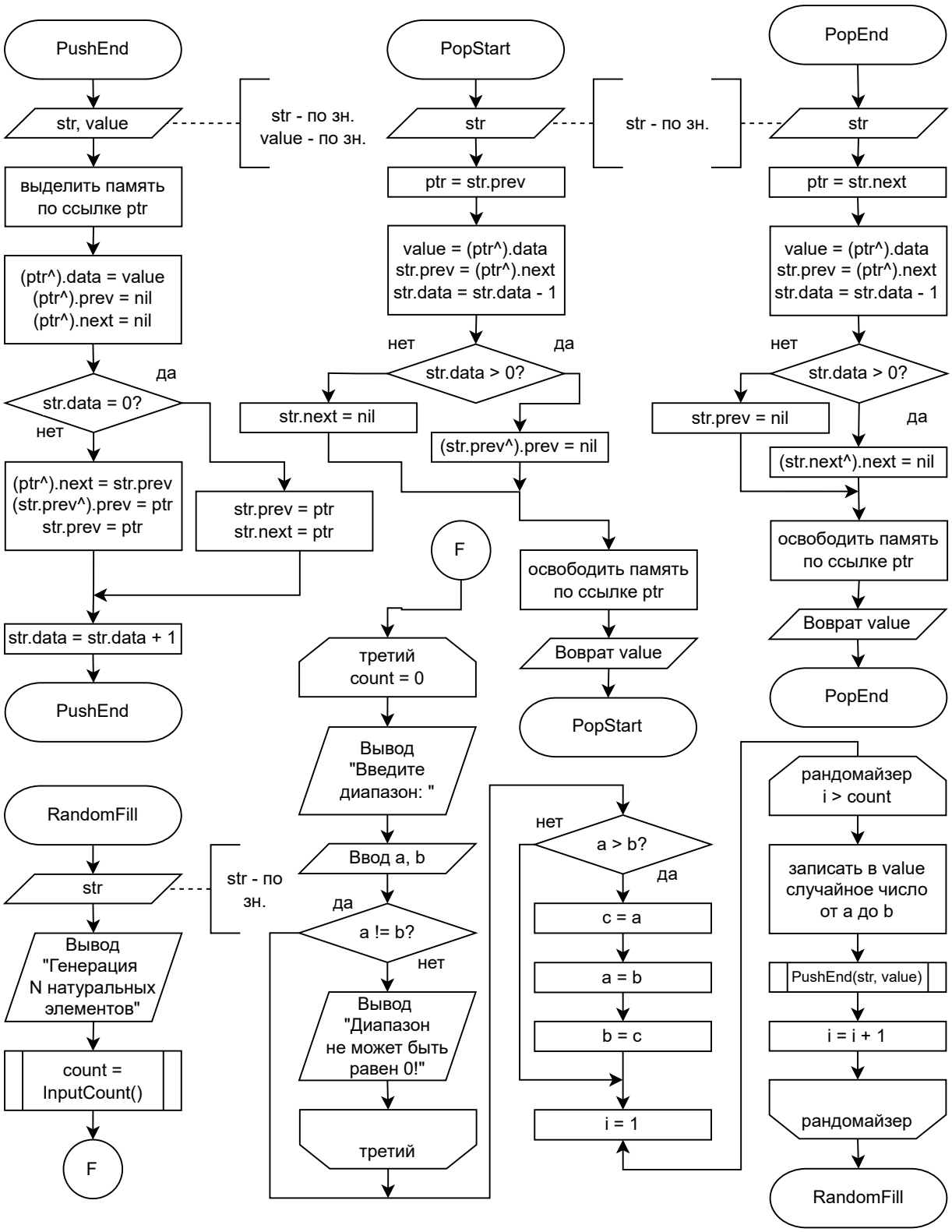

Рис. 2: функции и процедуры ч.2

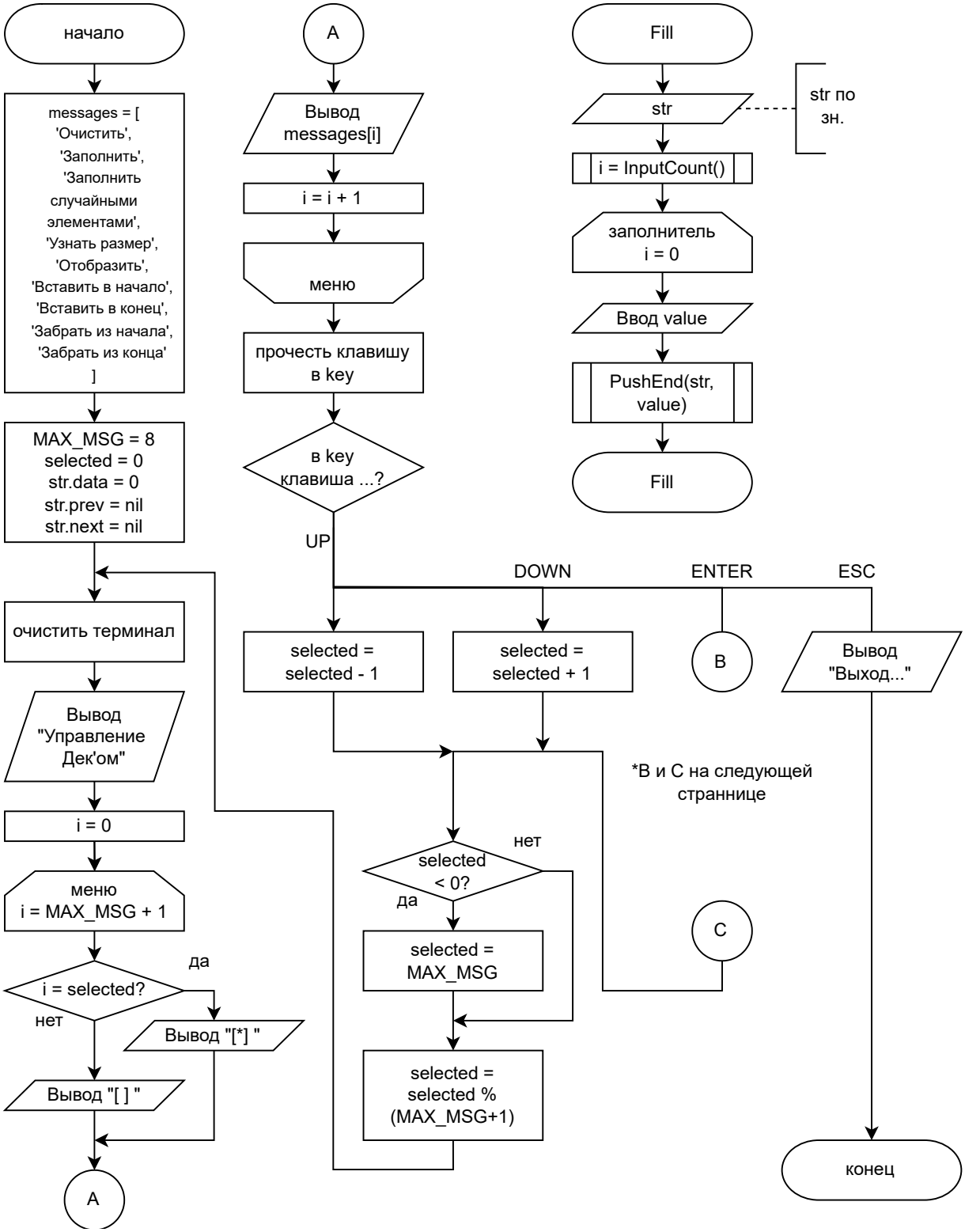

Рис. 3: основное тело программы ч.1

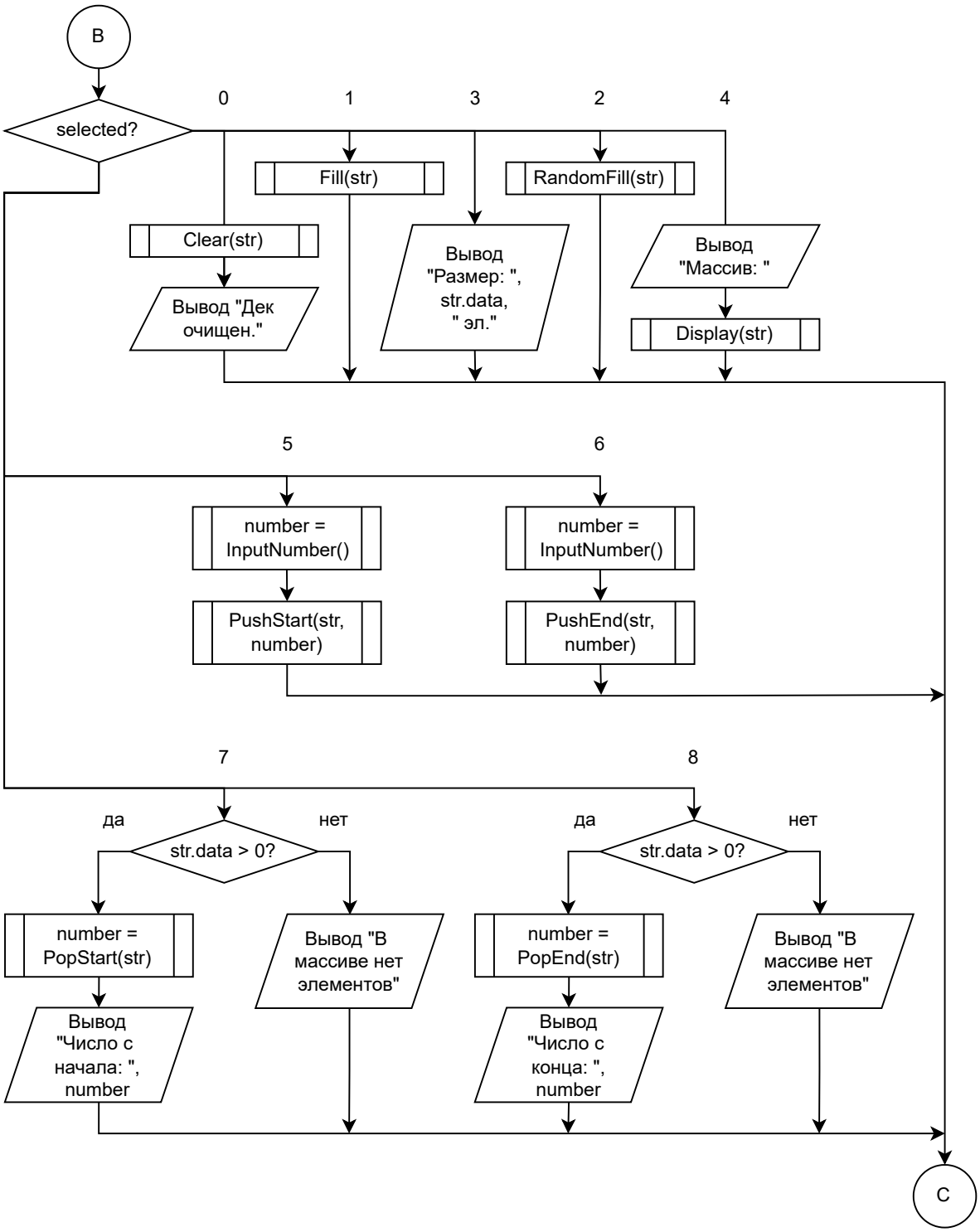

Рис. 4: основное тело программы ч.2

Код программы на Pascal:

```
program Main;
uses Keyboard, SysUtils;

const
  MAX_MSG = 8;

type
        mainType = longint;
  Element = record
    data: mainType;
    prev: ^Element;
    next: ^Element;
  end;



// ########################################################
// ФУНКЦИИ И ПРОЦЕДУРЫ

function InputCount(): mainType;
var
  count: mainType;
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(): mainType;
var
  number: mainType;
begin
```

```
  write('Введите число: ');
  readln(number);
  InputNumber := number;
end;

procedure Clear(var str: Element);
var
  ptr: ^Element;
begin
  while str.data > 0 do begin    // пока количество больше 0
    ptr := str.prev;             // ссылка на элемент
    str.prev := (ptr^).next;     // ссылка на следующий
    dispose(ptr);                // освобождаем память
    str.data := str.data - 1;
  end;
  str.next := nil;
end;

procedure Display(var str: Element);
var
  ptr: ^Element;
begin
  ptr := str.prev;               // указатель на текущий
  while ptr <> nil do begin      // пока указатель действительный
    write((ptr^).data, ' ');     // вывести значение элемента
    ptr := (ptr^).next;          // указатель на следующий
  end;
end;

procedure PushStart(var str: Element; value: mainType);
var
  ptr: ^Element;
begin
  new(ptr);                      // выделение памяти
