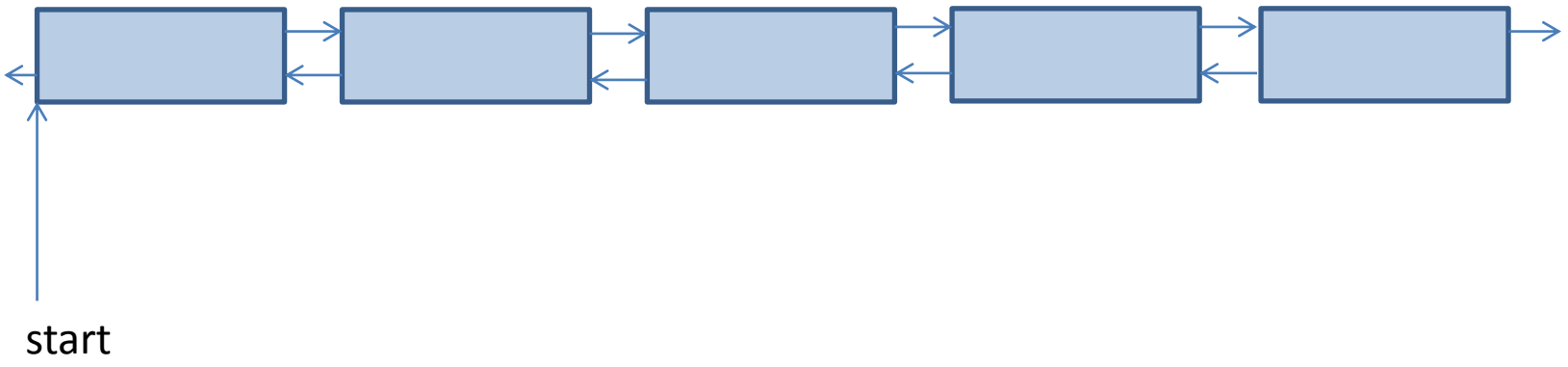
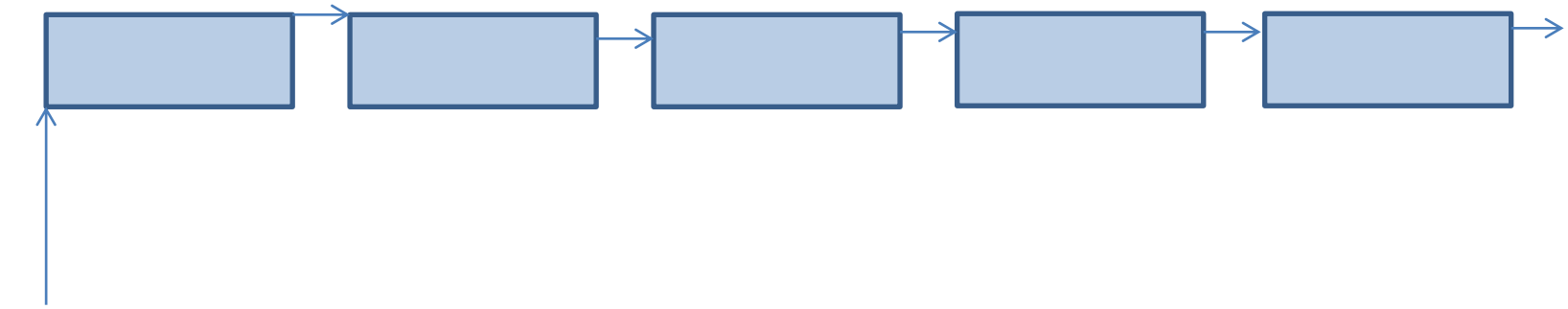
Двунаправленный (двусвязный) список – это структура данных, состоящая из последовательности элементов, каждый из которых содержит информационную часть и два указателя на соседние элементы (предыдущий элемент и следующий элемент).

