КАЛУЖСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО
ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ИМЕНИ Н.Э. БАУМАНА
(национальный исследовательский университет)»

Факультет «Информатика и управление»

Кафедра «Программное обеспечение ЭВМ, информационные технологии»

Типы и структуры данных

Лекция №1-2.
«Данные динамической структуры»

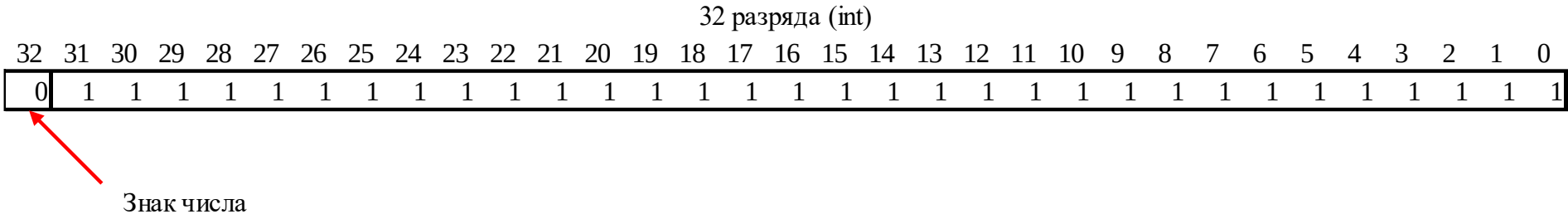
Калуга - 2022

Вопросы на лекции

- Представление чисел в памяти компьютера
- Связные списки (однонаправленные, двунаправленные)
- Основные операции над списками
- Стек
- Очередь
- Дек

Представление целых чисел со знаком в памяти компьютера

- Битовая структура числа:



Представление вещественных в памяти компьютера

- Вещественные числа в памяти компьютера представляются в форме с плавающей точкой.
- Форма с плавающей точкой использует представление **вещественного числа X в виде произведения мантиссы m на основание системы счисления E в некоторой целой степени p**, которую называют порядком:
  - $X = mE^p$

m – мантисса,
E – экспонента,
p – целый порядок со знаком.

$12{,}34 = 0{,}1234 * 10^2$

Представление вещественных в памяти компьютера

Общие сведения

Динамические структуры данных – это структуры данных, память под которые выделяется и освобождается по мере необходимости.

Динамическая структура данных характеризуется тем что:

- она не имеет имени;
- ей выделяется память в процессе выполнения программы;
- количество элементов структуры может не фиксироваться;
- размерность структуры может меняться в процессе выполнения программы;
- в процессе выполнения программы может меняться характер взаимосвязи между элементами структуры.

Общие сведения

Необходимость в динамических структурах данных обычно возникает в следующих случаях.

- Используются переменные, имеющие довольно большой размер (например, массивы большой размерности), необходимые в одних частях программы и совершенно не нужные в других.
- В процессе работы программы нужен массив, список или иная структура, размер которой изменяется в широких пределах и трудно предсказуем.
- Когда размер данных, обрабатываемых в программе, превышает объем сегмента данных.

Общие сведения

Достоинства связного представления данных – в возможности обеспечения значительной изменчивости структур:

- размер структуры ограничивается только доступным объемом машинной памяти;
- при изменении логической последовательности элементов структуры требуется не перемещение данных в памяти, а только коррекция указателей;
- большая гибкость структуры.

Общие сведения

Вместе с тем, связное представление не лишено и недостатков, основными из которых являются следующие:

- на поля, содержащие указатели для связывания элементов друг с другом, расходуется дополнительная память;
- доступ к элементам *связной структуры* может быть менее эффективным по времени.

Общие сведения

Порядок работы с *динамическими структурами данных* следующий:

- создать (отвести место в динамической памяти);
- работать при помощи указателя;
- удалить (освободить занятое структурой место).

