Watch Window

Using the Watch Window to View List Structures

Once the program is successfully compiled and ready to execute:

- Set breakpoints at the locations(s) in the code where you wish to view the list’s contents. Position the cursor on a line of code, right click, and select Insert/Remove Breakpoint. Select Build -> Debug -> Go from the menu or hit F5 to begin debugging.

The program will stop at the first breakpoint. If the Watch Window is not visible, select View -> Watch from the menu.
Watching Lists

Viewing Linked List Contents (continued)
- Double-click the structure variable name in the Source
  window to highlight it, then drag and drop it into the Watch
  Window.

- There will be a plus (+) sign to the left of the variable name
  in the name field of the Watch Window. Click on the plus
  (+) to “expand” the variable. This will in effect display the
  “contents” of the structure with each subsequent expansion
  of the variable. A minus (-) sign indicates that the variable is
  already fully expanded.

Watching Linked-List Variables

Sample Program

```cpp
/* Sample program to demonstrate use of the
 variables window. */

#include <iostream.h>

typedef struct node {
   int val;
   node* nextnode;
};

int i;
node* listhead, * currnode;

void main (void)
{
   listhead = new node;
   listhead->val = 0;
   currnode = listhead;
   for (i=1; i<=2; i++) {
      currnode->nextnode = new node;
      currnode = currnode->nextnode;
      currnode->val = i;
   }
   currnode->nextnode = NULL;
   for (currnode = listhead; currnode != NULL; currnode=currnode->nextnode)
      cout << currnode->val << " ";
   return;
}
```

create a list of size 3
and set the last node pointer to NULL
Dereferencing Invalid Addresses

- The tail of the list, when set to `NULL` will appear in the watch window as `0x00000000`. Any values at this point in the list will be inaccessible.

![List diagram](image)

- If the last node is NOT set to `NULL`, the values will still be accessible but will most likely produce a run-time error or undesired program behavior, even in the debugger.

```c
currnside = currnode->nextnode;
currnside->val = i;
// last node not set to NULL
for (currnside = listhead; currnode != NULL;
    currnode = currnode->nextnode)
    cout << currnode->val << " ";
```

![Debug window](image)