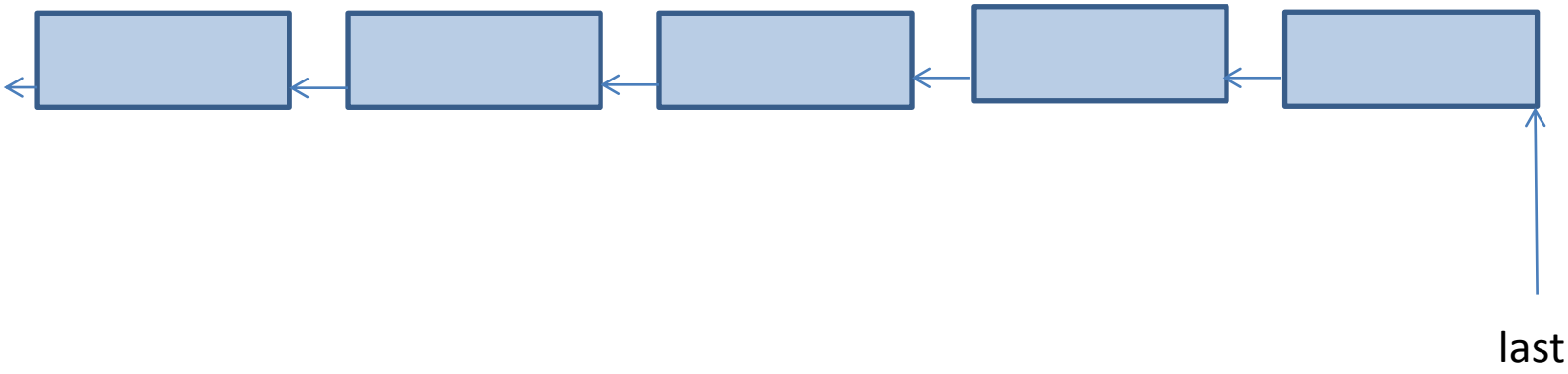


Однонаправленный (односвязный) список – это структура данных, представляющая собой последовательность элементов, в каждом из которых хранится значение и указатель на следующий (или предыдущий) элемент списка.



start

или



last

В языке С для представления элемента двунаправленного списка обычно используется следующая структура:

```
struct ListItem {
   struct ListItem *prev; /*указатель на предыдущий элемент*/
   struct ListItem *next; /*указатель на следующий элемент*/

   int info;                    /*информационная часть:
                                  здесь могут быть данные
                                  любого типа*/
};
```

Также создается переменная, представляющая собой начало списка:

```
struct ListItem *list = NULL;
```

Изначально наш список пуст, поэтому его начало – нулевой элемент.

Для добавления элементов в список нам необходимо последовательно присоединять их в его начало. Для этого необходимо выполнить следующие действия:

Создаем следующий элемент и записываем в него требуемую информационную часть:

```
struct ListItem* item = (struct ListItem*) malloc (sizeof(struct ListItem));
If (!item)
   return -1;   /* ошибка при выделении памяти */

item->info = info;
```

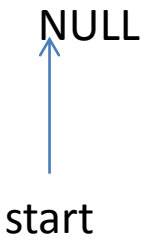
предыдущий к нему элемент пуст:
item->prev = NULL;

текущее начало списка теперь следующий элемент за вновь созданным:
item->next = start;

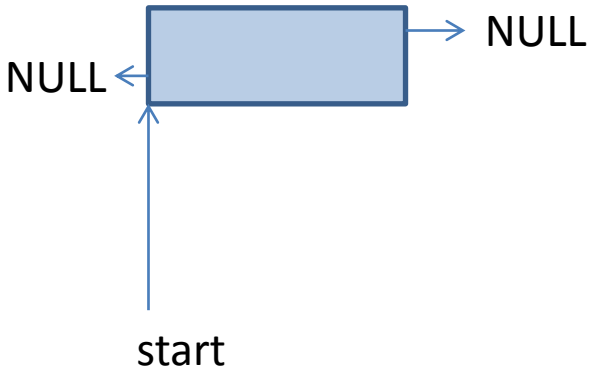
делаем новый элемент теперь предыдущим к текущему началу списка:
start->prev = item;