Классификация динамических структур данных

- *однонаправленные (односвязные) списки;*
- *двунаправленные (двусвязные) списки;*
- *циклические списки;*
- *стек;*
- *дек;*
- *очередь;*
- *бинарные деревья.*

Объявление динамических структур данных

```
struct TNode {
     int Data;
         //информационное поле
     TNode *Next;
         //адресное поле
};
```

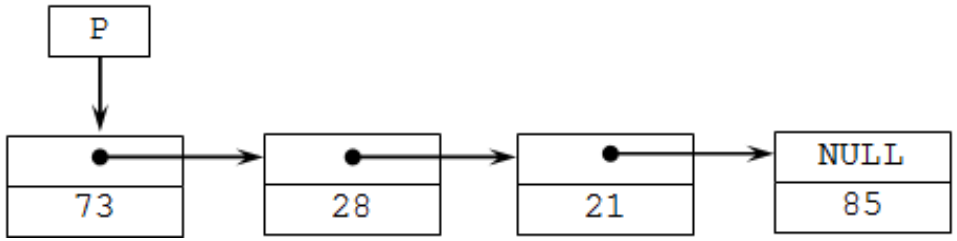


Рис. 1. Схематичное представление динамической структуры

Доступ к данным в динамических структурах

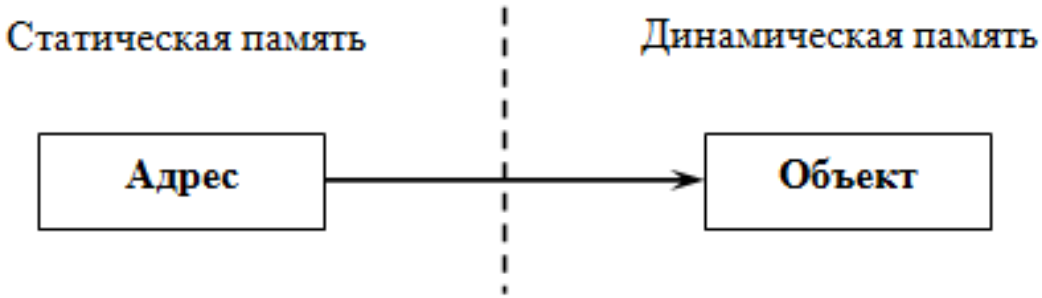


Рис. 2. Связь указателя с адресуемым объектом

```
УказательНаСтруктуру -> ИмяЭлемента
```

Работа с памятью при использовании динамических структур данных

```
struct Node {char *Name;
             int Value;
             Node *Next
            };
Node *PNode; //объявляется указатель

PNode = new Node; //выделяется память

PNode->Name = "STO"; //присваиваются значения
PNode->Value = 28;
PNode->Next = NULL;

delete PNode; // освобождение памяти
```

Однонаправленные (односвязные) списки

- **Списком** называется упорядоченное множество, состоящее из переменного числа элементов, к которым применимы *операции включения*, *исключения*. Список, отражающий отношения соседства между элементами, называется *линейным*.
- **Однонаправленный (односвязный) список** – это структура данных, представляющая собой последовательность элементов, в каждом из которых хранится значение и указатель на следующий элемент списка. В последнем элементе указатель на следующий элемент равен NULL.

Однонаправленные (односвязные) списки

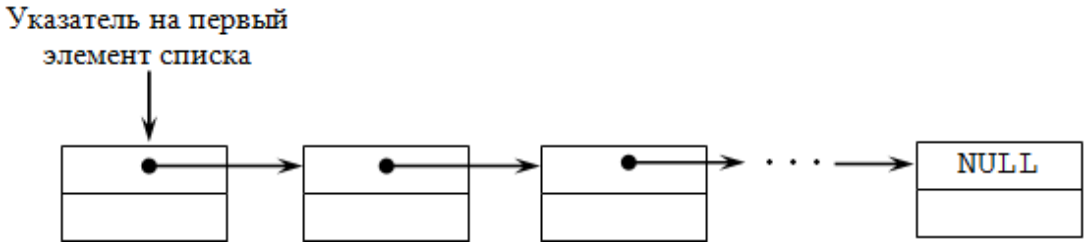


Рис. 3. Линейный однонаправленный список

Основными операциями, осуществляемыми с однонаправленными списками, являются:

- *создание списка;*
- *печать (просмотр) списка;*
- *вставка элемента в список;*
- *удаление элемента из списка;*
- *поиск элемента в списке*
- *проверка пустоты списка;*
- *удаление списка.*

