main( )
{
    int n;
    static const int maxn = 100;
    int P[maxn+1];
    bool D[maxn+1];

    cout << "n from 1 to " << maxn << "? ";
    cin >> n ;
    cout << "Permutation of [1.." << n << "]? ";
    for( int i = 1; i <= n; i++ )
    {
        cin >> P[i];
        D[i] = false;
    }

    //INVs: Values in P[1..n] give a permutation mapping i to P[i].
    //      For i=1 to n, D[i]==true if and only if i has
    //      been output as a number in a cycle of the perm.

    int cycle_start = 1;
    int cycle_count = 0;
    while ( cycle_start <= n )
    {
        if( ! D[cycle_start] )
        {
            D[cycle_start] = true;
            cout << "(" << cycle_start;
            // For i=1 to n, D[i]==true if and only if i has
            // been output as a number in a cycle of the perm.
            int cycle_next = P[cycle_start];
            while( cycle_next != cycle_start )
            {
                D[cycle_next] = true;
                cout << " " << cycle_next;
                // For i=1 to n, D[i]==true if and only if i has
                // been output as a number in a cycle of the perm.
                cycle_next = P[cycle_next];
            }
            cout << ") ";
            cycle_count++;
        }
        cycle_start++;
    }
    cout << endl;
    // For i=1 to n, D[i]==true.
    // Every i from 1 to n been output
    // as a number in a cycle of the permutation.
    cout << "Your permutation has the above " << cycle_count << " cycles." << endl;
}

//Homework Due 2/14
// Draw the arrays P and D, and variables cycle_start and cycle_next.
// Trace this programs execution on the following two input cases:
// Case 1:
//   6
// Case 2:
//   6
// 1 2 4 5 3 6