Quiz #1

(February 22, 2012)

Your Names: _____

Question #1. Consider the following independent code snippets (a)-(d).
What, if any, coding errors are exhibited in each? Be brief, specific, and complete.
Sketch a diagram of the situation in memory in each case – to help you (reason about the code) and to help me (see your thinking more clearly and grade effectively).

(a)
```
int *pnInt1 = new int(10);
int *pnInt2 = *pnInt1;
```

(b)
```
int nNum, *pnInt1, *pnInt2 = new int;
pnInt1 = pnInt2;
*pnInt1 = nNum * 2;
pnInt2 = &nNum;
delete pnInt1;
cout << *pnInt2;
delete pnInt2;
```

(c)
```
int pnNum[3], &rnInt = pnNum[2];
int *pnInt = &rnInt;
if (pnInt != NULL && rnInt = 0)
    pnInt = pnNum + 1;
```

(d)
```
int myFunc( int nNum )
{
    int *pnInt = new int;
    *pnInt = nNum * nNum;
    cout << *pnInt;
    return *pnInt;
}
```

Question #2. The following are independent functions, some of which use the `ListNode` declaration below. What coding errors, if any, are present in each function? Be brief, specific, and complete.

```
struct ListNode
{
    int        item;
    ListNode * next;
};
```

(a)
```
// prints the items stored in the list, starting from a given position
void printListItems( ListNode * poListPos )
{
    ListNode * poCurr = poListPos;
    while (poCurr.next != NULL)
    {
        cout << poCurr.item << endl;
        poCurr = poCurr.next;
    }
}
```

(b)
```
// deletes the first node of a linked list
void deleteFirst( ListNode * poListHead )
{
    ListNode * pTemp = poListHead;
    poListHead = poListHead->next;
    delete pTemp;
}
```

(c)
```
// puts a given character string in quotes, e.g. quiz becomes "quiz"
char * addQuotesToString( char * pcOriginal )
{
    // allocate space for the new string; it needs two extra cells
    int nNewLength = strlen(pcOriginal) + 2;
    char * pcQuoted = new char[nNewLength];

    // copy the original string to the destination,
    // starting with an offset to allow space for the initial quote: '"'
    for (int i=1; i<nNewLength-1; i++)
        pcQuoted[i] = pcOriginal[i-1];

    // add the quote symbols: one at the start, and one at the end
    pcQuoted[0] = '"';
    pcQuoted[nNewLength-1] = '"';

    return pcQuoted;
}
```