

```

0001 *
0002 *****
0003 * Copyright (c) 2004 Schellenbach & Associates, Inc. dba AccuSoft *
0004 * Enterprises as an unpublished work. Permission is hereby granted, free *
0005 * of charge, to any person obtaining a copy of this software, to use the *
0006 * software without restriction, including without limitation the rights *
0007 * to use, copy, modify, merge, publish or distribute the software, and *
0008 * to permit persons to whom the software is furnished to do so, subject *
0009 * to the following conditions: This copyright notice and permission *
0010 * notice shall be included in all copies or substantial portions of the *
0011 * software. THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY *
0012 * KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES *
0013 * OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND *
0014 * NONINFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS BE LIABLE FOR *
0015 * ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF *
0016 * CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION *
0017 * WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. *
0018 *****
0019 *****
0020 *
0021 * USER INTERFACE EXAMPLE PROGRAM 1
0022 *
0023 *****
0024 *****
0025 *
0026 * This example illustrates a very simple character-based user interface.
0027 * This example does not use any visual highlighting, or respond to
0028 * function or cursor keys.
0029 *
0030 *****
0031 *
0032 * The CUST.REC INCLUDE item declares the CUST.REC array, which is used to
0033 * store a working copy of the current data record. It also defines the
0034 * record layout by equating field names to array positions in the CUST.REC
0035 * array.
0036 *
0037 $INCLUDE UI EX. CUST. REC
0038 *
0039 *****
0040 *
0041 * EQUates for control characters and delimiters and other constants
0042 *
0043 EQU VM TO CHAR(253)
0044 EQU BEL TO CHAR(7)
0045 EQU FALSE TO 0
0046 EQU TRUE TO 1
0047 *
0048 *****
0049 *
0050 * Open our files...
0051 *
0052 OPEN 'CUST.SAMPLE' TO FN.CUST ELSE PRINT 'NO CUST.SAMPLE FILE'; STOP
0053 OPEN 'DICT CUST.SAMPLE' TO FN.DICT.CUST ELSE PRINT 'NO DICT CUST.SAMPLE FILE'; STOP
0054 OPEN 'CUST.SAMPLE.XREF' TO FN.CUST.XREF ELSE PRINT 'NO CUST.SAMPLE.XREF'; STOP
0055 *
0056 *****
0057 *
0058 * Define the data field control structures. NUMFLDS is the number of
0059 * fields. The CONTROL array defines the field control elements such as
0060 * field label, label position, field position and size, and field prompt.
0061 * The IDATA array stores the current field values, and is initialized from
0062 * the CUST.REC array when a data record is read from the file. When the
0063 * record is updated, values are copied from the IDATA array to the
0064 * CUST.REC array and the CUST.REC array is passed to the CUST.UPDATE
0065 * subroutine to update the data and index files (a separate subroutine is
0066 * used to update the file since the update process may handle other
0067 * functions like updating indexes).
0068 *
0069 NUMFLDS = 12

```

```

0070 DIM CONTROL(12)
0071 DIM IDATA(12)
0072 *
0073 * Define field control elements
0074 * value 1: field label text
0075 * value 2: field label column
0076 * value 3: field label row
0077 * value 4: field label width (0 for actual width, no padding)
0078 * value 5: field data column
0079 * value 6: field data row
0080 * value 7: field data width
0081 * value 8: field data height
0082 * value 9: field prompt message
0083 CONTROL(1) = 'Customer IDy7y2y20y28y2y30y1yEnter the ID, or name to search for, or N f
0084 CONTROL(2) = 'Contacty7y4y20y28y4y30y1yEnter the contact name'
0085 CONTROL(3) = 'Company namey7y5y20y28y5y30y1yEnter the company name'
0086 CONTROL(4) = 'Address line 1y7y6y20y28y6y30y1yEnter address line 1'
0087 CONTROL(5) = 'Address line 2y7y7y20y28y7y30y1yEnter address line 2'
0088 CONTROL(6) = 'Cityy7y8y20y28y8y25y1yEnter the city'
0089 CONTROL(7) = 'State or provincey7y9y20y28y9y15y1yEnter the state or province abbreviat
0090 CONTROL(8) = 'Zip/postal codey7y10y20y28y10y10y1yEnter the zip or postal code'
0091 CONTROL(9) = 'Countryy7y11y20y28y11y18y1yEnter the country'
0092 CONTROL(10) = 'Phoney7y12y20y28y12y15y1yEnter the phone number'
0093 CONTROL(11) = 'Faxy7y13y20y28y13y15y1yEnter the fax number'
0094 CONTROL(12) = 'Notesy7y14y20y28y14y30y1yEnter any notes about this customer'
0095 *
0096 *****
0097 *
0098 * Define some screen control strings for prompts & errors
0099 *
0100 PROMPT ''
0101 CLR = @(-1) ; * Clear entire screen
0102 CEOL = @(-4) ; * Clear to end of line
0103 PL = @(5,22):CEOL ; * Prompt line
0104 EL = @(5,23):CEOL ; * Error line
0105 *
0106 *****
0107 *
0108 * This program processes one customer record at a time, and is organized
0109 * using three nested loops. The outermost loop (the RECORD loop) is
0110 * executed once for each record accessed. The loop repeats until the EXIT
0111 * control variable is set to TRUE.
