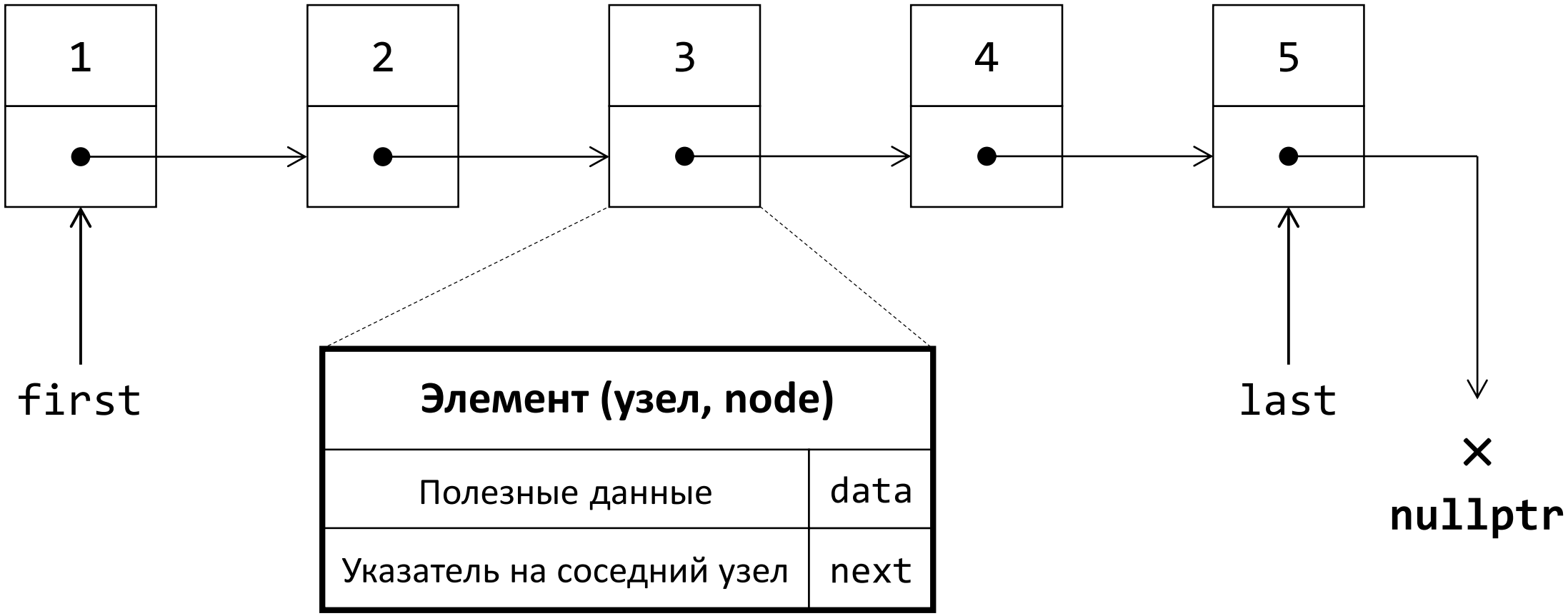
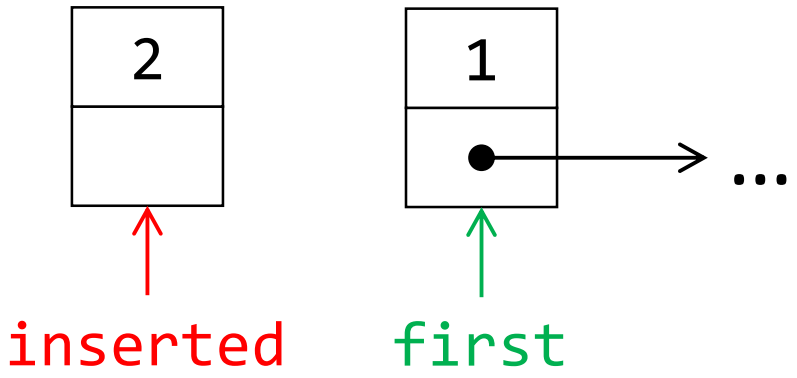
Односвязный список (singly-linked list)

