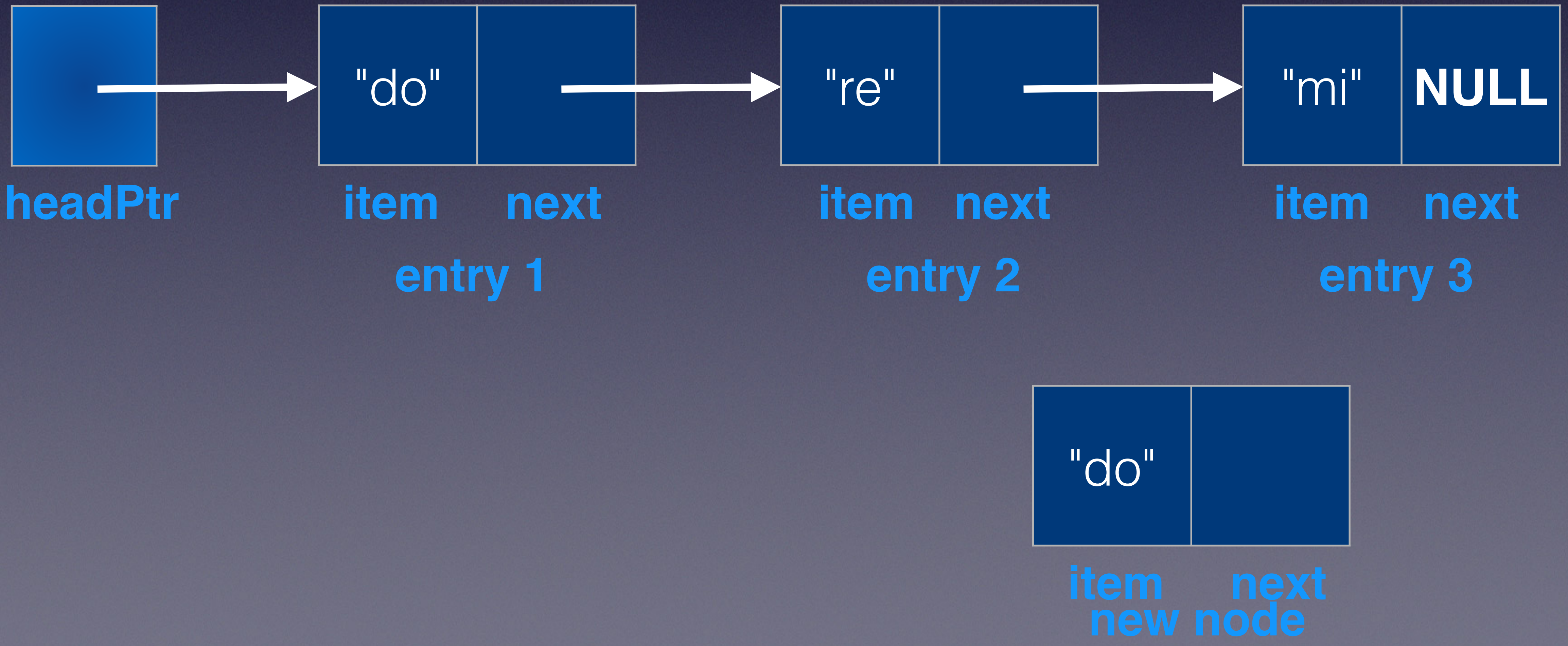
Linked Lists
Recursion

CS110C
Max Luttrell, CCSF

recursion in linked list

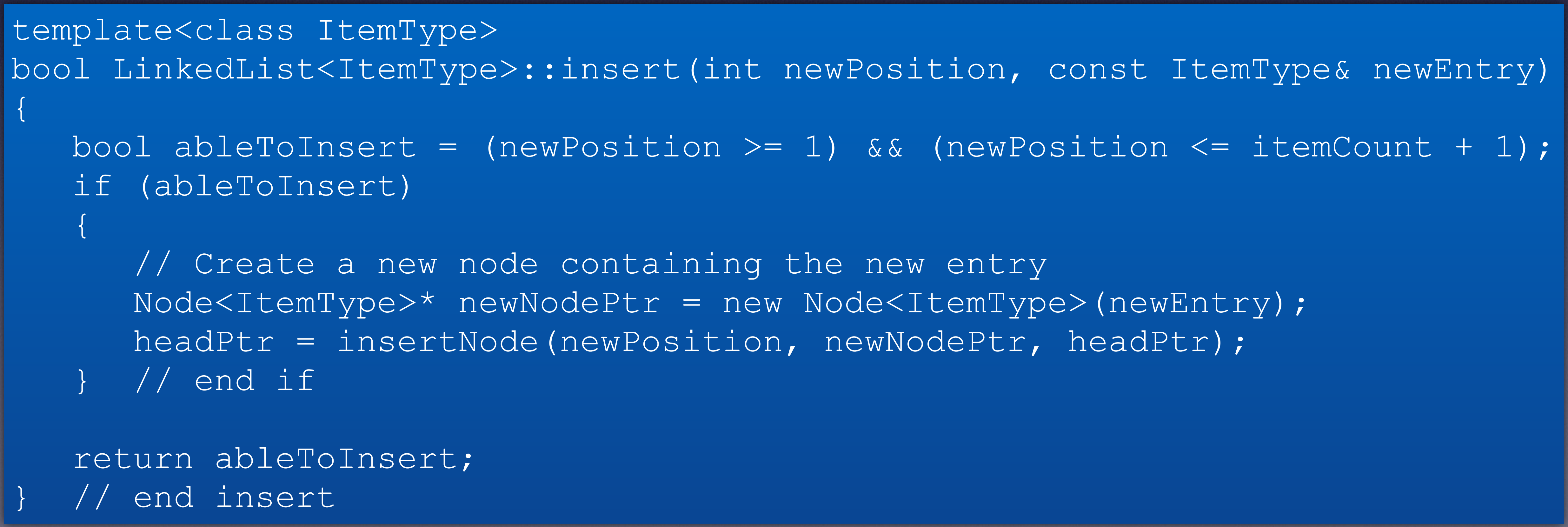
- Let's say I want to insert another "do" at position 3 of my list. We can use recursion to do this instead of our previous iterative approach.

```
insert(3,"do")
```



recursive insert

- We still need to provide our standard List ADT insert() function.
- It will call a helper function (private member function) called **insertNode** to do our recursion.



```
template<class ItemType>
bool LinkedList<ItemType>::insert(int newPosition, const ItemType& newEntry)