Вставка элемента в однонаправленный список

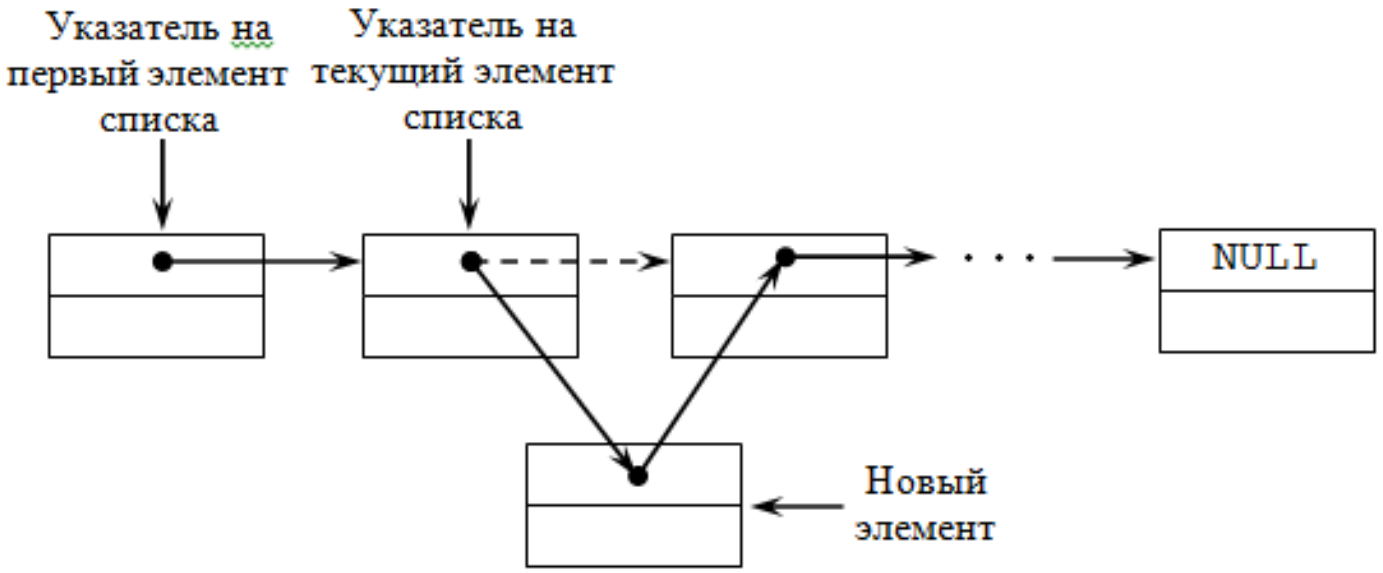


Рис. 4. Вставка элемента в однонаправленный список

Удаление элемента из однонаправленного списка

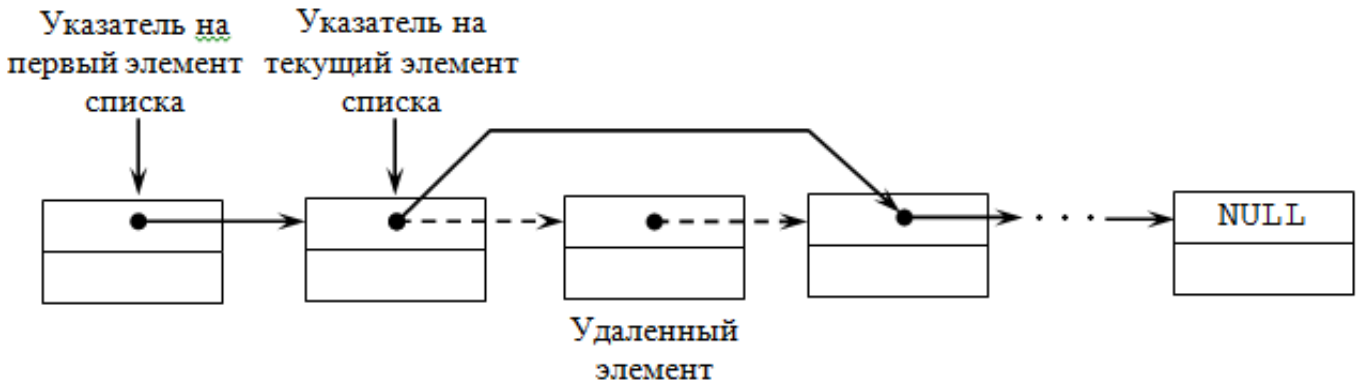


Рис. 5. Удаление элемента из однонаправленного списка

Двунаправленные (двусвязные) списки

- **Двунаправленный (двусвязный) список** – это структура данных, состоящая из последовательности элементов, каждый из которых содержит информационную часть и два указателя на соседние элементы. При этом два соседних элемента должны содержать взаимные ссылки друг на друга.

