```cpp
#include <iostream>
#include <queue>
#include <set>
#include <stack>
using namespace std;

queue<int> reverseQueue(queue<int> input) {
    queue<int> result;
    stack<int> temp;

    while(!input.empty()) {
        temp.push(input.front());
        input.pop();
    }
    while(!temp.empty()) {
        result.push(temp.top());
        temp.pop();
    }
    return result;
}

set<int> makeUnion(set<int> s1, set<int> s2) {
    set<int> result;

    set<int>::iterator iter = s1.begin();
    while(iter != s1.end()) {
        result.insert(*iter);
        iter++;
    }
    set<int>::iterator itera = s2.begin();
    while(itera != s2.end()) {
        result.insert(*itera);
        itera++;
    }
    return result;
}

stack<int> removeNegative(stack<int> input) {
    stack<int> result, temp;

    while(!input.empty()) {
        if(input.top() >= 0)
            temp.push(input.top());
        input.pop();
    }
    while(!temp.empty()) {
        result.push(temp.top());
        temp.pop();
    }

    return result;
}


int main()
{
    std::cout << "Hello, World!" << std::endl;

    cout << endl << "Problem 0" << endl;
    queue<int> q;
    q.push(1); //gets food first
    q.push(2);
    q.push(3); //gets food last. can't get here without getting to the preceding ones

    queue<int> reversedQ = reverseQueue(q);
    while(!reversedQ.empty()) {
        cout << reversedQ.front() << endl;
        reversedQ.pop();
    }

    cout << endl << "Problem 1" << endl;
    set<int> s1, s2;
    s1.insert(1);
    s1.insert(2);
    s2.insert(1);
```

```cpp
    s2.insert(3);
    s2.insert(4);

    set<int> unionized = makeUnion(s1, s2);
    set<int>::iterator iter = unionized.begin();
    while(iter != unionized.end()) {
        cout << *iter << endl;
        iter++;
    }

    cout << endl << "Problem 2" << endl;

    stack<int> st;
    st.push(1);
    st.push(-1);
    st.push(-11);
    st.push(10);

    stack<int> posS = removeNegative(st);
    while(!posS.empty()) {
        cout << posS.top() << endl;
        posS.pop();
    }

    return 0;
}

```