{
   bool ableToInsert = (newPosition >= 1) && (newPosition <= itemCount + 1);
   if (ableToInsert)
   {
      // Create a new node containing the new entry
      Node<ItemType>* newNodePtr = new Node<ItemType>(newEntry);
      headPtr = insertNode(newPosition, newNodePtr, headPtr);
   }  // end if

   return ableToInsert;
}  // end insert
```

insertNode() - recursive

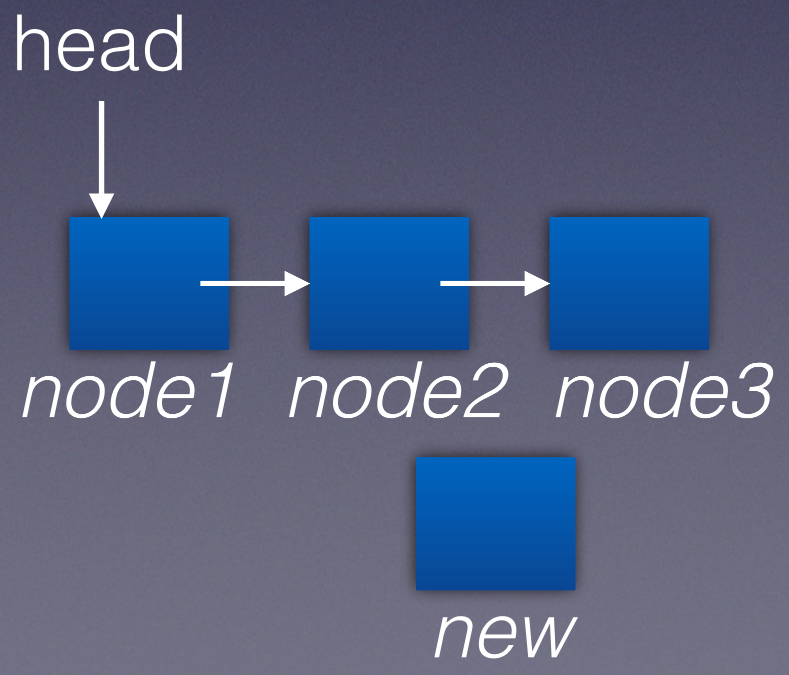
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                   Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                             subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