теперь вновь созданный элемент становится новым началом нашего списка:
start = item;

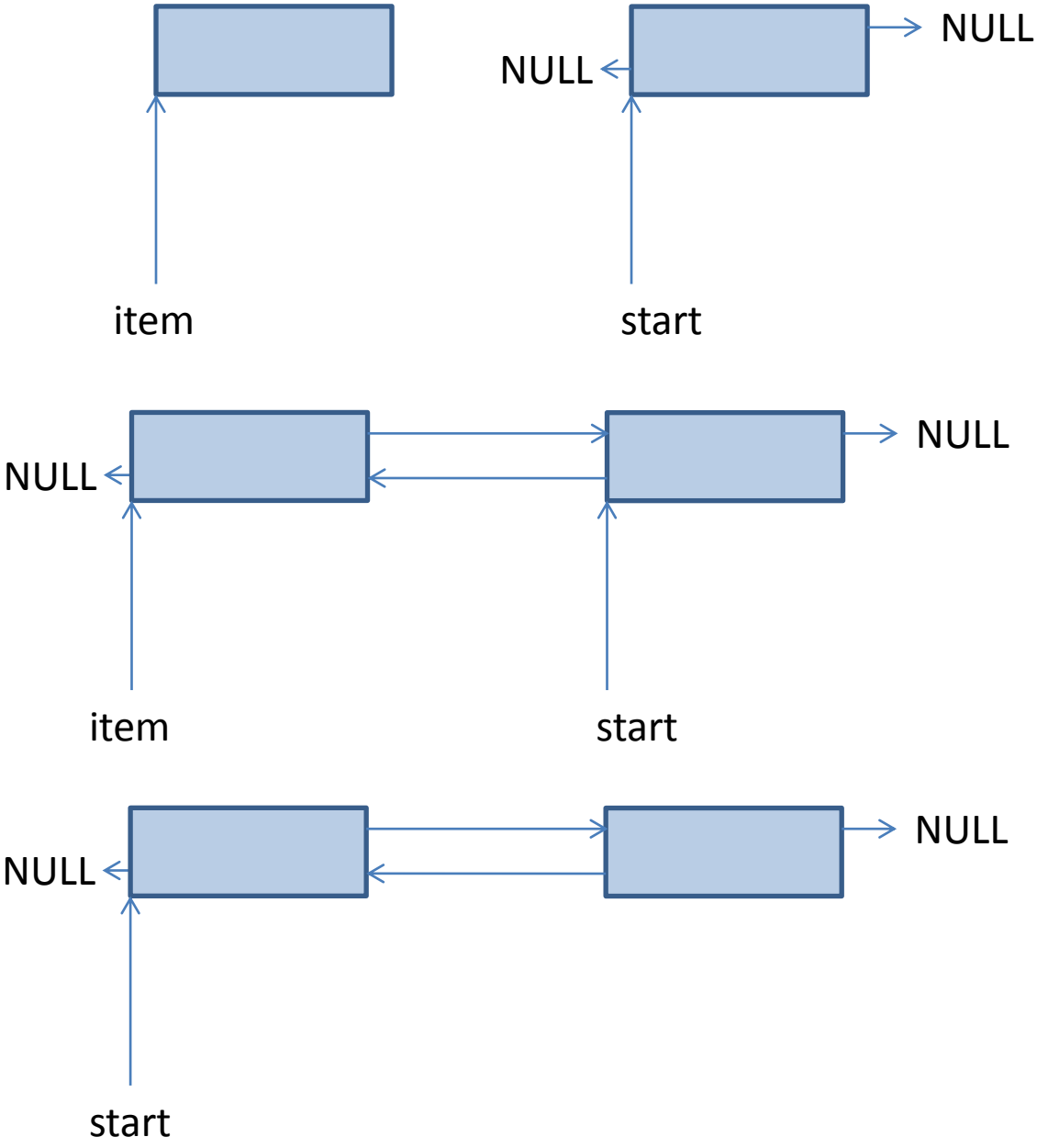
Изначально список пуст:

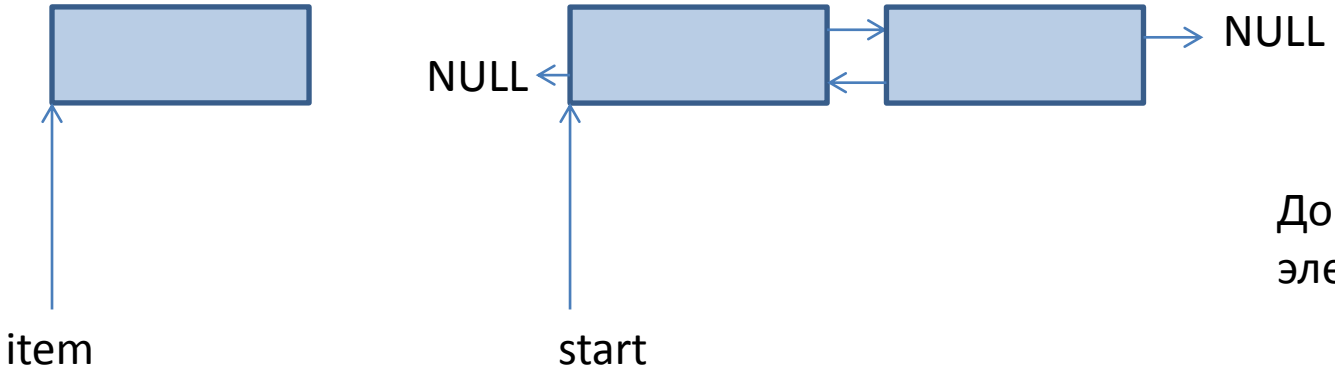


Создаем новый элемент, который становится началом нашего списка.
Предыдущий и следующий за ним элементы пока пусты:

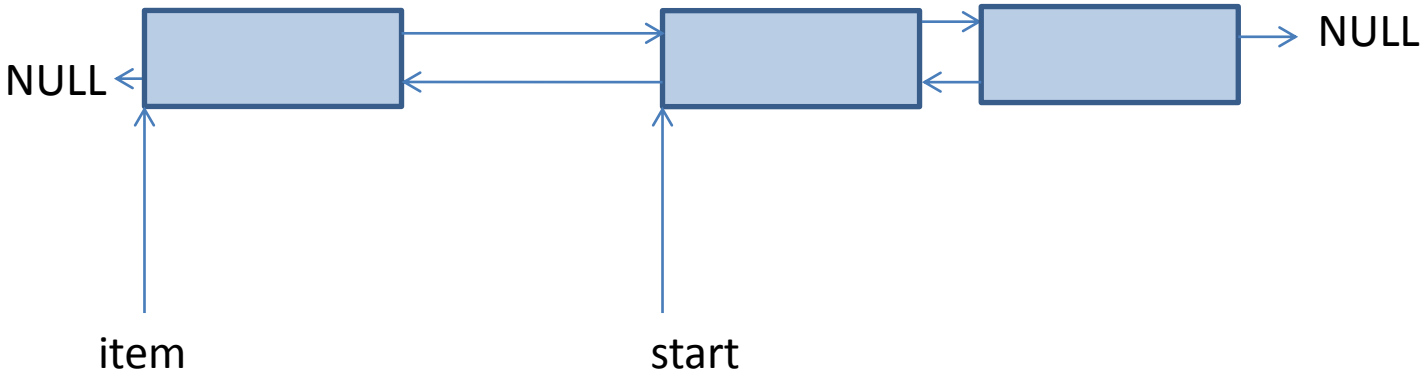


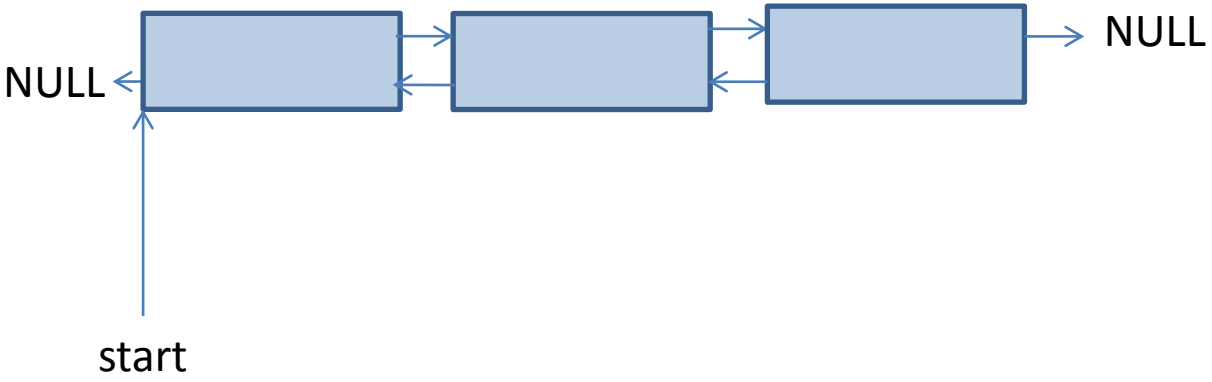
Добавление второго элемента к списку





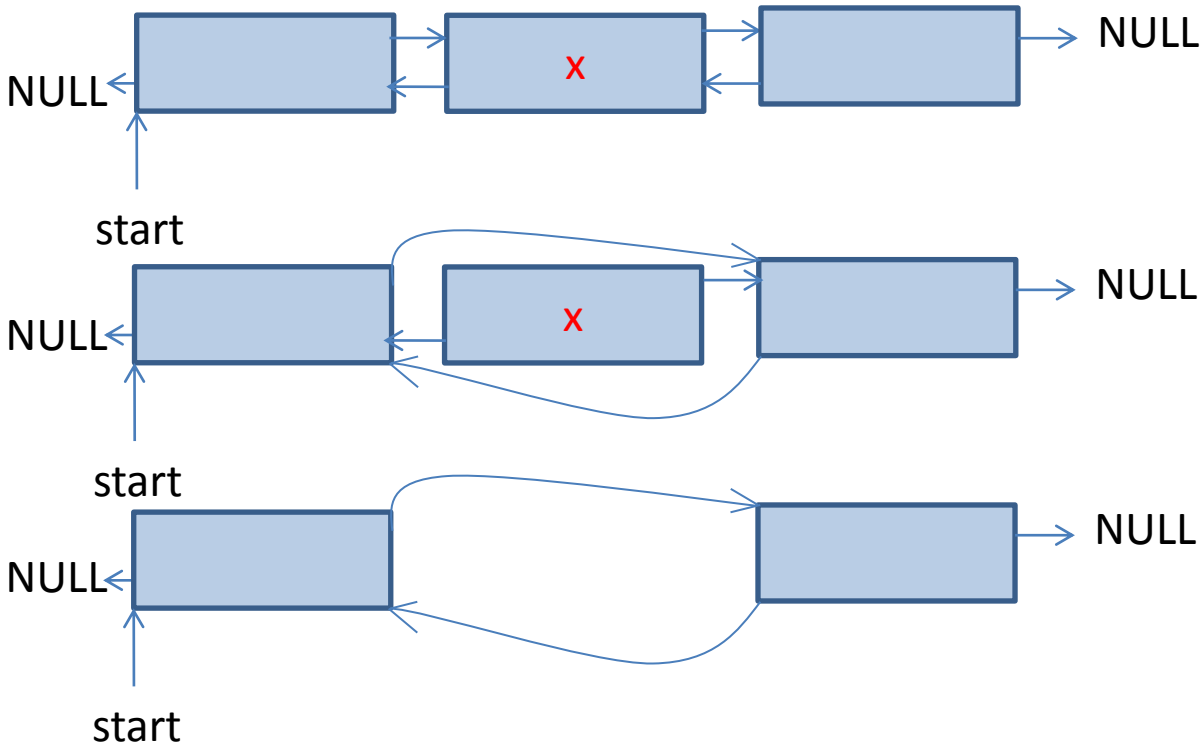
Добавление третьего элемента к списку





Для удаления элемента из середины списка необходимо:

1) предыдущим к следующему за удаляемым элементу сделать предыдущий к удаляемому элемент.
2) следующим за предыдущим к удаляемому элементу сделать следующий за удаляемым элемент.
3) удалить необходимый элемент.

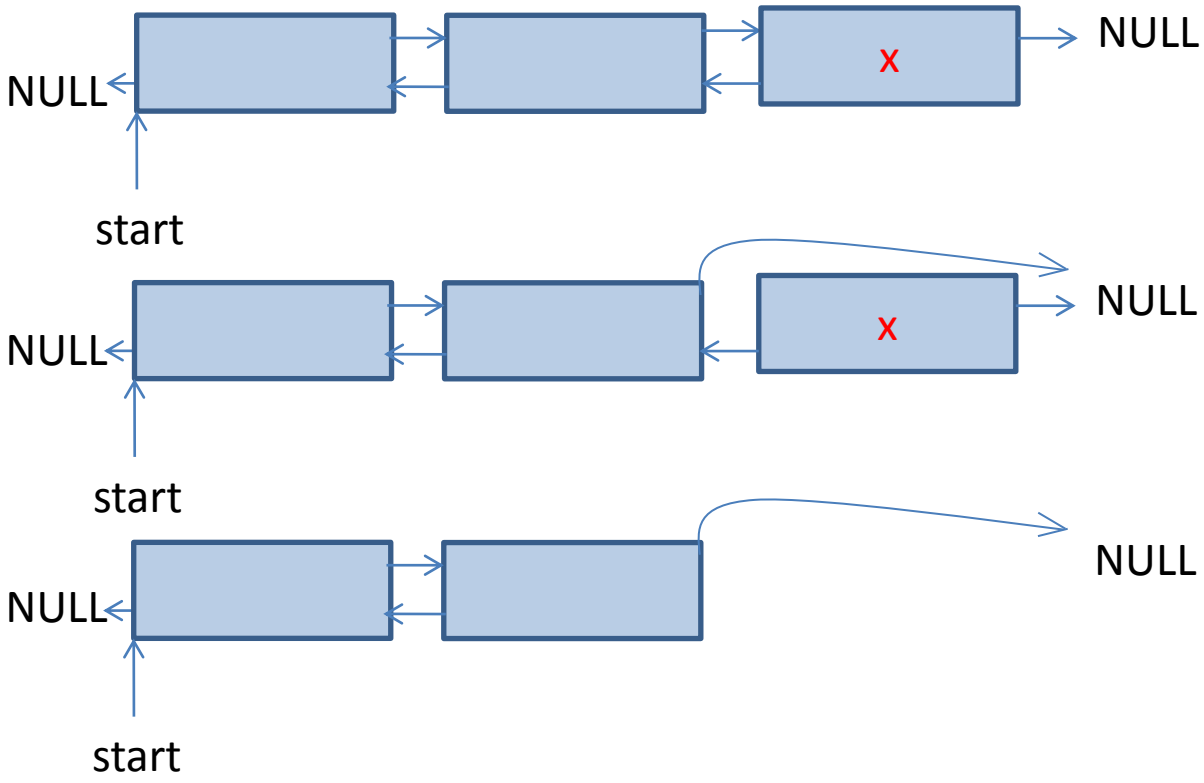


Возможны две специальные ситуации:

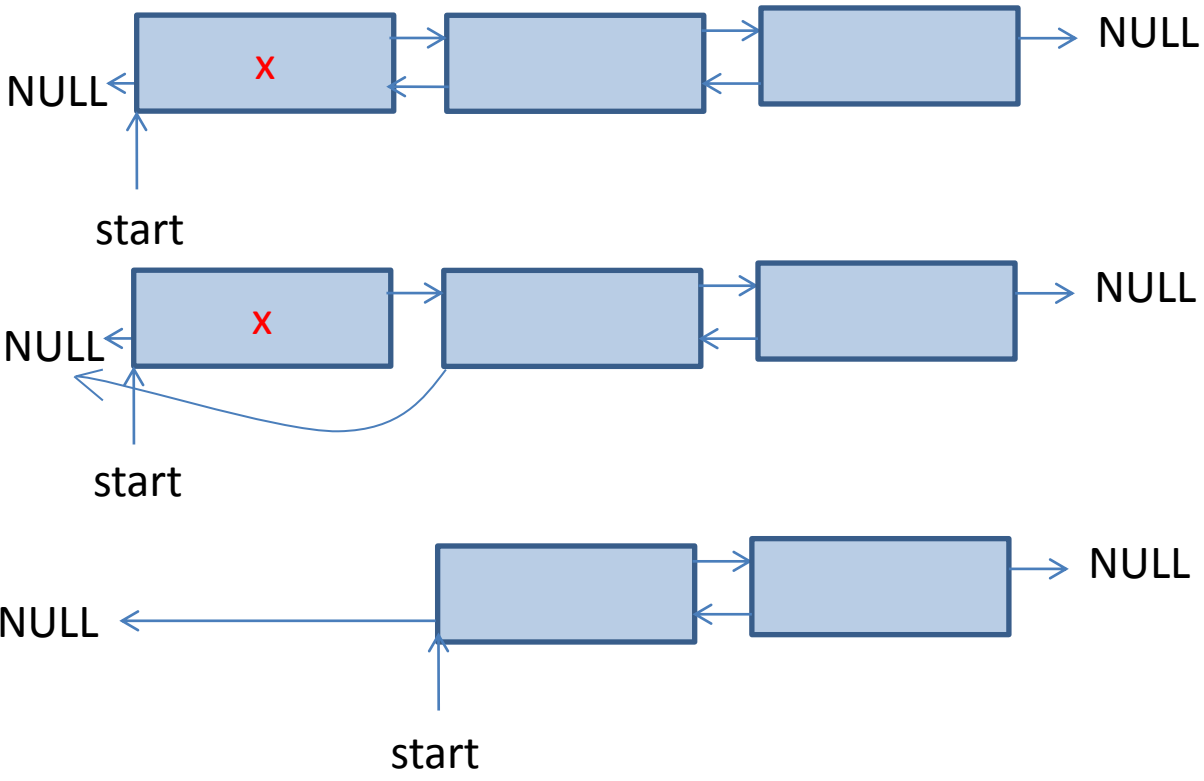
а) мы удаляем последний элемент списка:

в этом случае нам надо пропустить шаг (1)

ведь следующего за удаляемым элемента нету.



