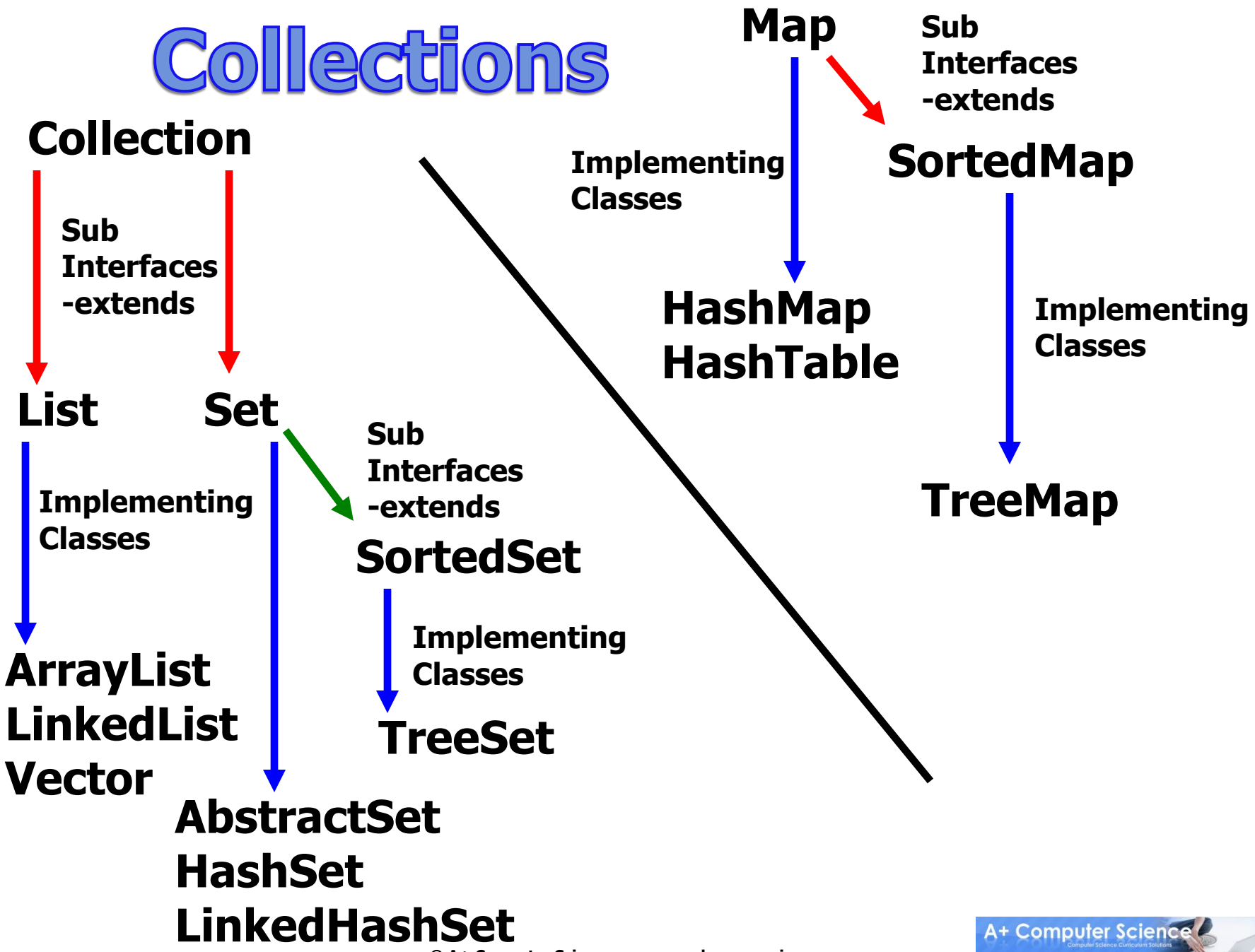
A+ Computer Science

Linked lists

Java LinkedList

Collections



LinkedList Methods

LinkedList

frequently used methods

Name	Use
add(x)	adds item x to the list
set(x,y)	set location x to the value y
get(x)	get the item at location x
size()	returns the # of items in the list
remove()	removes an item from the list
clear()	removes all items from the list

```
import  java.util.LinkedList;
```

add() method

```
LinkedList<String> list;
list = new LinkedList<String>();

list.add("c");
list.add("b");
list.add("a");
list.add(1, "d");

out.println(list);
```

OUTPUT

[c, d, b, a]

get() method

```
LinkedList<String> list;
list = new LinkedList<String>();

list.add("c");
list.add("b");
list.add("a");
list.add(1, "d");

out.println(list.get(0) );
out.println(list.get(1) );
out.println("first " + list.getFirst());
out.println("last " + list.getLast());
```

OUTPUT

```
c
d
first c
last a
```

linkedlistadd.java
linklistget.java

Work on Programs!