0112 *
0113 EXIT = FALSE
0114 LOOP UNTIL EXIT DO
0115 *
0116 *****
0117 *
0118 * Before prompting for the customer ID, reset the internal data array
0119 * (IDATA) and CUST.ID variables, then clear the screen and display the
0120 * heading and field labels.
0121 *
0122 GOSUB RESTART
0123 GOSUB DSPSCRN
0124 *
0125 *****
0126 *
0127 * The middle loop is the ACTION loop. It is executed for the current
0128 * record until the user performs an action that terminates processing of
0129 * that record, such as exiting, cancelling, saving or deleting the
0130 * record. The field number to begin prompting (NXTFLD) is initialized to
0131 * 1, causing the ID field to be prompted first. The loop immediately
0132 * enters the PROMPT loop, followed by the field modification prompt. The
0133 * loop repeats until the DONE control variable is set to TRUE.
0134 *
0135 NXTFLD = 1
0136 DONE = FALSE
0137 LOOP
0138 *

```

```
0139 *****
0140 *
0141 * The inner loop is the PROMPT loop. It is executed for each prompt
0142 * field as specified by the NXTFLD variable. The prompt loop simply
0143 * calls the local INPUT.FIELD subroutine which performs field input,
0144 * data validation and keyboard command decoding. The FIELD loop repeats
0145 * until the field number is greater than the number of fields, or until
0146 * the DONE control variable is set to TRUE. The DONE control variable
0147 * may be set to TRUE in the INPUT.FIELD subroutine, if the user enters a
0148 * NULL to for the item-ID.
0149 *
0150 LOOP
0151   CURFLD = NXTFLD
0152   UNTIL DONE OR CURFLD > NUMFLDS DO
0153     *
0154     *****
0155     *
0156     * Prompt for the next field. Update the NXTFLD variable with the field
0157     * number to prompt next.
0158     *
0159     GOSUB INPUT.FIELD
0160     REPEAT ; * end of PROMPT loop
0161     *
0162     *****
0163     *
0164     * If the FIELD loop exited with the DONE control variable set to TRUE,
0165     * bypass the modification prompt because no action is required.
0166     * Otherwise, prompt for which field to modify, or other actions such as
0167     * save or delete.
0168     *
0169     IF NOT(DONE) THEN
0170       *
0171       NXTFLD = NUMFLDS + 1 ; * assume we need to reprompt for action
0172       *
0173       PRINT PL: 'Enter field number to modify or FI to save, DE to delete, EX to exit: ':
0174       INPUT ANS:
0175       PRINT EL:
0176       *
0177       *****
0178       *
0179       * Decode the response
0180       *
0181       ANS = OCONV(ANS, 'MCU')
0182       BEGIN CASE
0183         CASE NUM(ANS) AND ANS >= 1 AND ANS <= NUMFLDS; NXTFLD = ANS
0184         CASE ANS EQ 'FI'; GOSUB SAVE.RECORD
0185         CASE ANS EQ 'DE'; GOSUB DELETE.RECORD
0186         CASE ANS EQ 'EX' OR ANS EQ ''; GOSUB CHECK.EXIT
0187       END CASE
0188       *
0189       END
0190       *
0191     UNTIL DONE DO REPEAT ; * end of ACTION loop
0192     *
0193   REPEAT ; * end of RECORD loop
0194   *
0195   *****
0196   *
0197   * All done - clear the screen and exit!
0198   PRINT CLR:
0199   STOP
0200   *
0201   *
0202   *****
0203   *****
0204   * LOCAL SUBROUTINES
0205   *****
0206   *****
0207   *
```

```

0208 *
0209 *****
0210 *
0211 * The INPUT.FIELD subroutine is the main prompting routine. This routine
0212 * displays the prompt string (from the CONTROL array), and enters a loop,
0213 * prompting for a specified field (CURFLD) and setting the next field
0214 * variable (NXTFLD) appropriately. The loop repeats until the NXTFLD
0215 * (initially set to CURFLD) changes. This causes the field prompt to be
0216 * repeated in case invalid data is entered (illegal customer ID, etc.) If
0217 * a NULL is entered for the ID field, and there is no current record, the
0218 * ACTION loop control variable, DONE, and the RECORD loop control
0219 * variable, EXIT, are set to TRUE, and this routine exits.
