```
javascript:

/* Get the frame that contains the edit form. */

var frmContent =
      document.getElementsByName(
                'frmContent')[0];

var frmContentWin =
      frmContent.contentWindow;

var frmContentDoc =
      frmContentWin.document;

/* Get the table displayed in the edit frame. */

var tblData =
      frmContentDoc.getElementById('tblData');

/* Inject/update the textarea which will hold
   the updated values. */

var hackTrId = "updValsHackTr";
var hackTextareaId = "updValsHackTextarea";

if (frmContentDoc.getElementById(hackTrId)
      != null) {
    var hackTr =
            frmContentDoc.getElementById(
                      hackTrId);
    hackTr.parentNode.removeChild(hackTr);
}

/* Create the hack UI. */

var tblDataTbody =
      tblData.getElementsByTagName('tbody')[0];

var hackTr = document.createElement('tr');
hackTr.setAttribute("id", hackTrId);
tblDataTbody.insertBefore(hackTr,
      tblDataTbody.getElementsByTagName(
                'tr')[0]);
```

```
var hackTd = document.createElement('td');
hackTd.setAttribute("colspan", '7');
hackTr.appendChild(hackTd);

var hackTextarea =
      document.createElement('textarea');
hackTextarea.setAttribute(
      "id", hackTextareaId);
hackTd.appendChild(hackTextarea);

var hackButton =
      document.createElement('input');
hackButton.setAttribute("type", "button");
hackButton.setAttribute("value", "Update Values");
hackButton.setAttribute("onclick",
      "updateValues(parseNewVals())");
hackTd.appendChild(hackButton);

/**
 * Parse the new values contained in the hack
 * textarea.
 *
 * Assuming that the data inputted comes directly
 * in the same format as exported from AY,
 * the column headers are as folows:
 * Label
 * Choice Value
 * Choice Order
 * Header: "Yes" or blank
 * Related Value
 * Inactive Date: m/d/yy
 *
 * @return Array of new values
 */
frmContentDoc.parseNewVals = function() {
    var vals = hackTextarea.value.split(/[\r\n]/);
    for (i = 0; i < vals.length; i++)
      vals[i] = vals[i].split(/\t/);

    return vals;
};
```

```
/**
 * Update the choice group values based on the
 * data dumped into the hack's textarea, assuming
 * the data is in TSV format, following the same
 * column order as in the file exported from AY.
 *
 * @param newVals Array of new values
 */
frmContentDoc.updateValues = function(newVals) {

    /* Find the value headers row.

     Assume the header row can be identified
     by bgcolor="#CCCCCC". */

    var valHeaderTr = null;

    var tblDataTrs = tblDataTbody.getElementsByTagName('tr');
    for (i = 0; i < tblDataTrs.length & valHeaderTr == null; i++) {
      var tr = tblDataTrs[i];
      if (tr.getAttribute("bgcolor") == "#CCCCCC")
           valHeaderTr = tr;
    }

    /* Find the first value row. */

    var lineRow = valHeaderTr.nextSibling.nextSibling;
    var spacerRow = lineRow.nextSibling.nextSibling;
    var firstValRow = spacerRow.nextSibling.nextSibling;

    /* Adjust the form to contain the correct number of rows. */

    var trDecorative = frmContentDoc.getElementById("trDecorative");

    var lastValRow = getPreviousTr(trDecorative);

    var lastValTds = lastValRow.getElementsByTagName("td");

    var difference = eval(newVals.length + "-" +
lastValTds[0].innerHTML);

    if (difference > 0) {
      for (i = 0; i < difference; i++)
```

```
            frmContentWin.addRow();
    }
    else if (difference < 0) {
      for (i = 0; i > difference; i--) {
            lastValTds = lastValRow.getElementsByTagName("td");
            var lastValDelCheckbox =
lastValTds[1].getElementsByTagName("input")[0];
            if (lastValDelCheckbox.checked == false)
                  lastValDelCheckbox.click();

            lastValRow = getPreviousTr(lastValRow);
      }
    }

    /* Iterate through the new values and update the choice group. */

    var prevValRow = null;
    var currValRow = firstValRow;
    for (i = 0; i < newVals.length; i++) {
      var val = newVals[i];
      var valTds = currValRow.getElementsByTagName("td");

      /* To delete */

      var valDelCheckbox =
valTds[1].getElementsByTagName("input")[0];
      if (valDelCheckbox.checked)
            valDelCheckbox.click();

      /* Label */

      var valLabelTd = valTds[2];
      valLabelTd.getElementsByTagName("input")[0].value = val[0];

      /* Code */

      var valCodeTd = valTds[3];
      valCodeTd.getElementsByTagName("input")[0].value = val[1];

      /* Order */

      var valOrderTd = valTds[4];
      valOrderTd.getElementsByTagName("input")[0].value = val[2];
```

```
      /* Header */

      var valHeaderTd = valTds[5];
      var valHeaderCheckbox =
valHeaderTd.getElementsByTagName("input")[0];

      if (valHeaderCheckbox.checked == false && val[3] == "Yes")
            valHeaderCheckbox.click();
      else if (valHeaderCheckbox.checked == true && val[3] == "")
            valHeaderCheckbox.click();

      /* Related Choice / Inactive Date */

      var valRelChoiceTd = valTds[6];
      valRelChoiceTd.getElementsByTagName("input")[0].value =
val[4];
      valRelChoiceTd.getElementsByTagName("input")[1].value =
val[5];

      /* Setup for next iteration. */

      prevValRow = currValRow;
      currValRow = getNextTr(currValRow);
    }

    alert("Done!");
};

/**
 * Thanks to this page for help:
 *
http://www.w3.org/TR/REC-DOM-Level-1/level-one-core.html#ID-745549614
 */
function getPreviousTr(elem) {
    var prevTr = null;

    while (elem.previousSibling != null
            && prevTr == null) {
      elem = elem.previousSibling;
      if (elem.nodeName == "TR")
            prevTr = elem;
    }
```

```
    return prevTr;
};

function getNextTr(elem) {
    var nextTr = null;

    while (elem.nextSibling != null
            && nextTr == null) {
      elem = elem.nextSibling;
      if (elem.nodeName == "TR")
            nextTr = elem;
    }

    return nextTr;
};
```