Crank

Some Code!

Linked Lists

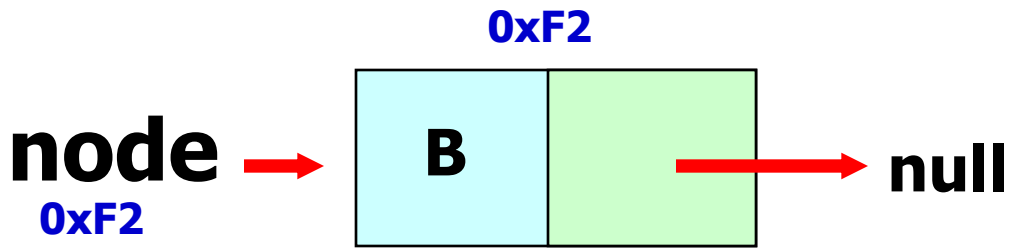
A linked list is a group of nodes. Each node contains a value and a reference to the next node in the list.

Simple Node Class

```
public class Node
{
  private Comparable data;
  private Node next;

  public Node(Comparable dat, Node nxt)
  {
    data=dat;
    next=nxt;
  }
}
```

A Single Node



A node typically has a data component and a reference to the next node.

Linkable Interface

```
public interface Linkable
{
   Comparable getValue();
   Linkable getNext();
   void setNext(Linkable next);
   void setValue(Comparable value);
}
```

```
public class ListNode implements Linkable
{
  private Comparable listNodeValue;
  private ListNode nextListNode;

  public ListNode(){
    listNodeValue = null;
    nextListNode = null;
  }

  public ListNode(Comparable value, ListNode next)
  {
    nextListNode = next;
    listNodeValue = value;
  }

 //other methods not shown
 //refer to the Linkable interface
}
```

ListNode Class

This ListNode class is similar to the AP ListNode.

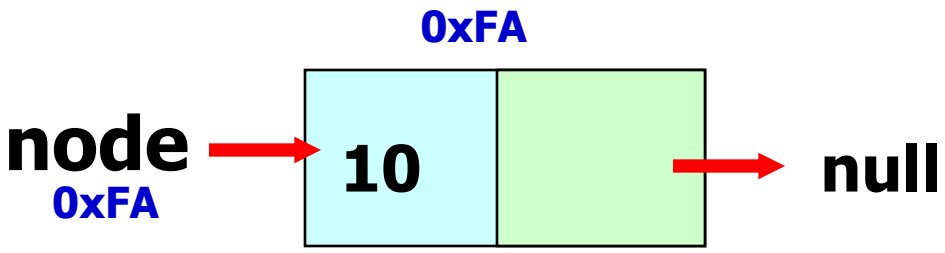
You can obtain the official AP ListNode class from the college board website. You will be provided with a copy of the AP ListNode class when you take the AP Computer Science AB exam.

Creating a Single ListNode

```
Linkable node = new ListNode("10", null);
out.println(node.getValue());
out.println(node.getNext());
```

OUTPUT

```
10
null
```



onenode.java

Linking Nodes

Linking Nodes

OUTPUT

10
12
11

```
ListNode x = new ListNode("10",
                 new ListNode("11",
                 new ListNode("12",null)));

out.println(x.getValue());
out.println(x.getNext().getNext().getValue());
out.println(x.getNext().getValue());
```