0220 *
0221 INPUT.FIELD:
0222 *
0223 *****
0224 *
0225 * Display the prompt message
0226 *
0227 PRINT PL: CONTROL(CURFLD)<1, 9>:
0228 *
0229 *****
0230 *
0231 * Initialize current value, next field
0232 *
0233 PREVAL = IDATA(CURFLD) ; * Save previous field value to detect change in value
0234 *
0235 *****
0236 *
0237 * Prompt for this field until NXTFLD variable is updated
0238 *
0239 LOOP
0240 *
0241 *****
0242 *
0243 * Assume next field number is next sequential field
0244 *
0245 NXTFLD = CURFLD + 1
0246 *
0247 * Prompt for input
0248 *
0249 PRINT @(CONTROL(CURFLD)<1, 5>, CONTROL(CURFLD)<1, 6>):
0250 INPUT IDATA(CURFLD):
0251 *
0252 * Check for NULL
0253 *
0254 K = LEN(IDATA(CURFLD))
0255 IF K = 0 THEN
0256 *
0257 * No change if NULL entered
0258 *
0259 IDATA(CURFLD) = PREVAL
0260 END ELSE
0261 *
0262 * Erase old data
0263 *
0264 IF K < CONTROL(CURFLD)<1, 7> THEN
0265 PRINT @(K+CONTROL(CURFLD)<1, 5>, CONTROL(CURFLD)<1, 6>): SPACE(CONTROL(CURFLD)<1, 7> - K
0266 END
0267 END
0268 *
0269 * Clear the error line
0270 *
0271 PRINT EL:
0272 *
0273 *****
0274 *
0275 * Check for any special values (like 'END' or 'EXIT')
0276 *

```

```

0277 IF OCONV(IDATA(CURFLD),'MCU') EQ 'END' THEN
0278 IDATA(CURFLD) = PREVAL ;* Restore previous value
0279 GOSUB CHECK.ABANDON ;* Ensure OK to loose changes
0280 IF OK THEN
0281 *
0282 *****
0283 *
0284 * No changes, or user OKs abandoning them, so we are outa here!
0285 *
0286 DONE = TRUE
0287 EXIT = TRUE
0288 RETURN
0289 *
0290 END ELSE
0291 *
0292 *****
0293 *
0294 * User does not want to abandon changes, so reprompt
0295 *
0296 XLINE = CURFLD ;* Set the field number to refresh
0297 GOSUB DSPLINE ;* Refresh previous value
0298 NXTFLD = CURFLD ;* Reprompt
0299 *
0300 END
0301 END
0302 IF IDATA(CURFLD) EQ SPACE(LEN(IDATA(CURFLD))) THEN IDATA(CURFLD) = ''
0303 *
0304 *****
0305 *
0306 * Perform field data validation if next field number changed
0307 *
0308 IF NXTFLD NE CURFLD THEN
0309 BEGIN CASE
0310 *
0311 CASE CURFLD EQ 1
0312 *
0313 *****
0314 *
0315 * Validate the ID field. If NULL, quit. If changed, read new record.
0316 *
0317 IF CUST.ID EQ '' AND IDATA(CURFLD) EQ '' THEN
0318 DONE = TRUE
0319 EXIT = TRUE
0320 RETURN
0321 END
0322 *
0323 IF IDATA(CURFLD) NE PREVAL THEN
0324 ID = IDATA(CURFLD)
0325 GOSUB CHECK.ID ;* Check the newly entered ID handling index lookups
0326 IF OK THEN
0327 *
0328 *****
0329 *
0330 * New ID (or result of index lookup) is good!
0331 *
0332 IDATA(CURFLD) = ID ;* Update the current field value in case of index lookup
0333 *
0334 END ELSE
0335 *
0336 *****
0337 *
0338 * New ID is invalid
0339 *
0340 IDATA(CURFLD) = PREVAL ;* Restore previous value
0341 XLINE = CURFLD ;* Set the field number to refresh
0342 GOSUB DSPLINE ;* Refresh previous value
0343 NXTFLD = CURFLD ;* Reprompt
0344 *
0345 END

```

```
0346     END
0347     *
0348     END CASE
0349     END
0350     *
0351     WHILE CURFLD EQ NXTFLD DO REPEAT
0352     *
0353     RETURN
0354     *
0355     *
0356     *****
0357     *
0358     * The CHECK.EXIT subroutine checks if any field data has changed, and
0359     * prompts if the user wants to abandon changes. If no changes, or the user
0360     * decides to abandon the changes, the DONE and EXIT loop control variables
0361     * are set to TRUE, causing all three loops to terminate, and the program
0362     * itself to exit.