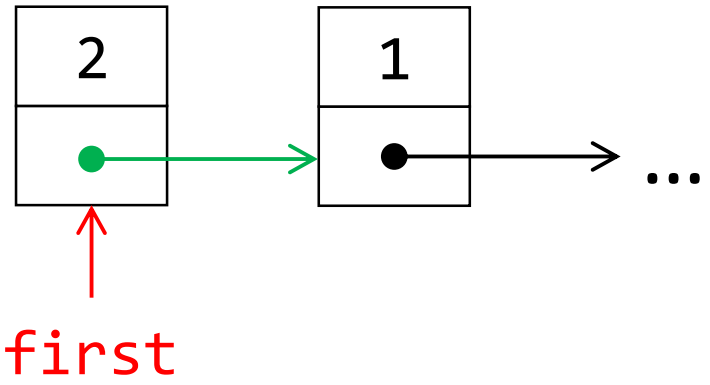


Вставка элемента: а) в начало списка

Было:
список и новый узел



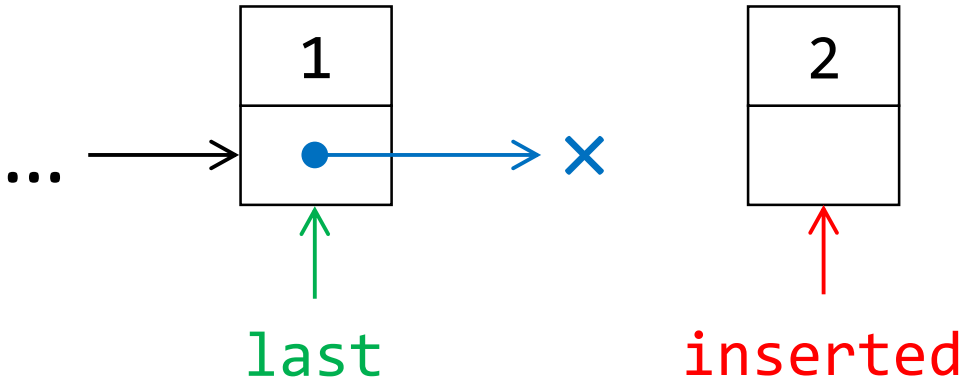
Стало:
список с новым узлом в начале



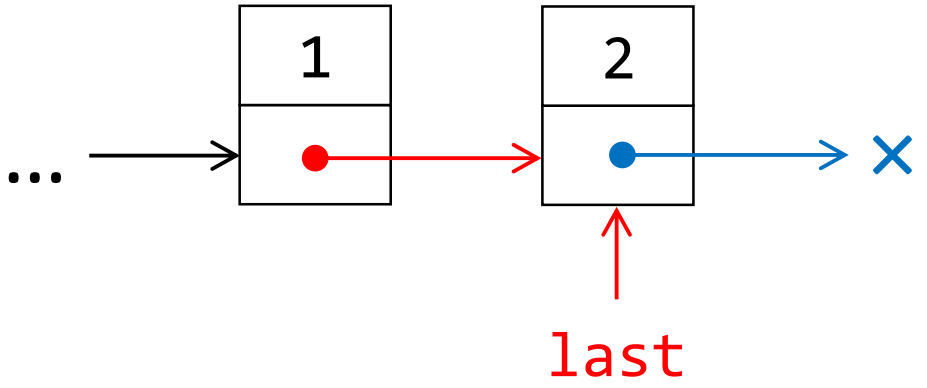
Один цвет — один адрес (значение указателя), кроме черных и серых (которые не важны).

