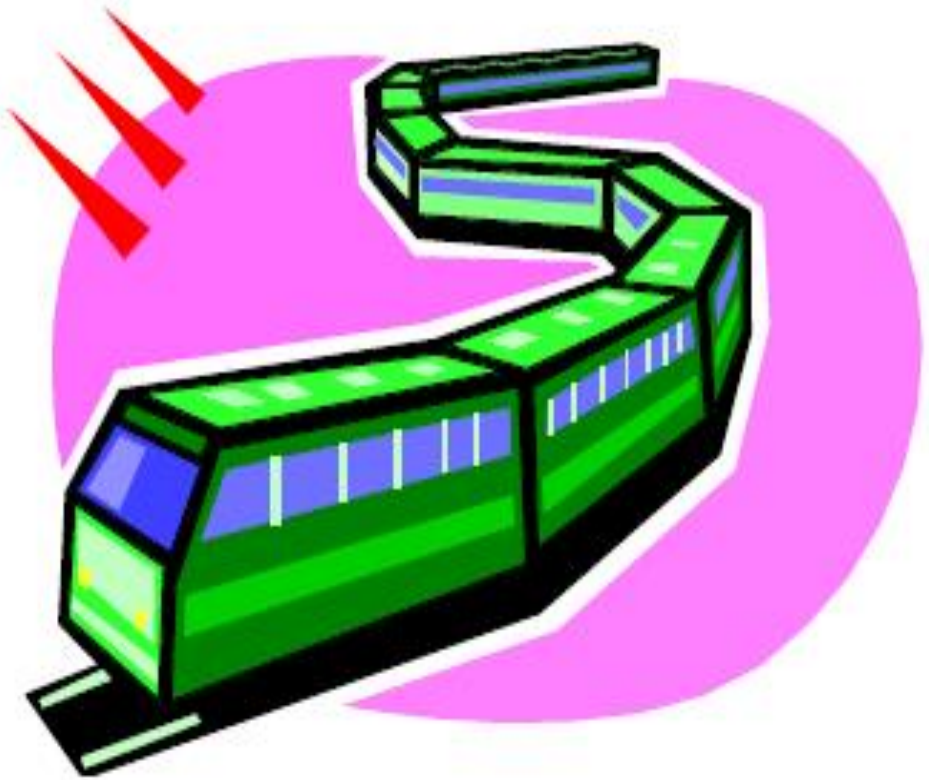
Data Structures & Algorithms

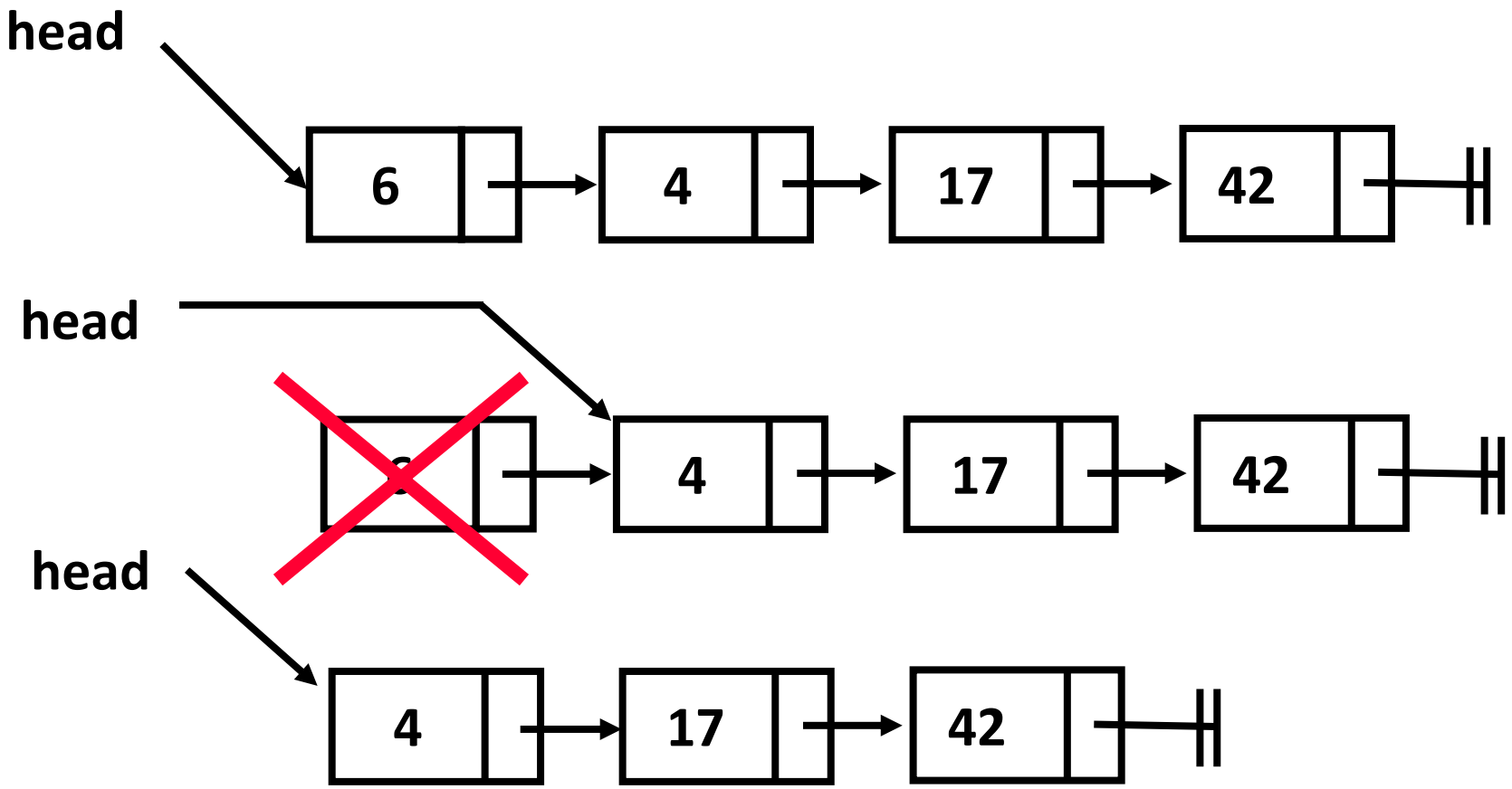
Linked List



Deleting a node from a Singly-linked list

- Deleting the first node.
- Deleting the last node.
- Deleting an intermediate node.