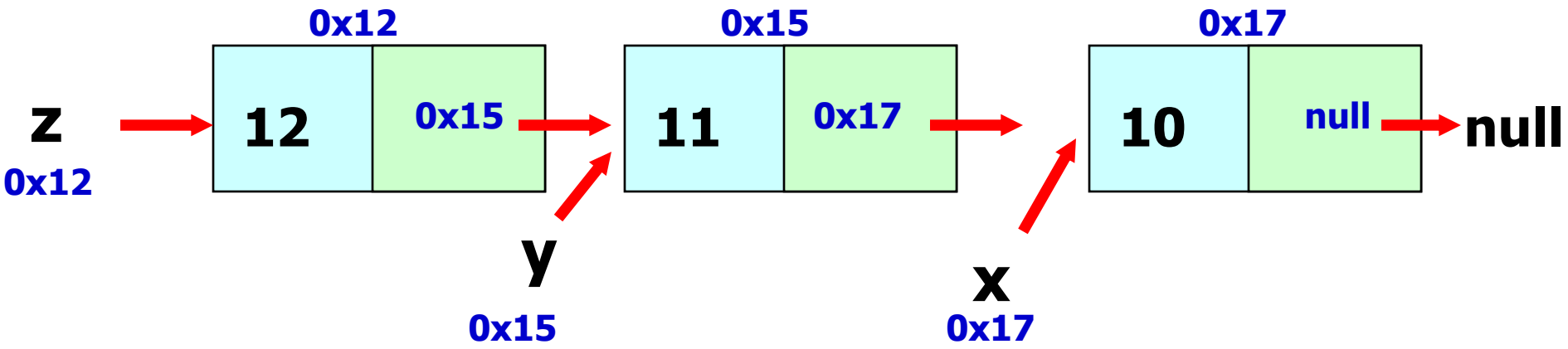
linkone.java

Linking Nodes

```
ListNode x = new ListNode("10", null);

ListNode y = new ListNode("11",x);

ListNode z = new ListNode("12",y);
```

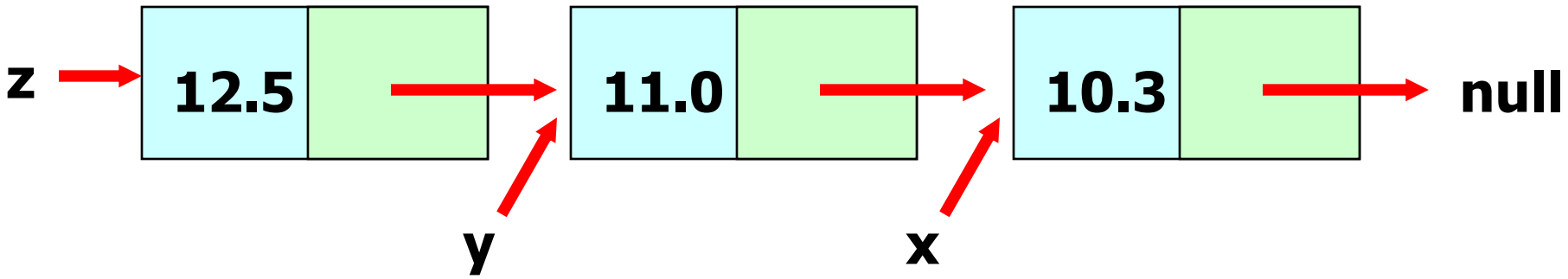


Linking Nodes

```
ListNode x = new ListNode(10.3, null);

ListNode y = new ListNode(11.0, x);

ListNode z = new ListNode(12.5, y);
```



Linking Nodes

```
ListNode x = new ListNode(10.3, null);
ListNode y = new ListNode(11.0, x);
ListNode z = new ListNode(12.5, y);

out.println(z.getValue());
out.println(z.getNext().getNext().getValue());
out.println(z.getNext().getValue());
```

OUTPUT

12.5
10.3
11.0

linklistdemo.java
linktwo.java

Processing Lists with Loops

Displaying a List

```
ListNode x = new ListNode("10",
                 new ListNode("11",
                 new ListNode("12",null)));

while( x != null )
{
  out.println( x.getValue() );
}
```

OUTPUT

```
10
10
10
. . .
```

printone.java

Displaying a List

```
ListNode x = new ListNode("10",
                  new ListNode("11",
                  new ListNode("12",null)));

while( x != null )
{
   out.println( x.getValue() );
   x = x.getNext();
}
```

OUTPUT

```
10
11
12
```

printtwo.java

Adding All Nodes

```
ListNode x = new ListNode(11,
                 new ListNode(8,
                 new ListNode(5,null)));

int sum=0;
while( x != null )
{
  sum = sum + (Integer)x.getValue();
  x = x.getNext();
}
out.println(sum);
```