Вставка элемента: б) в конец списка

Было:
список и новый узел

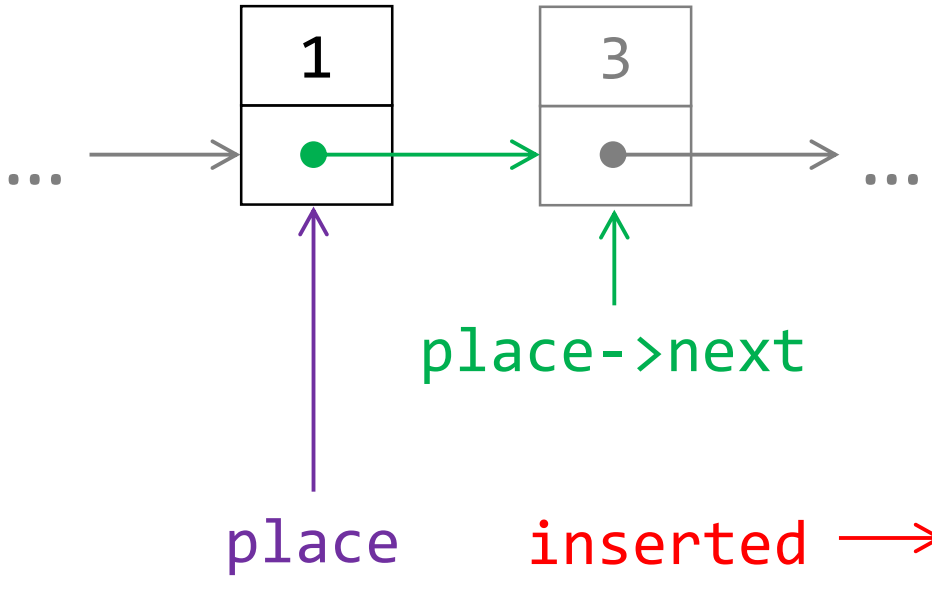


Стало:
список с новым узлом в конце

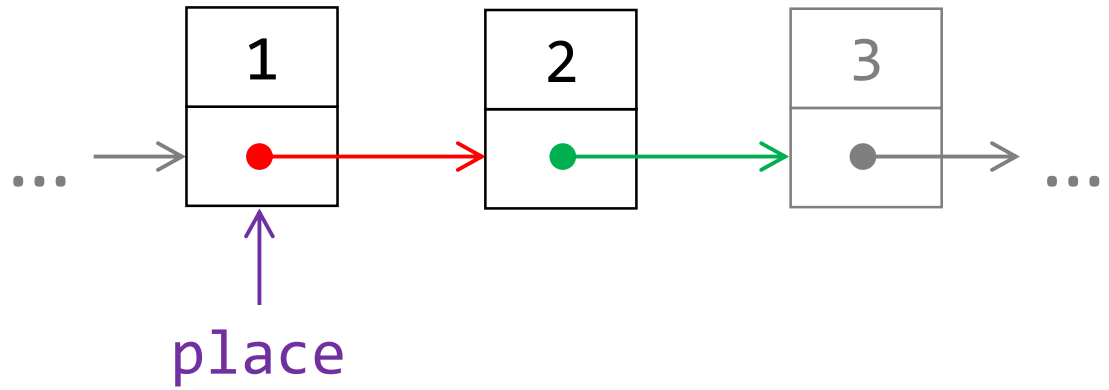


Вставка элемента: в) в произвольное место

Было:
список, новый узел
и указатель на узел в списке



Стало:
список с новым узлом в конце



Проход по списку (enumeration)

- Размер списка неизвестен:

```
Node* current = first;
while (current)
   current = current->next;
```

- До элемента № index:

```
Node* current = first;
for (i = 0; i < index; ++i)
   current = current->next;
```

- Интервал [first; last)
  или любой другой:

```
Node* current = fist;
while (current != last)
   current = current->next;
```

➢ Можно проверять current:

```
while (current &&
       current != last) {
   …
}
```

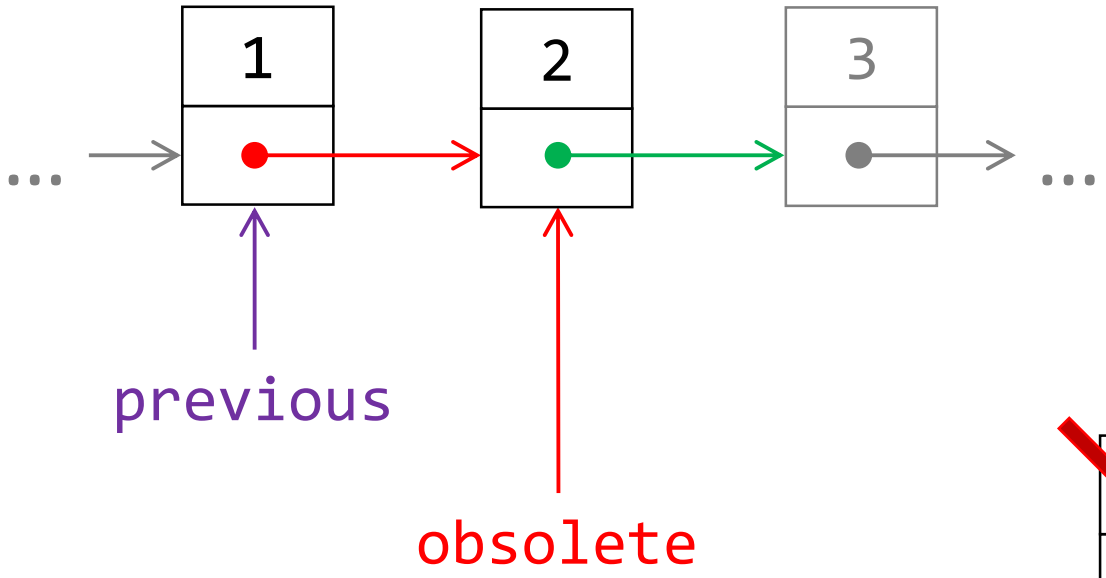
Проход по списку с запоминанием предыдущего элемента

```
Node* previous = nullptr;
Node* current = first;
while (current) {
      …
      previous = current;
      current = current->next;
}
```