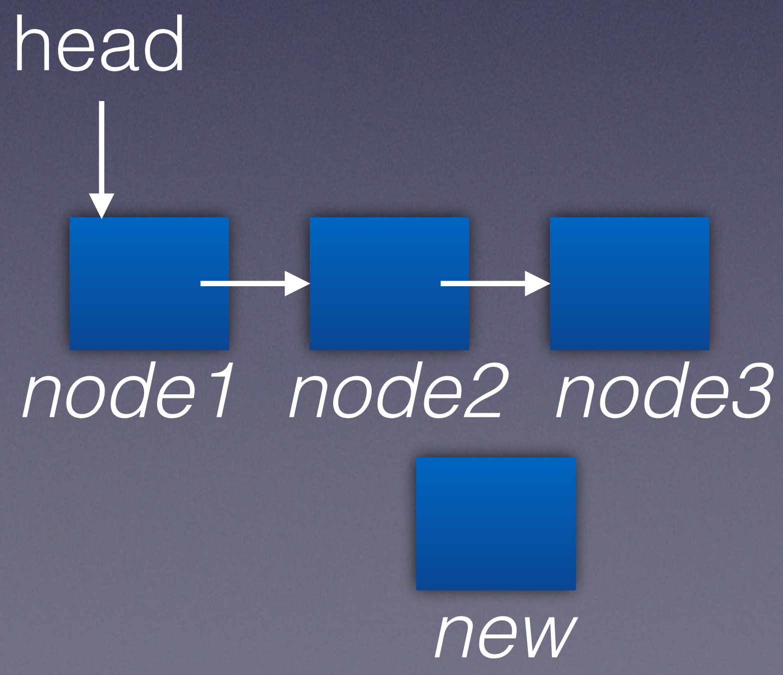
```
insertNode(3, new, head)
```



```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if  (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

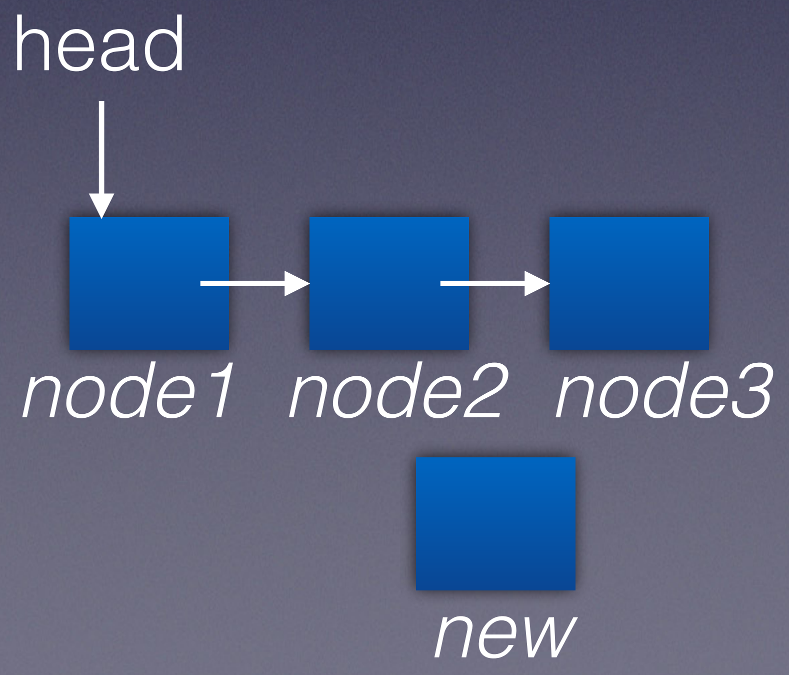
```
insertNode(3, new, h)
 - insertNode(2, new, node2)
```



```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                 Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

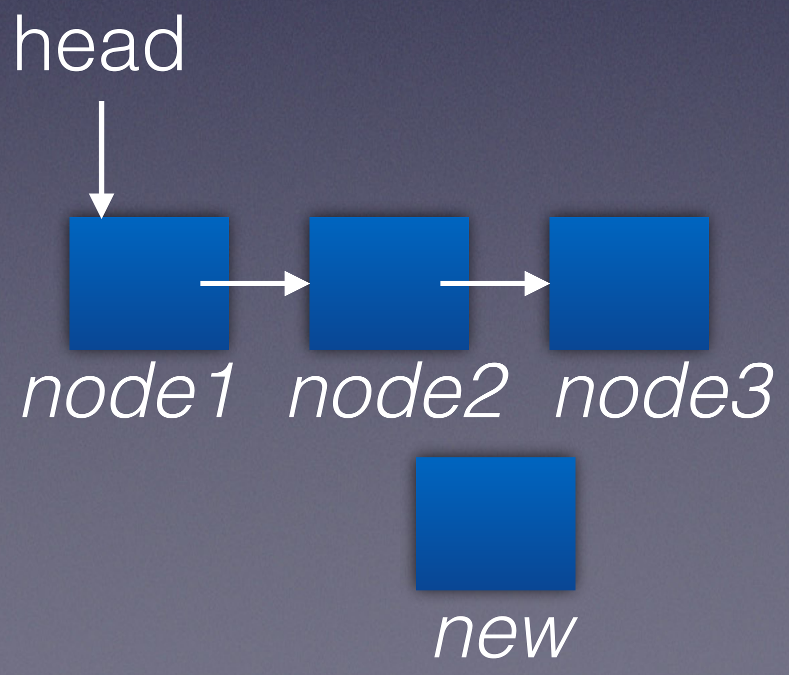
```
insertNode(3, new, h)
 - insertNode(2, new, node2)
