Quiz 8 Name:________________________

Question 1. (12pt) Given an array of MyDataType:

```
MyDataType myArray[10];
```

and, my linked list node definition:

```
struct LinkNode {
    MyDataType *data;
    LinkNode   *next;
    // Constructor
    LinkNode(MyDataType *d, LinkNode *nxt) : data(d), next(nxt) {}
};
```

a. (2pt) Please show how you would allocate and initialize, myHead, a dummy header for a linked list of LinkNodes.

LinkNode*myHead = new LinkNode(nullptr, nullptr);

b. (4pt) Please complete the following two lines of code to dynamically allocate and initialize two LinkNodes, firstNode and secNode, to point to the first two elements of MyArray and both nodes with nullptr in their next fields.

LinkNode *firstNode = new LinkNode(&(myArray[0]), nullptr);

LinkNode *secNode = LinkNode *firstNode = new LinkNode(&(myArray[1]), nullptr);

c. (3pt) Please show code to link the above three LinkNodes into a linked list with myHead being the header, firstNode being the first node, and secNode being the tail of the list.

myHeader->next = firstNode;
firstNode->next = secNode;

d. (3pt) Please show how you would release the memory allocated for the linked list without affecting the memory of MyArray.

delete myHeader
delete firstNode
delete secNode

Question 2. (18pt) Now, for a more straightforward case of linked list of integers,

```
struct Node {
    int data;
    Node* next;
};
```

I have a dummy header for my linked list,

```
Node *header = new Node(-1, nullptr);  // dummy header
…
// assume the list is initialized with positive integers that are greater than 0
// assume all nodes in the list are dynamically allocated
```

a. (4pt) Now, I want to print only the odd data content of the list with this function call:

```
PrintOdd(header);  // remember header points to a dummy header
```

Please show, with clear and detailed parameter declaration, the implementation of `PrintOdd()` function. For example, if I have a list with 2 3 6 7 8, only values of 3 and 7 will be printed. Note, the data-field of the dummy header should be ignored.

```
bool isOdd(int value) {
    return (value % 2 != 0);
}


void PrintOdd(string msg, Node *head) {
    cout << msg << ": ";
    Node *current = head->next; // skip dummy header
    while (current != nullptr) {
        if (isOdd(current->data)) {
            cout << current->data << " ";
        }
        current = current->next;
    }
    cout << endl;
}
```

b. (6pt) The following function removes the first node with odd data from the list without releasing any memory. If the list has no node with odd data, then a nullptr is returned.

```
Node *n = RemoveOdd(header);  // remember header points to a dummy header
```

Please show, with clear and detailed parameter declaration, how you would implement the `RemoveOdd()` function. . For example, if I have a list with 2 3 6 7 8, this function will remove and return the node with the 3, where the list pointed to `header` will become 2 6 7 8.

```
Node* RemoveOdd(Node *head) {
    Node *current = head; // dummy header
    Node *toRemove = nullptr;
    while ((current->next != nullptr) && (toRemove == nullptr)) {
        if (isOdd(current->next->data)) {
            toRemove = current->next;
            current->next = toRemove->next;
            toRemove->next = nullptr;
        } else {
            // only execute if needs to advance
            // if only one node left and it is the data to remove,
            // trying to advance will access nullptr
            current = current->next;
        }
    }
    return toRemove; // nullptr if not found
}
```

c. (8pt) Please show how you would call the RemoveOdd() function to remove all nodes with odd numbers in the header list and use the removed nodes to create a new list to be pointed to by the following oHeader. The numbers in the new list must appear in the same order as they are removed. For example, if the original list is 2 3 6 7 8, after your code, the original list will become: 2 6 8, and the list pointed to by oHeader would be 3 7. Notice that the number 3 is removed before 7 and thus will appear in the oHeader list before the 7.
WARNING, ***only*** solutions based on calling the RemoveOdd() function will receive credits.

```
// assume a properly initialized list pointed to by: header (with dummy header)
Node *oHeader = new Node(-1, nullptr);  // dummy header for the odd-number list
// please show how to create the list as specified above
```

```
Node *oHeader = init_dummy_header();

bool done = false;
Node *tail = oHead;
Node *oddNode = RemoveOdd(header);
while (oddNode != nullptr) {
  tail->next = oddNode;
  oddNode->next = nullptr;
  tail = tail->next;
  oddNode = RemoveOdd(header);
}
```