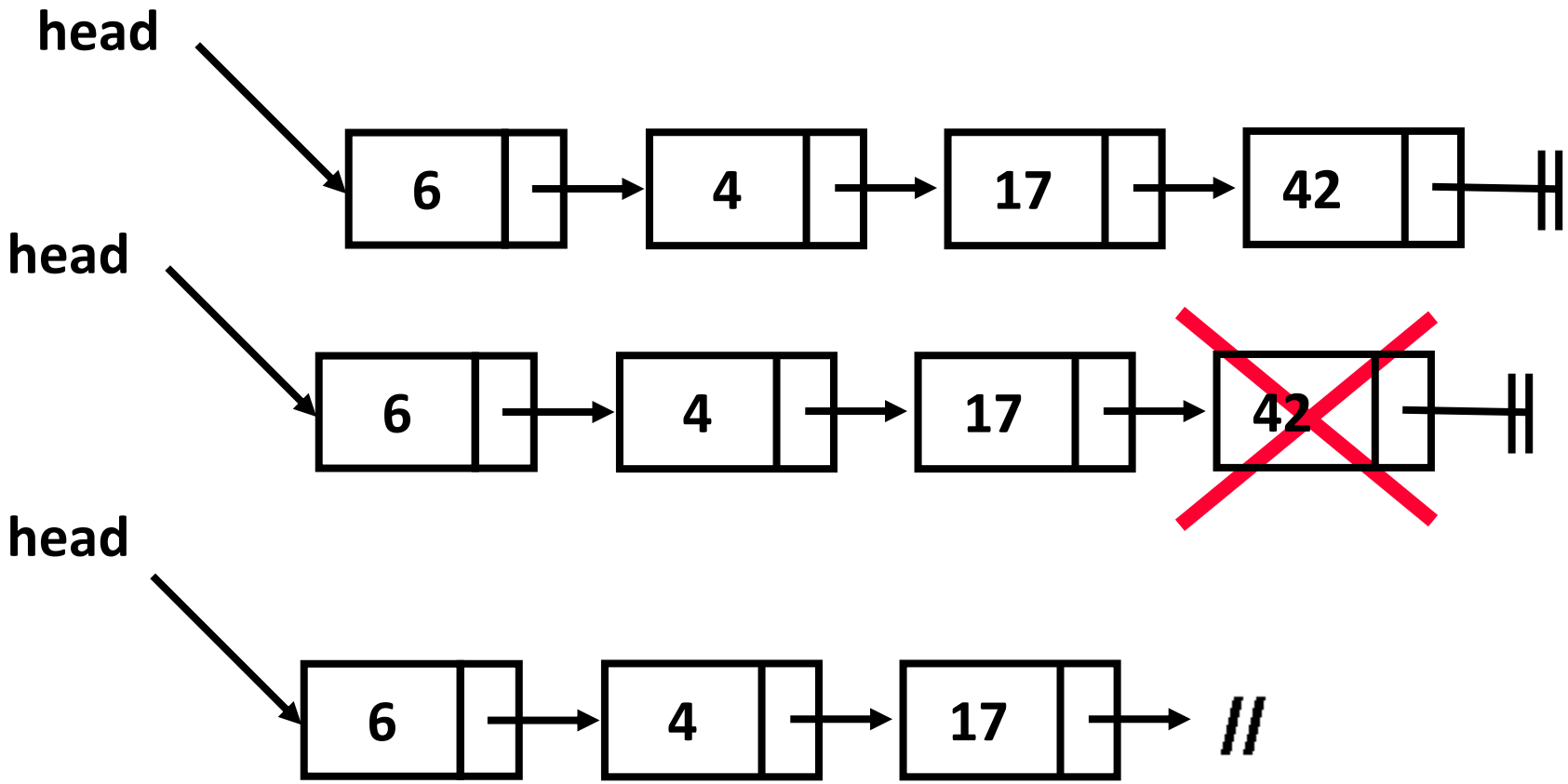
Deleting a node from the beginning in a Singly-linked list



Deleting a node from the beginning in a Singly-linked list

```
delete_first(head)
{
   tmp = head;
   head = head->next;
   free(tmp);
}
```