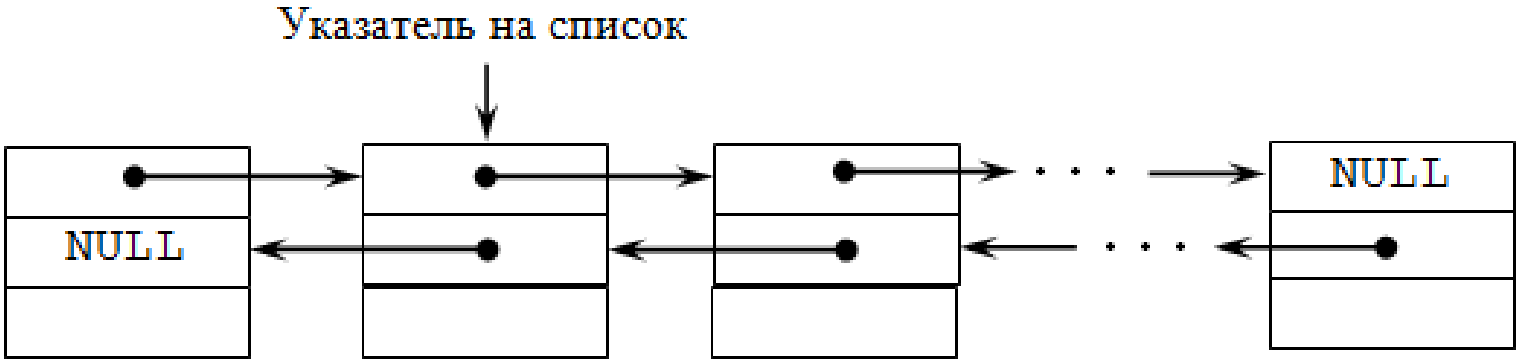


Рис. 6. Двунаправленный список

Двунаправленные (двусвязные) списки. Пример

```
struct Double_List {
             int Data;
        //информационное поле
            Double_List *Next,
         //адресное поле 1
                                  *Prior;
        //адресное поле 2
 };
. . . . . .
Double_List *Head;
//указатель на первый элемент списка
. . . . . .
Double_List *Current;
//указатель на текущий элемент списка (при необходимости)
```