  (ptr^).data := value;          // запись значения в A
  (ptr^).prev := nil;            // в A ссылка на следующий
  (ptr^).next := nil;            // в A ссылка на предыдущий
```

```
  if str.data = 0 then begin      // если дек пуст
    str.prev := ptr;               // в дек ссылку на начало
    str.next := ptr;               // в дек ссылку на конец
  end else begin                   // иначе
    (ptr^).next := str.prev;       // в А ссылка на следующий В
    (str.prev^).prev := ptr;       // в В ссылка на предыдущий А
    str.prev := ptr;               // в начало дек ссылку на А
  end;
  str.data := str.data + 1;        // счётчик +1
end;

procedure PushEnd(var str: Element; value: mainType);
var
  ptr: ^Element;
begin
  new(ptr);                        // выделение памяти
  (ptr^).data := value;           // запись значения в А
  (ptr^).prev := nil;              // в А ссылка на следующий
  (ptr^).next := nil;              // в А ссылка на предыдущий
  if str.data = 0 then begin      // если дек пуст
    str.prev := ptr;               // в дек ссылку на начало
    str.next := ptr;               // в дек ссылку на конец
  end else begin                   // иначе
    (ptr^).prev := str.next;       // в А ссылка на предыдущий Б
    (str.next^).next := ptr;       // в Б ссылка на следующий А
    str.next := ptr;               // в конец дек ссылку на А
  end;
  str.data := str.data + 1;        // счётчик +1
end;

function PopStart(var str: Element): mainType;
var
  value: mainType;
  ptr: ^Element;
begin
  ptr := str.prev;
  value := (ptr^).data;
```

```
  str.prev := (ptr^).next;