   - insertNode(1, new, node3)
```



```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                   Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

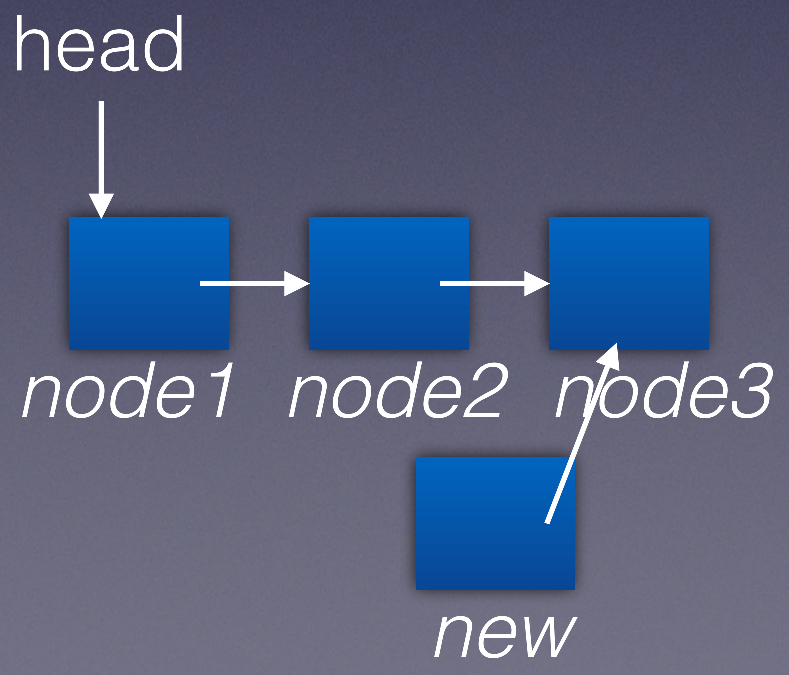
```
insertNode(3, new, h)
 - insertNode(2, new, node2)
   - insertNode(1, new, node3)
```



```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                Node<ItemType>* newNodePtr,
                                                 Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

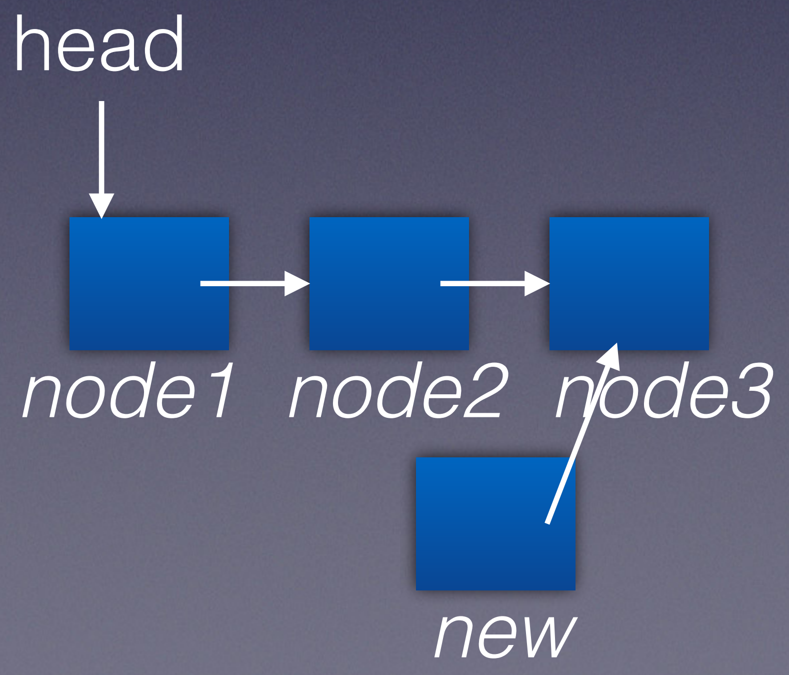
```
insertNode(3, new, h)