Вставка элемента в двунаправленный список

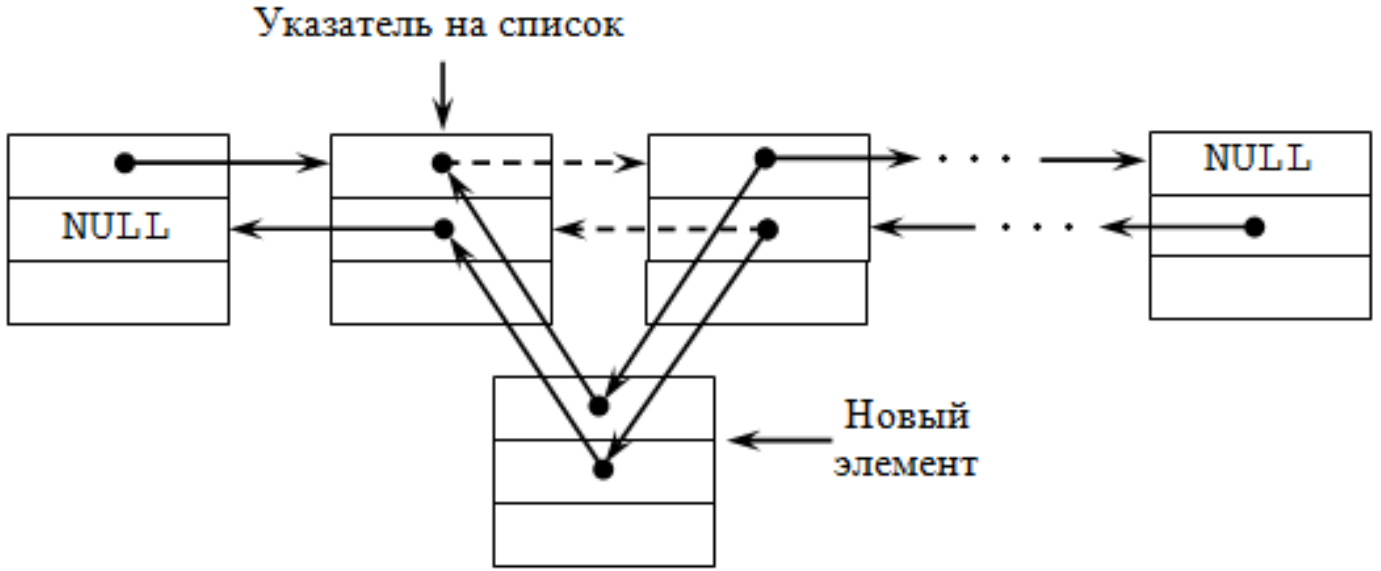


Рис. 7. Вставка элемента в двунаправленный список

Удаление элемента из двунаправленного списка

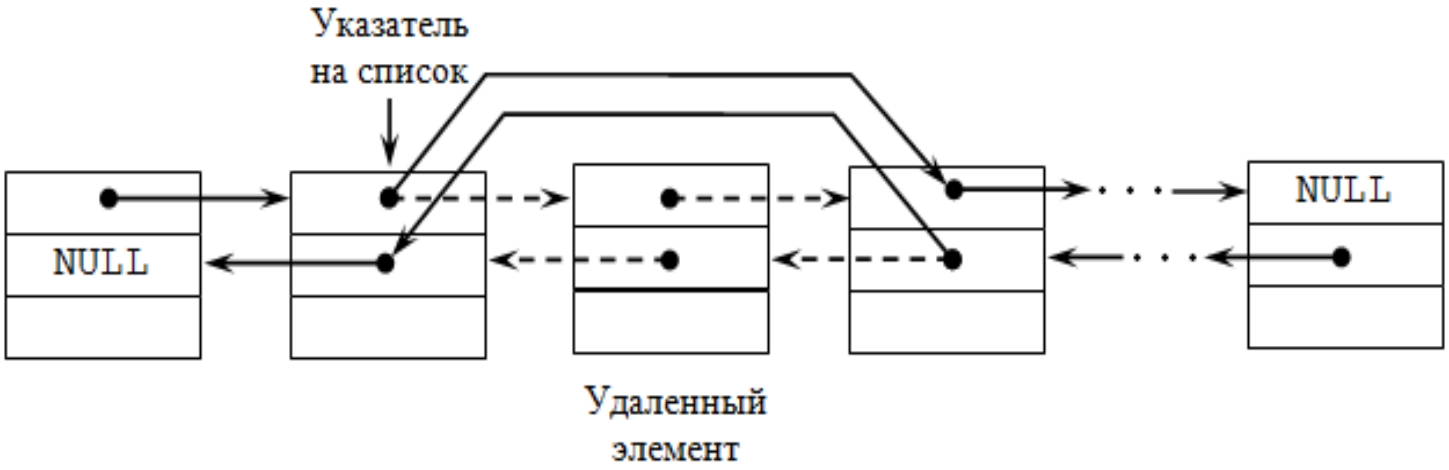


Рис. 8. Удаление элемента из двунаправленного списка

Стек

Стек (англ. *stack* – стопка) – это структура данных, в которой новый элемент всегда записывается в ее начало (вершину) и очередной читаемый элемент также всегда выбирается из ее начала. В стеках используется *метод доступа* к элементам *LIFO* (*Last Input – First Output*, "последним пришел – первым вышел").