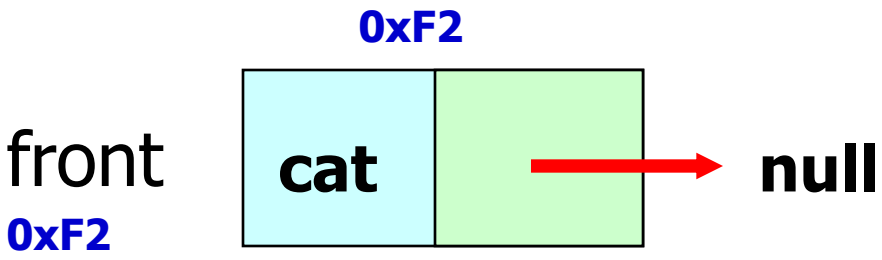
OUTPUT

24

sumone.java

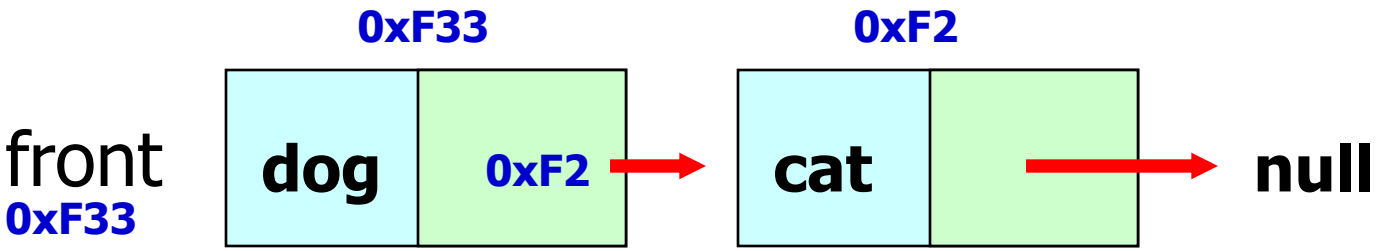
Adding Nodes

Adding List Nodes



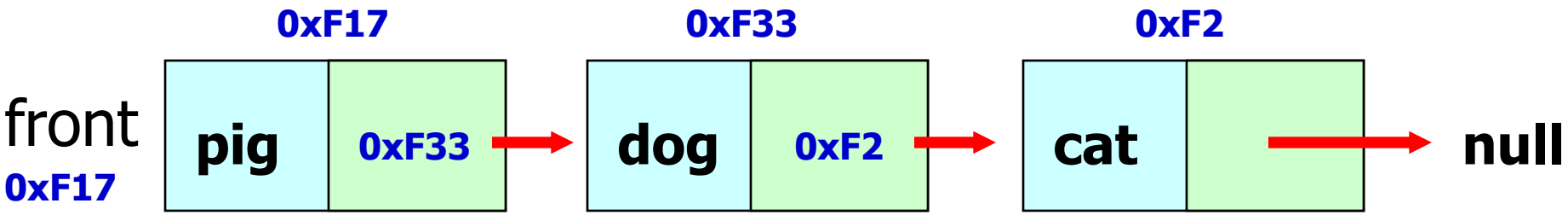

```
front = new ListNode(word, null);
```

Adding List Nodes




```
front = new ListNode(word, front);
```

Adding List Nodes




```
front = new ListNode(word, front);
```

Adding List Nodes

```
ListNode front=null;
front = new ListNode("10", front);
front = new ListNode("11",front);
front = new ListNode("12",front);

out.println(front.getValue());
out.println(front.getNext().getNext().getValue());
out.println(front.getNext().getValue());
```

OUTPUT

```
12
10
11
```

add.java

Searching Lists

Searching for Values

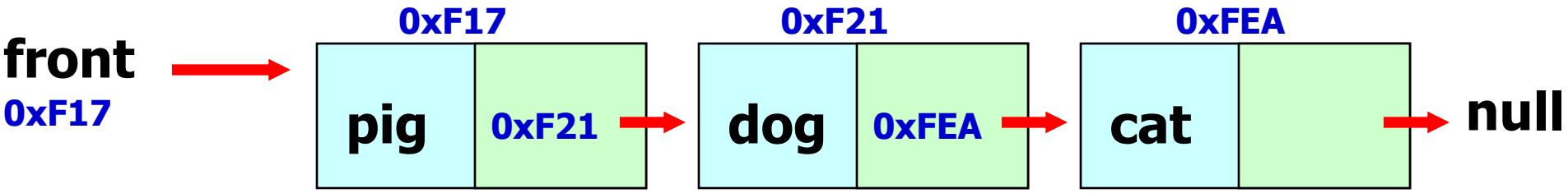
```
ListNode list = front;
while ( there are more nodes to check )
{
   if( a node containing the value was found )
      return true;
   move to the next node to check
}
return false;
```