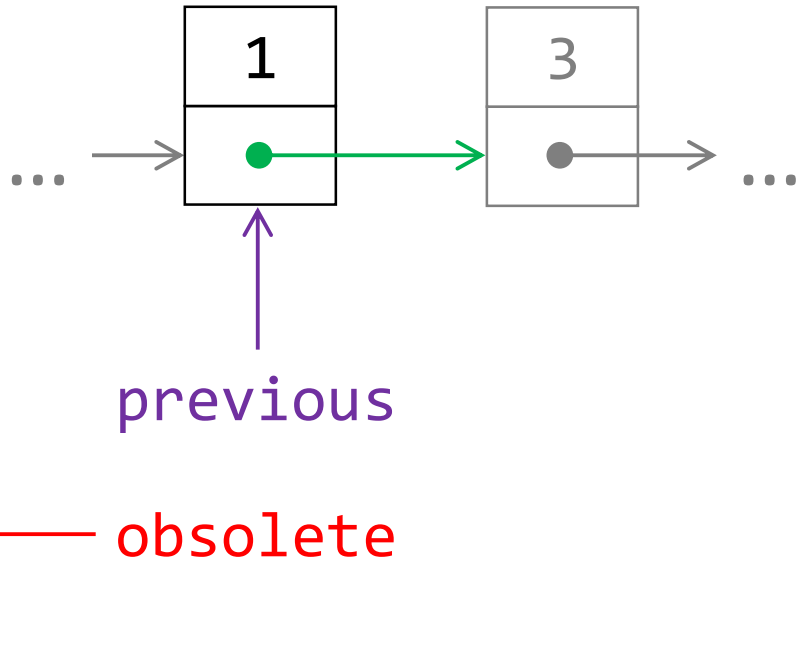
1. `previous == nullptr`
   `current == first`
2. `previous == first`
   `current == first->next`
3. `previous == first->next`
   `current == first->next->next`
4. `...`