 - insertNode(2, new, node2)
   - insertNode(1, new, node3)
```



```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

```
insertNode(3, new, h)
 - insertNode(2, new, node2)
   - insertNode(1, new, node3)
```



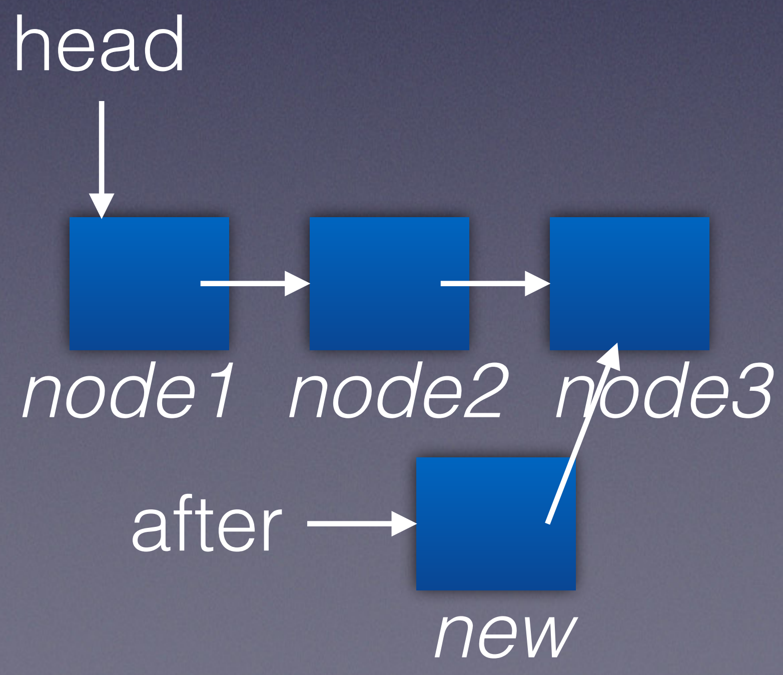
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

 - **insertNode(2, new, node2)**

~~- insertNode(1, new, node3)~~



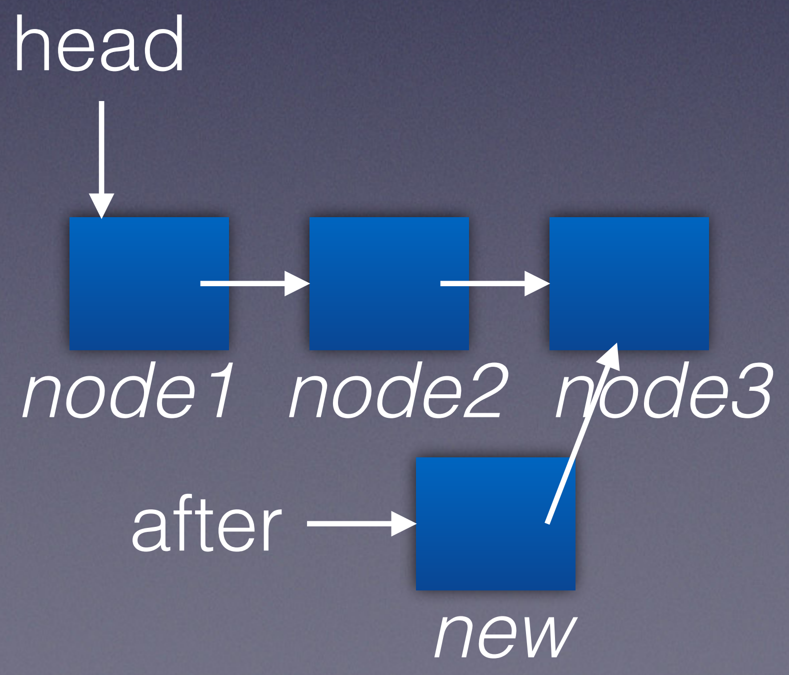
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                   Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                             subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