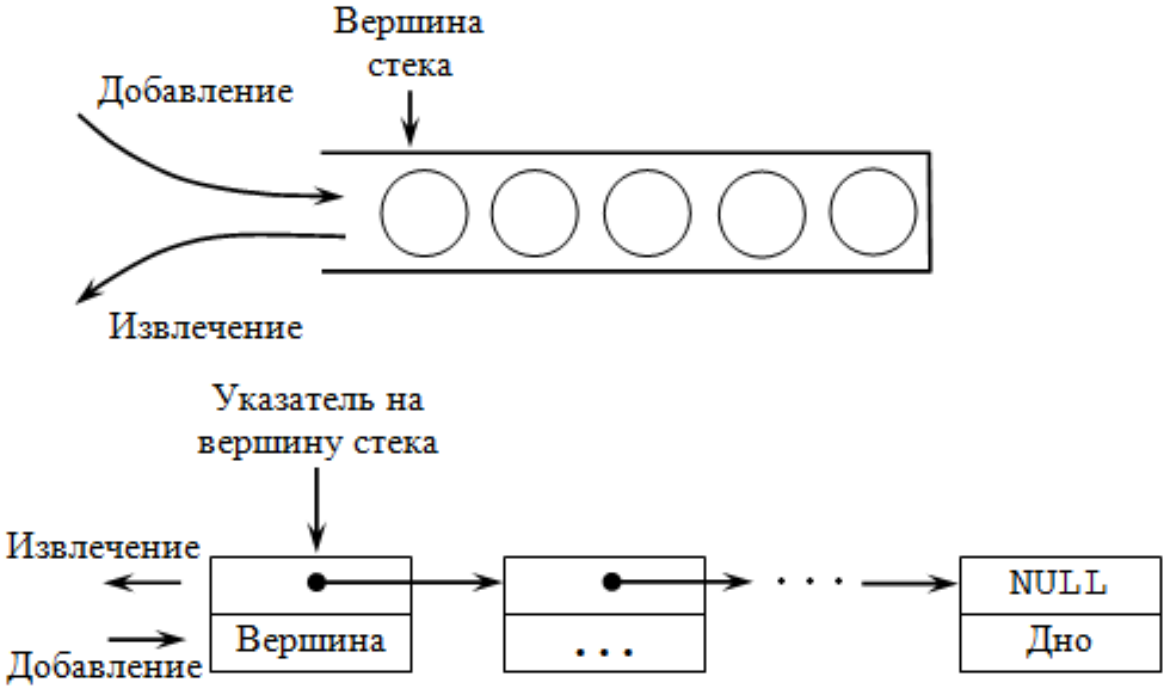


Рис. 9. Стек и его организация

Очередь

Очередь – это структура данных, представляющая собой последовательность элементов, образованная в порядке их поступления. Каждый новый элемент размещается в конце очереди; элемент, стоящий в начале очереди, выбирается из нее первым. В очереди используется принцип доступа к элементам *FIFO* (*First Input – First Output*, «первый пришёл – первый вышел»).

