Exercise 1.

Draw a diagram showing the structure of the list and the position of the pointers \( p1 \) and \( p2 \) after executing each fragment of code in parts a-d. Some of the fragments may produce memory leaks or errors. Assume class node is defined as follows:

```cpp
class node
{
    public:
        node( int x, node* ptr )
        {
            data = x;
            next = ptr;
        }
        int data;
        node* next;
};
```

a) \( \text{node } *p1; \)
\( p1 = \text{new node}( 10, \text{NULL} ); \)
\( p1 = \text{new node}( 20, p1 ); \)
\( p1 = \text{new node}( 30, p1 ); \)

b) \( \text{node } *p1, *p2; \)
\( p1 = p2 = \text{new node}( 10, \text{NULL} ); \)
\( p2 = p2->\text{next} = \text{new node}( 20, \text{NULL} ); \)
\( p2 = p2->\text{next} = \text{new node}( 30, \text{NULL} ); \)

c) \( \text{node } *p1, *p2; \)
\( p1 = \text{new node}( 10, \text{NULL} ); \)
\( p2 = \text{new node}( 20, p1 ); \)
\( p2->\text{next->next} = \text{new node}( 30, p1 ); \)
\( p2->\text{next} = p2; \)
\( p2 = p2->\text{next->next}; \)

d) \( \text{node } *p1, *p2; \)
\( p1 = \text{new node}( 10, \text{NULL} ); \)
\( p2 = \text{new node}( 20, p1 ); \)
\( p1->\text{next} = p2; \)
\( p2 = p2->\text{next->next->next}; \)
Exercise 2.

Assume a list exists with the following content:

```
ptr 10 20 30 40 50 NULL
```

a) Show the output produced by the function call: `mystery(ptr, 50);`
b) Show the output produced by the function call: `mystery(ptr, 10);`
c) Describe the operation performed by the mystery function in one or two sentences.

class node
{
    public:
        node(int x, node* ptr) { data = x; next = ptr; }
        int data;
        node* next;
};

void mystery(node*& head, int x)
{
    node* p2 = NULL;
    node* p1 = head;
    while ( p1 != NULL && x != p1->data )
    {
        p2 = p1;
        p1 = p1->next;
    }
    if ( p1 != NULL )
    {
        if ( p1 == head )
        {
            head = head->next;
        }
        else
        {
            p2->next = p1->next;
        }
        delete p1;
    }
}
Exercise 3.

The file lab3.zip contains a nearly complete implementation of a simple singly linked list class. Two member functions are not implemented: print and rprint. Your tasks are to:

(a) write the member function:

    ```cpp
    void print(ostream&) const;
    ```

    that print the contents of a singly linked list.

(b) write the member function:

    ```cpp
    void rprint(ostream&) const;
    ```

    that print the contents of a singly linked list in reverse order. Hint - add a private member function that uses recursion to print the list in reverse order.