contains.java

Deleting Values

Deleting First Node

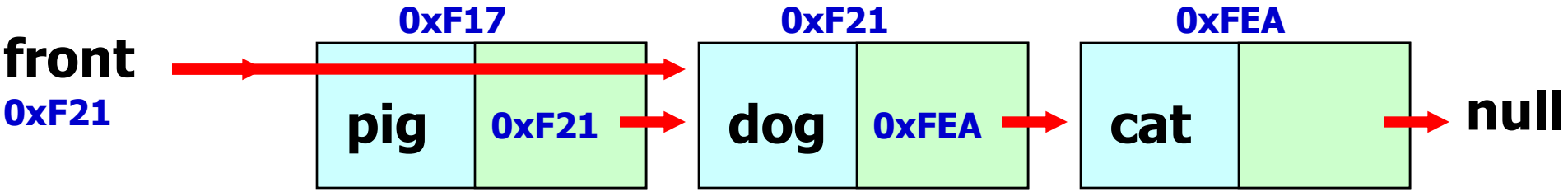
pig is to be removed.



front refers to the 1st node in the list.

Deleting First Node

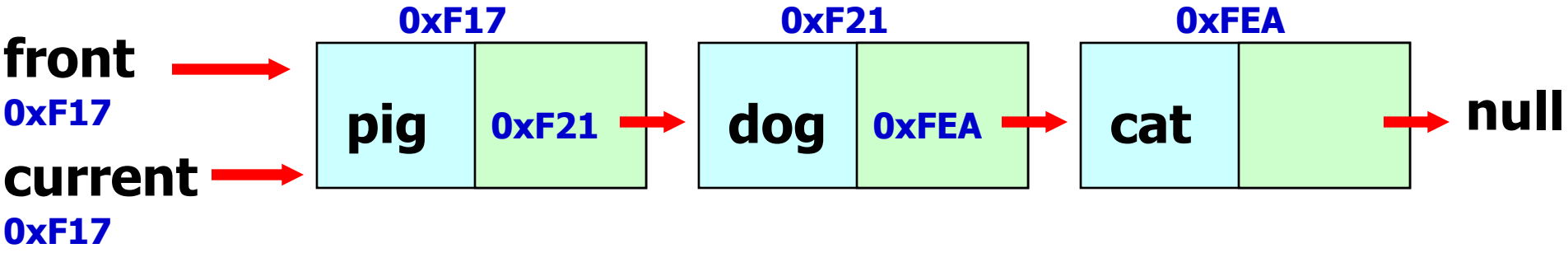
```
front = front.getNext();
```



front moves up one node.

Deleting Any Node

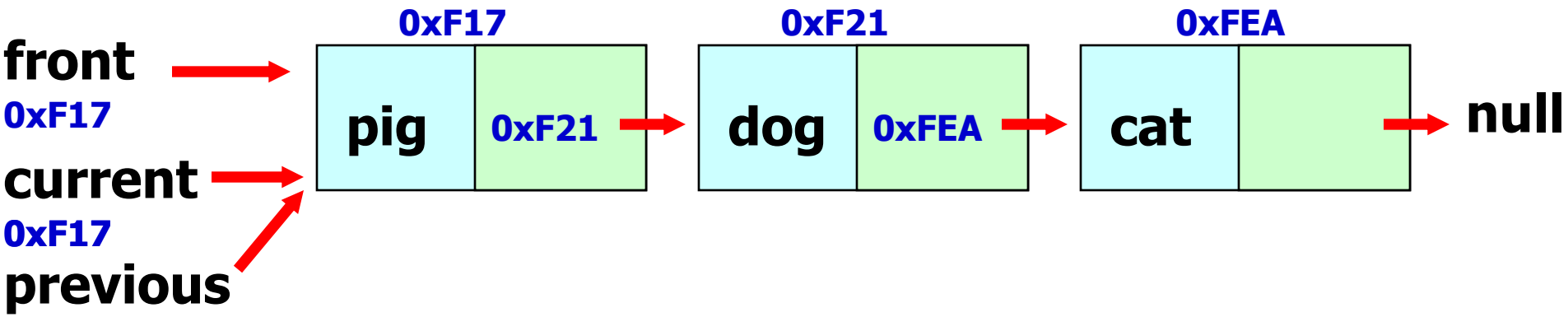
dog is to be removed.
current = front;



front and current store the same memory address.