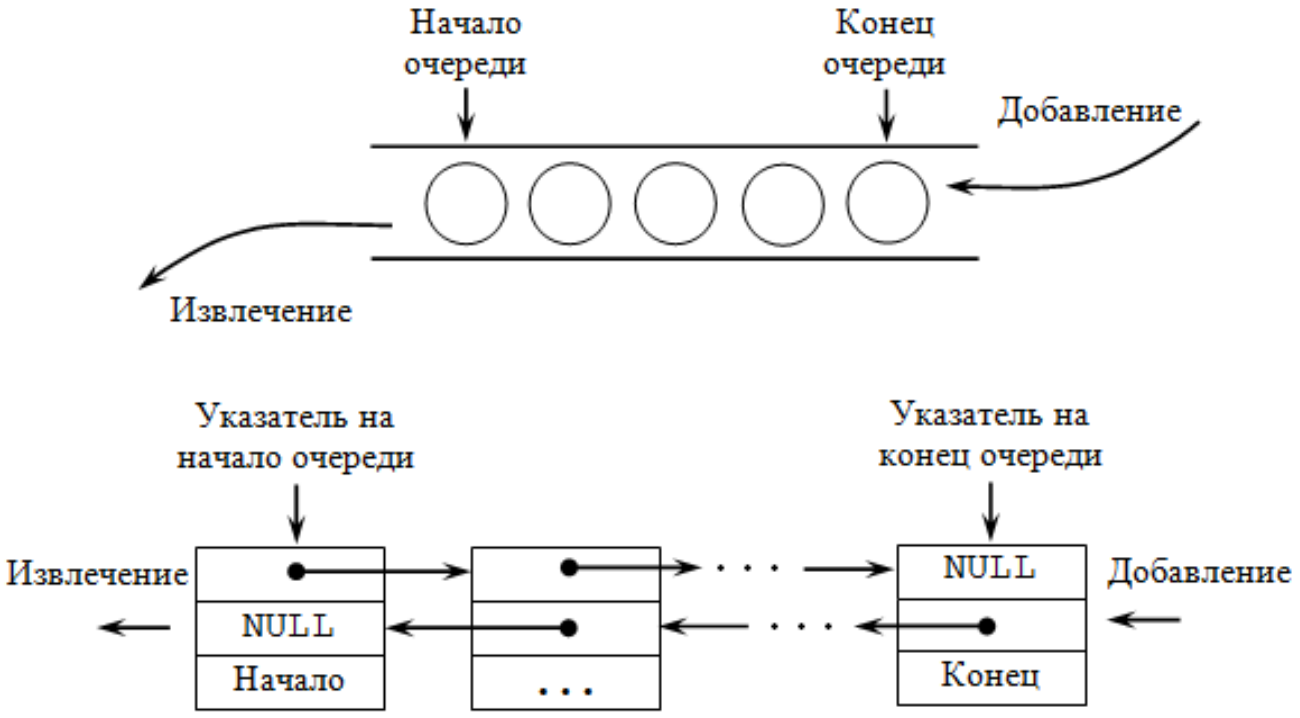


Рис. 10. Очередь и ее организация

Циклические списки

Циклический (кольцевой) список – это структура данных, представляющая собой последовательность элементов, последний элемент которой содержит указатель на первый элемент списка, а первый (в случае *двунаправленного* списка) – на последний.