Удаление элемента

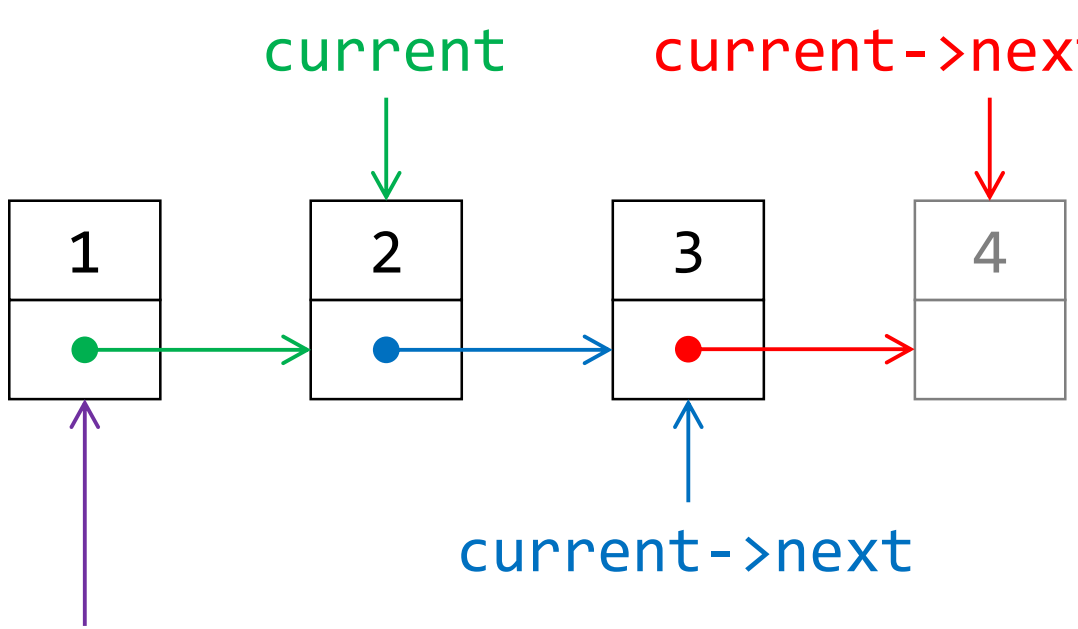
Было:
список и узел к удалению



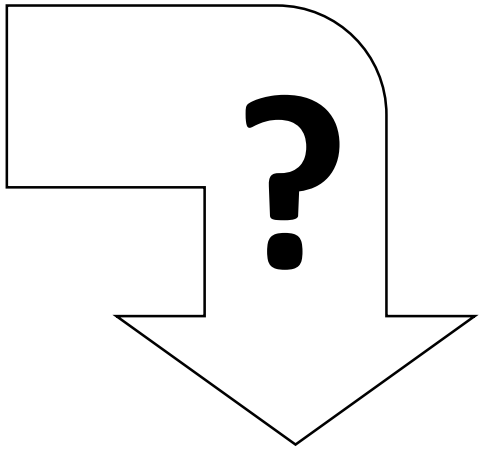
Стало:
список без указанного узла

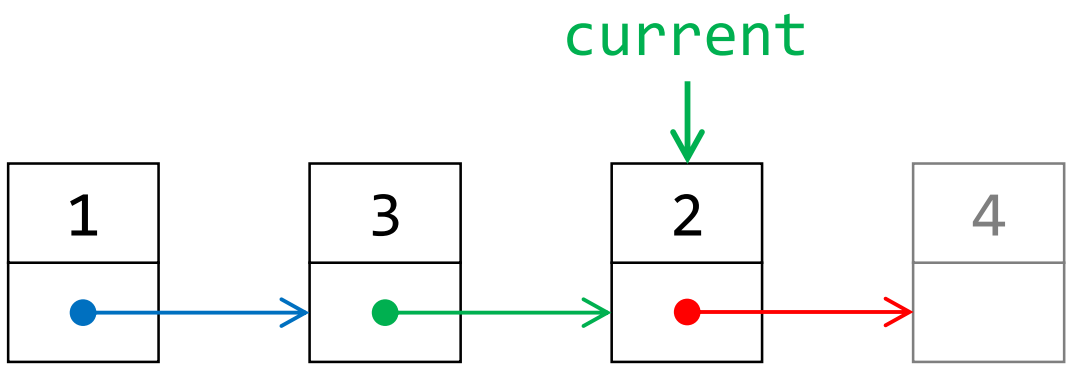


Обмен элементов местами



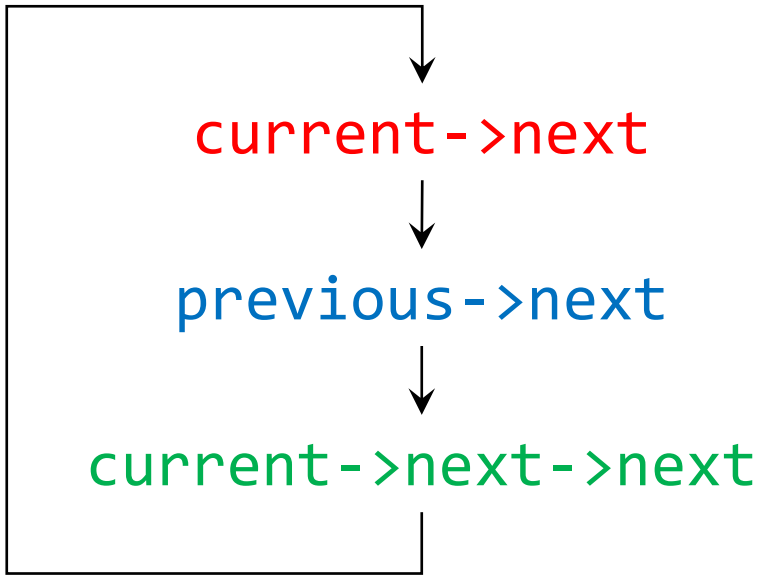
previous->next == current





Обмен элементов местами (итог)

- Задача сводится к циклической перестановке:



- Особый случай: начало списка

  `previous->next` — это `first`

  Решение — ссылка:

```
Node* &that = начало ?
    first : previous->next;
```

- Особый случай: конец списка

  `current->next` — это `last`

  Решение: `last = current` в конце.