  str.data := str.data - 1;
  if str.data > 0 then
    (str.prev^).prev := nil
        else
                str.next := nil;
  dispose(ptr);
  PopStart := value;
end;

function PopEnd(var str: Element): mainType;
var
  value: mainType;
  ptr: ^Element;
begin
  ptr := str.next;
  value := (ptr^).data;
  str.next := (ptr^).prev;
  str.data := str.data - 1;
  if str.data > 0 then
    (str.next^).next := nil
        else
                str.prev := nil;
  dispose(ptr);
  PopEnd := value;
end;

procedure Fill(var str: Element);
var
  value, i: mainType;
begin
  for i := InputCount() downto 1 do begin
    read(value); PushEnd(str, value);
  end;
end;

procedure RandomFill(var str: Element);
```

```
var
  a, b, c, count, i, value: mainType;
begin
  writeln('Генерация N натуральных элементов');
  count := InputCount();

  while count <> 0 do begin
    write('Введите диапазон: ');
    readln(a, b);
    if a <> b then break;
    writeln('Диапазон не может быть равен 0!');
  end;

  if (a > b) and (count <> 0) then begin
    c := a; a := b; b := c;
  end;

  for i := 1 to count do begin
    // записать в value случайное число от a до b
    value := Random(b - a + 1) + a;
    PushEnd(str, value);
  end;
end;
// ########################################################

var
  _key_event: TKeyEvent;

  str: 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, i: mainType;
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 := GetKeyEventCode(_key_event);
```

```
// ################################

// в 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
        Clear(str);
        writeln('Дек очищен.');
      end;

      1: begin
        Fill(str);
      end;

      2: begin
        RandomFill(str);
      end;

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

      4: begin
        write('Массив: '); Display(str);
      end;
```

```
          5: begin
            number := InputNumber();
            PushStart(str, number);
          end;

          6: begin
            number := InputNumber();
            PushEnd(str, number);
          end;

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

          8: begin
            if str.data > 0 then begin
              number := PopEnd(str);
              writeln('Число с конца: ', number);
            end else
              writeln('В массиве нет элементов');
          end;
        end;
        readln;
      end;
    end;

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

  until false;
  DoneKeyboard;
end.
```

Выводы

В ходе работы была реализованна структура данных "Дек" на двусвязном списке. В данном случае двусвязный список используется как для прямого хода, так и для обратного хода по элементам, но элементы добавляются/удаляются только в концах списка. При добавлении элемента выделяется память для его значения и двух указателей: на предыдущий и следующий элемент. При удалении элемента память освобождается, а указатели присваиваются на действительный элемент. При очистке все элементы удаляются по цепочке, пустой список не занимает динамическую память. Также были повторены функции, процедуры, указатели, case-меню и была реализованна навигация в меню при помощи стрелочек.