Deleting Any Node

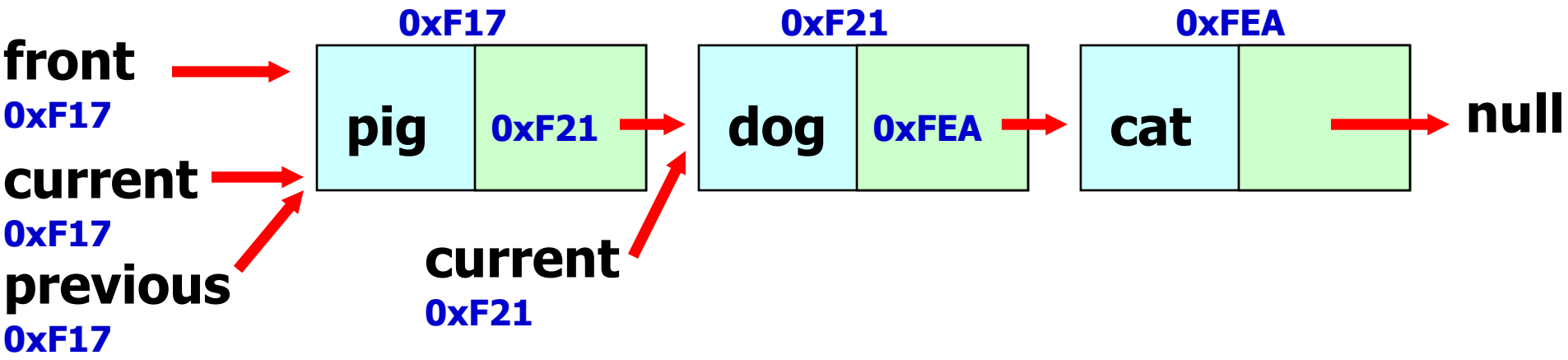
previous = current;



front, current, and previous all store the same memory address.

Deleting Any Node

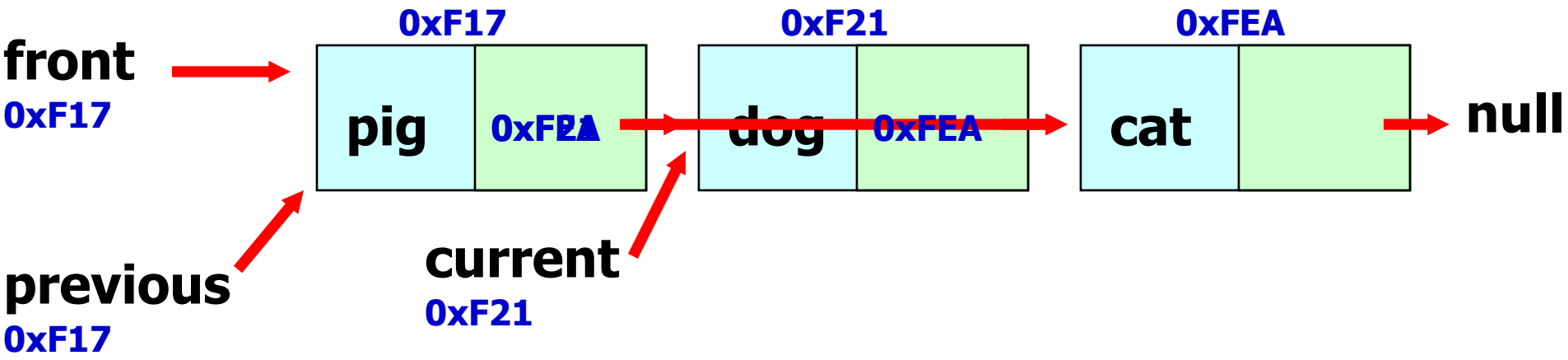
```
current=current.getNext();
```



current moves forward one node.

Deleting Any Node

```
previous.setNext(current.getNext());
```



Found dog. Removed dog from the list.

Deleting Any Node

Some things you have to account for!

1. What if the linked list is null?
2. What if I need to remove the 1st node?
3. How do I process the remaining nodes?
4. Do I remove more than 1 occurrence of the same value or just the 1st one?

remove.java

Work on Programs!

Crank

Some Code!

A+ Computer Science

Linked lists