#1)

```cpp
#include <iostream>

using namespace std;

struct date {
    int month;
    int day;
    int year;
};

struct records {
    string name;
    int numStonks;
    double priceStonks;
    date dt;
};

int main()
{
    date today;
    records rec1;
    records rec2;
    char c = '-';

    cout << "Enter name 1: ";
    cin >> rec1.name;
    cout << "Enter name 2: ";
    cin >> rec2.name;

    cout << "Enter today's date (Month, Day, Year): ";
    cin >> today.month >> today.day >> today.year;

    rec1.dt.day = rec2.dt.day = today.day - 1;
    rec1.dt.month = rec2.dt.month = today.month;
    rec1.dt.year = rec2.dt.year = today.year;


    rec1.numStonks = 300;
    rec2.numStonks = 450;

    rec1.priceStonks = rec2.priceStonks = 27.15;

    cout << "Record 1" << endl;
    cout << "Name: " << rec1.name << endl;

    cout << "Number of stocks: " << rec1.numStonks << endl;

    cout << "Total Price: " << rec1.numStonks * rec1.priceStonks << endl;
    cout << "Date: " << rec1.dt.month << c << rec1.dt.day << c << rec1.dt.year <<
endl << endl;

    // 2

    cout << "Record 2" << endl;
    cout << "Name: " << rec2.name << endl;
```

```
    cout << "Number of stocks: " << rec2.numStonks << endl;

    cout << "Total Price :" << rec2.numStonks * rec2.priceStonks << endl;
    cout << "Date: " << rec2.dt.month << c << rec2.dt.day << c << rec2.dt.year;


    return 0;
}
```

#2)

```
#include <iostream>

    using namespace std;
    struct Node
    {
        string name;
        Node* next;
    };

    Node* p1;
    Node* p2;

    int main()
    {
        //p1 points to node p
        //p2 -> next != NULL
        p1 = p1->next;

        NodePtr tempHold = p2->next->next; //preserves the list after the deleted
node
        delete p2->next;
        p2->next = tempHold;

        return 0;
    }
```

#3)

```
#include <iostream>
#include <string>
using namespace std;

struct Node
{
    string name;
    Node* link;
};
typedef Node* NodePtr;

int main()
{
    NodePtr listPtr, tempPtr;
    listPtr = new Node;
    listPtr->name = "Emily";
    tempPtr = new Node;
    tempPtr->name = "James";
    listPtr->link = tempPtr;
```

```
    tempPtr->link = new Node;
    tempPtr = tempPtr->link;
    tempPtr->name = "Joules";
    tempPtr->link = NULL;
    //2a
    NodePtr holder = listPtr;
    while (listPtr != NULL) {
        cout << listPtr->name << " ";
        listPtr = listPtr->link;
    }
    listPtr = holder;
    cout << endl;
    //2b
    NodePtr newPtr;
    newPtr = new Node;
    newPtr->name = "Joshua";
    NodePtr old = listPtr->link->link;
    newPtr->link = old;
    listPtr->link->link = newPtr;

    while (listPtr != NULL) {
        cout << listPtr->name << " ";
        listPtr = listPtr->link;
    }
    cout << endl;
    listPtr = holder;
    //2c
    delete listPtr->link->link->link;
    listPtr->link->link->link = NULL;
    while (listPtr != NULL) {
        cout << listPtr->name << " "; //deletes the link to node "Joules" and then
outputs new list
        listPtr = listPtr->link;
    }
    cout << endl << endl;
    listPtr = holder;
    //2d
    while (listPtr->link != NULL) {
        NodePtr tempHold = listPtr->link->link; //Deletes the second link
        delete listPtr->link;
        listPtr->link = tempHold;

        while (listPtr != NULL) {
            cout << listPtr->name << " "; //outputs the new list with the deleted
node
            listPtr = listPtr->link;
        }

        cout << endl;
        listPtr = holder;
    }


    delete listPtr;
    return 0;
}
```