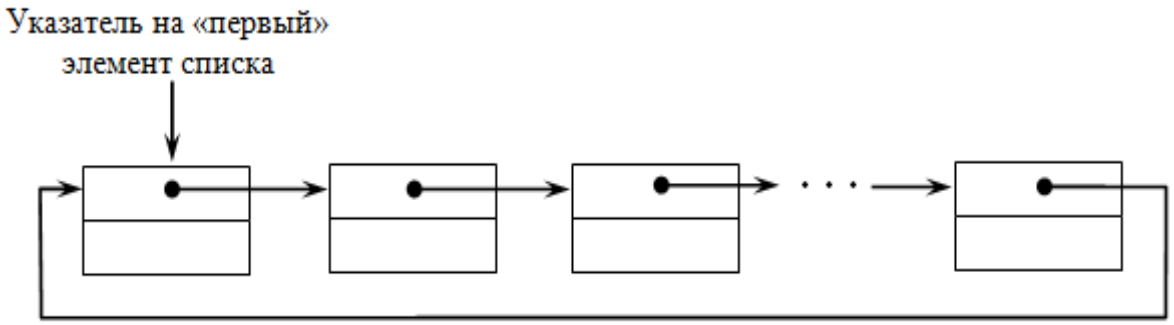


Рис. 11. Циклический однонаправленный список

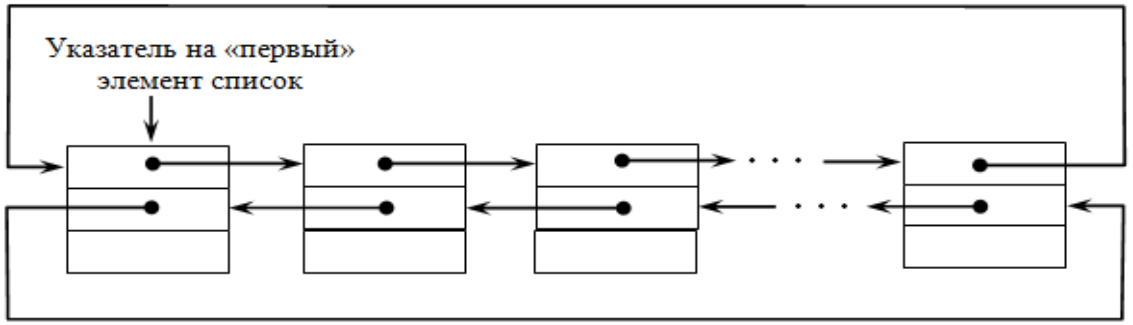


Рис. 12. Циклический двунаправленный список

Дек

Дек (англ. deque – аббревиатура от double-ended queue, двухсторонняя очередь) – это структура данных, представляющая собой последовательность элементов, в которой можно добавлять и удалять в произвольном порядке элементы с двух сторон (рис. 6). Первый и последний элементы дека соответствуют входу и выходу дека.

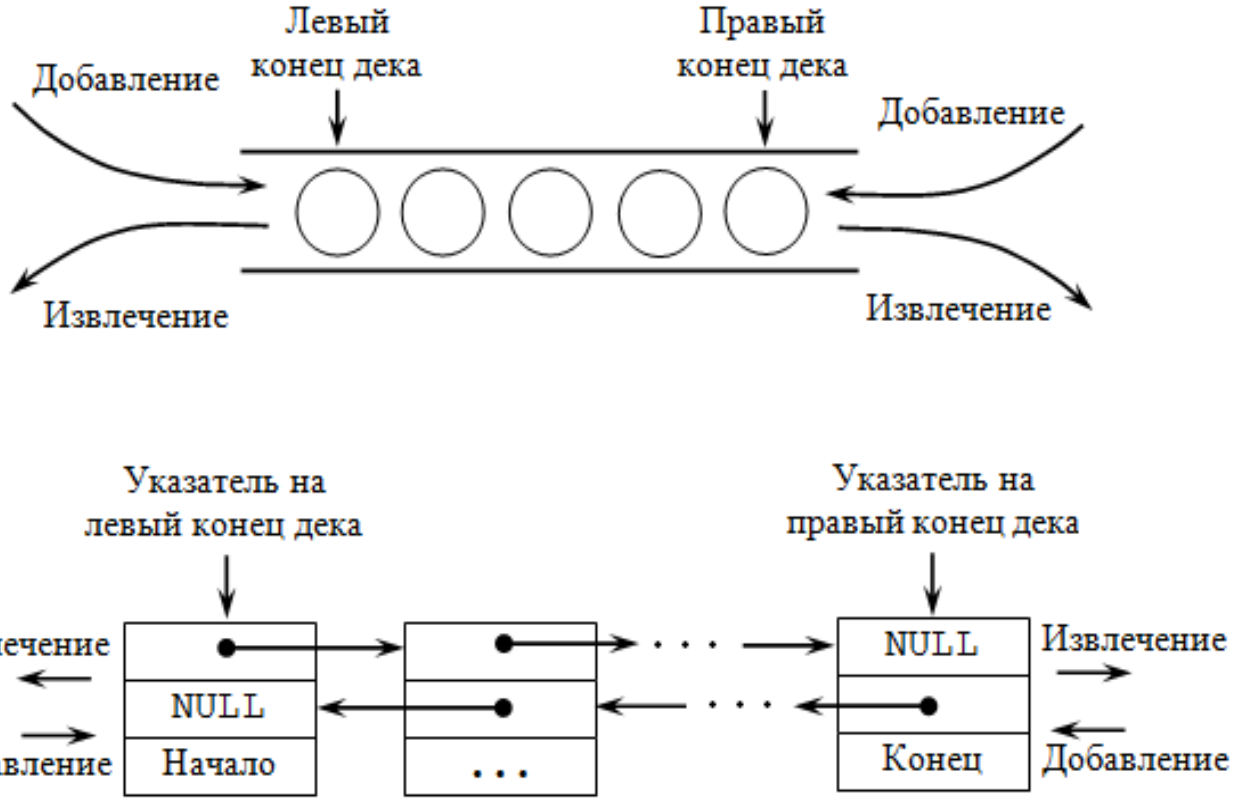


Рис. 13. Дек и его организация