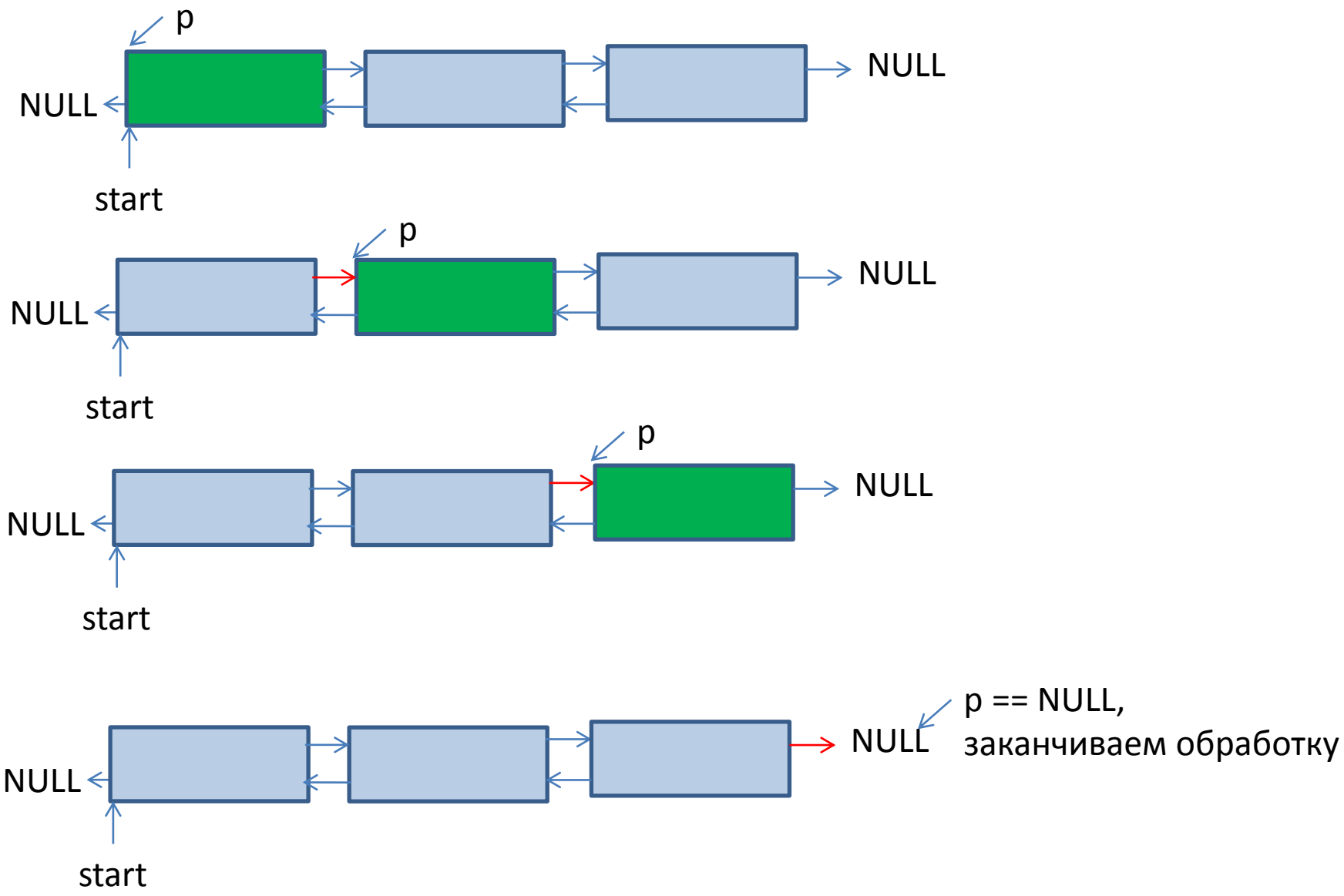
б) при удалении первого элемента списка вместо шага (2) нам необходимо передвинуть **start** на второй элемент. Если список состоял из единственного элемента, то **start** станет равно **NULL** и наш список будет считаться пустым.



Для обработки всех элементов списка (вывод на экран, поиск, очистка памяти) нам необходимо двигаться по нему, начиная с первого элемента (**start**) и переходя на **next** до тех пор, пока мы на дойдем до **NULL**:

```
struct ListItem *p = start;
while(p) {
   /*делаем что-то с элементом на который указывает p*/

  p = p->next;
}
```


p
NULL
NULL
start
p
NULL
NULL
start
p
NULL
NULL
start
NULL
p == NULL,
заканчиваем обработку
NULL
start