- insertNode(2, new, node2)

~~- insertNode(1, new, node3)~~



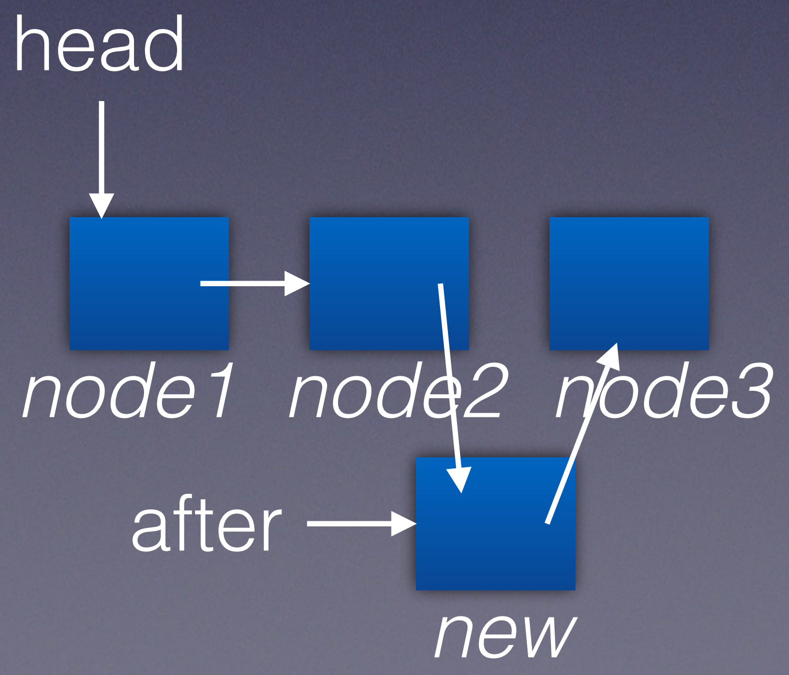
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                 Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

 - **insertNode(2, new, node2)**

~~ - insertNode(1, new, node3)~~



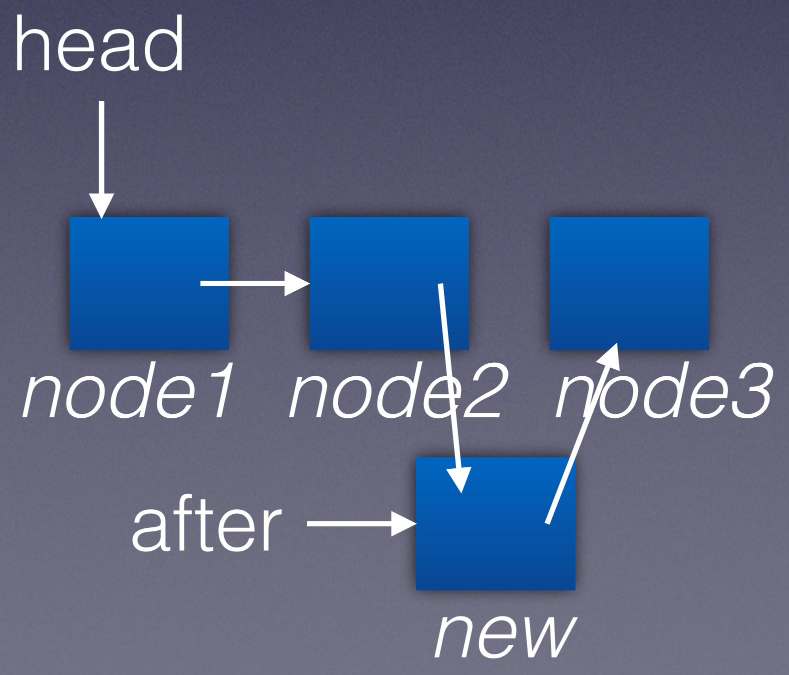
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

- insertNode(2, new, node2)

~~- insertNode(1, new, node3)~~



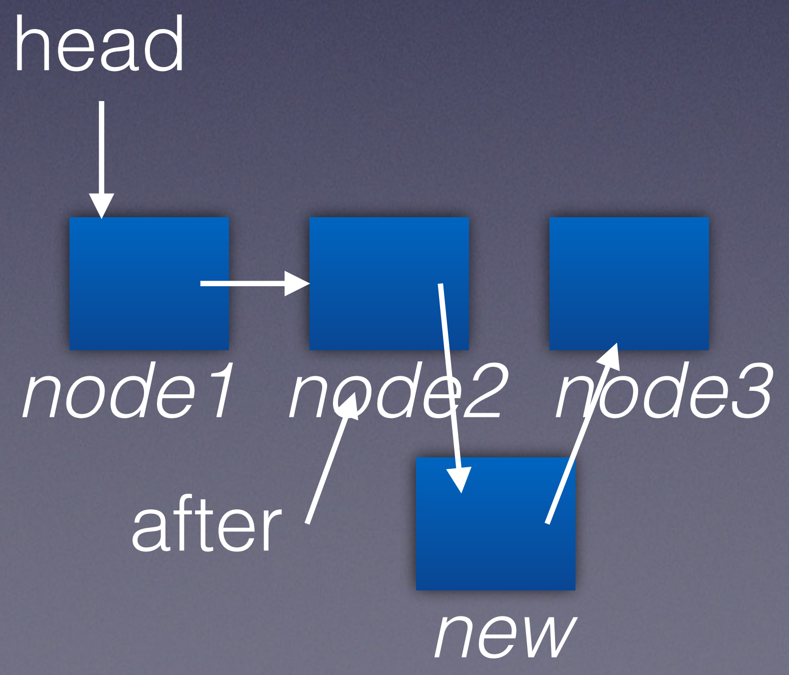
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                 Node<ItemType>* newNodePtr,