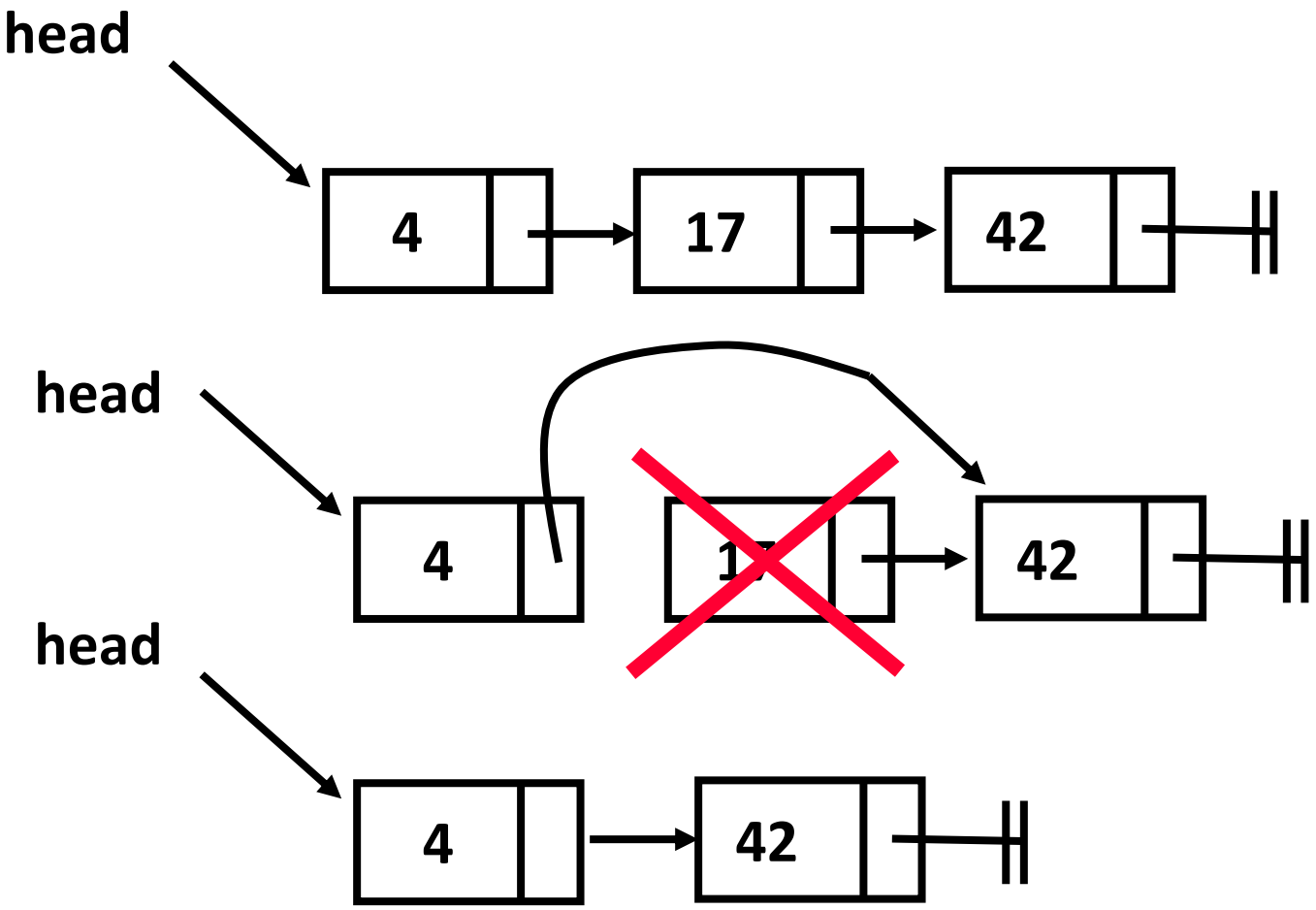
Deleting a node from the end in a Singly-linked list



Deleting a node from the end in a Singly-linked list

```
delete_first(head)
{
      currNode = head;
      prevNode = head;
      while (currNode->next != NULL)   {
        prevNode = currNode;
        currNode = currNode->next;
      }
      prevNode->next = NULL;
      free(currNode);
 }
```

Deleting a given node in a Singly-linked list



Deleting a given node in a Singly-linked list

Assume that item is present and the list contains unique items.

```
delete_item(head, item)
{
      currNode = head;
      prevNode = head;
      while (currNode->data != item)   {
        prevNode = currNode;
        currNode = currNode->next;
      }
      prevNode->next = currNode->next;
      free(currNode);
 }
```

Deleting a node after a given element in a Singly-linked list

Write an algorithm to delete a node after a given node/element in a singly linked list.