                                                 Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                            subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

~~- insertNode(2, new, node2)~~

~~- insertNode(1, new, node3)~~



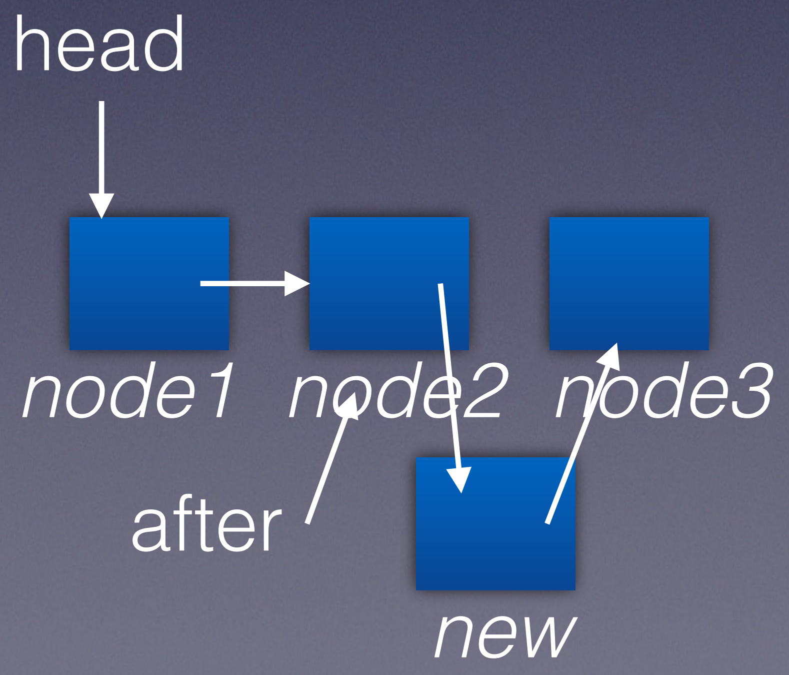
```
template<class ItemType>
Node<ItemType>* LinkedList<ItemType>::insertNode(int position,
                                                  Node<ItemType>* newNodePtr,
                                                  Node<ItemType>* subChainPtr)
{
   if (position == 1)
   {
      // Insert new node at beginning of subchain
      newNodePtr->setNext(subChainPtr);
      subChainPtr = newNodePtr;
      itemCount++;  // Increase count of entries
   }
   else
   {
      Node<ItemType>* afterPtr = insertNode(position - 1, newNodePtr,
                                             subChainPtr->getNext());
      subChainPtr->setNext(afterPtr);
   }  // end if

   return subChainPtr;
}  // end insertNode
```

insertNode(3, new, h)

~~- insertNode(2, new, node2)~~

~~- insertNode(1, new, node3)~~


head
node1
node2
node3
after
new