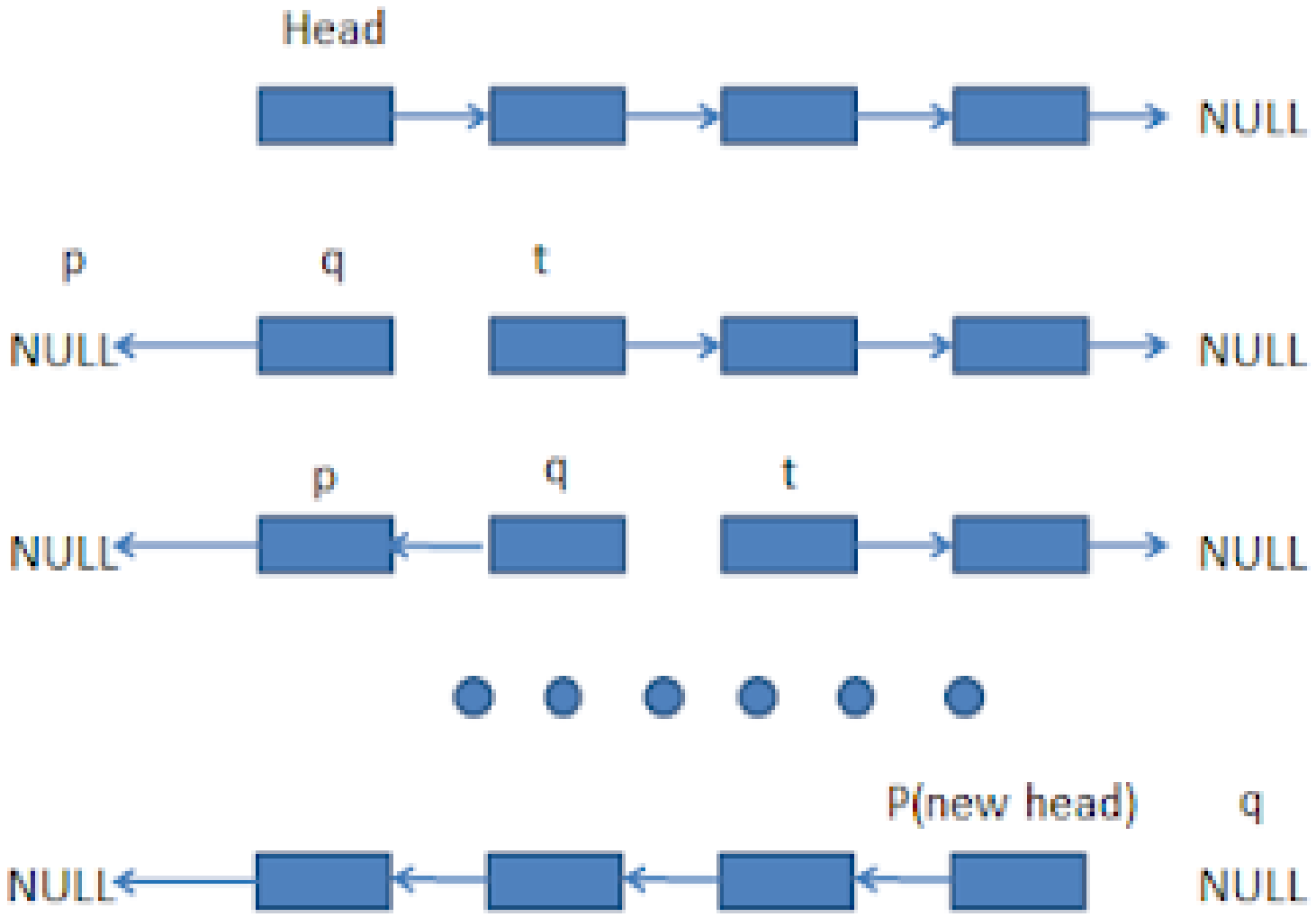
```
delete_item(head, after)
{
      ----------
      ----------
 }
```

Complexity analysis for insertion & deletion Singly-linked list

Insertion and deletion at the beginning of the singly-linked lists are very fast, O(1) time.

Insertions and deletion at the end can be supported in O(n) time.

Reversing the elements of a Singly-linked list



Reversing the elements of a Singly-linked list

```
ReverseSinglyList(head)
{
  prevNode = NULL;
  currNode = head;
  nextNode = currNode->next;
  while(nextNode != NULL)    {
     prevNode = currNode;
     currNode = nextNode;
     nextNode = currNode->next;
     currNode->next = prevNode;
  }
  head = currNode;
}
```

For you to do

Write an algorithm to delete all nodes from a singly linked list.

Write an algorithm to delete all occurrences of a given element from a singly linked list.

Write a recursive algorithm to reverse the elements of a singly linked list.