0363     *
0364     CHECK.EXIT: *
0365     *
0366     GOSUB CHECK.ABANDON
0367     IF OK THEN
0368         DONE = TRUE
0369         EXIT = TRUE
0370     END
0371     RETURN
0372     *
0373     *
0374     *****
0375     *
0376     * The CHECK.ID subroutine validates a newly entered item-ID. If the
0377     * current record has unsaved changes, the user is prompted to abandon the
0378     * changes. If no changes, or the user abandons the changes, and the new ID
0379     * is not NULL, an attempt is made to read a record using the new ID. If
0380     * the read is not successful, the ID is assumed to be a search string, and
0381     * the search subroutine is called to select an ID based on the search
0382     * string. If a valid ID is returned (or if one was initially entered), the
0383     * new record data is displayed. If the new ID is null, or if the search
0384     * routine did not return a valid ID, a warning message is displayed and
0385     * the OK indicator variable is set to FALSE. Otherwise it is set to TRUE.
0386     *
0387     CHECK.ID: *
0388     *
0389     *****
0390     *
0391     * Make sure we don't have any unsaved data before changing the ID
0392     *
0393     GOSUB CHECK.ABANDON
0394     IF NOT(OK) THEN RETURN ;* Reprompt for the ID
0395     IF ID EQ '' THEN
0396         OK = FALSE ;* NULL is not a valid ID!
0397     END ELSE
0398     *
0399     *****
0400     *
0401     * Check if user wants new ID
0402     *
0403     IF ID EQ 'N' OR ID EQ 'n' THEN
0404         * Get next sequential ID
0405         READVU ID FROM FN.DICT.CUST, 'NEXT', 2 THEN
0406             WRITEV ID + 1 ON FN.DICT.CUST, 'NEXT', 2
0407             GOSUB RESTART
0408             IDATA(1) = ID
0409         END ELSE
0410             PRINT EL:'Next item counter record not found!':BEL:
0411             INPUT ANS:
0412             PRINT EL:
0413             OK = FALSE
0414             RETURN
```

```
0415     END
0416 END ELSE
0417 *
0418 *****
0419 *
0420 * Try to read the customer record from the entered ID
0421 *
0422 GOSUB READ.RECORD
0423 IF NOT(OK) THEN
0424 *
0425 *****
0426 *
0427 * The attempt to read a record failed - assume the ID is a search string
0428 *
0429 CALL UI EX. GET. CUST. IDX(XID, ID, FN. CUST. XREF, FN. CUST)
0430 GOSUB DSPSCRN ; * Refresh the screen after index lookup
0431 *
0432 *****
0433 *
0434 * If the user did not select an item in the search routine, reprompt
0435 *
0436 IF XID EQ '' THEN RETURN
0437 *
0438 *****
0439 *
0440 * Try to read the customer record from the selected ID
0441 *
0442 ID = XID
0443 GOSUB READ.RECORD
0444 *
0445     END
0446     END
0447 END
0448 *
0449 *****
0450 *
0451 * If success, display the new record, otherwise show warning message
0452 *
0453 IF OK THEN
0454     GOSUB DSPDATA ; * Display new record
0455 END ELSE
0456     PRINT EL: 'Please enter a valid customer ID!': BEL:
0457 END
0458 RETURN
0459 *
0460 *
0461 *****
0462 *
0463 * The READ.RECORD subroutine reads a new customer record from the file and
0464 * initializes the internal field data array (IDATA) from the record array
0465 * (CUST.REC). If the record does not exist, the routine returns with the
0466 * OK indicator variable set to FALSE. Otherwise OK is set to TRUE, and the
0467 * CUST.ID variable is set to the new ID.
0468 *
0469 READ.RECORD: *
0470 *
0471 MATREAD CUST.REC FROM FN. CUST, ID THEN
0472     CUST.ID = ID
0473     IDATA(1) = CUST.ID
0474     IDATA(2) = CUST.CONTACT
0475     IDATA(3) = CUST.NAME
0476     IDATA(4) = CUST.ADDRESS1
0477     IDATA(5) = CUST.ADDRESS2
0478     IDATA(6) = CUST.CITY
0479     IDATA(7) = CUST.ST
0480     IDATA(8) = CUST.ZIP
0481     IDATA(9) = CUST.COUNTRY
0482     IDATA(10) = CUST.PHONE
0483     IDATA(11) = CUST.FAX
```

```
0484 I DATA(12) = CUST.HISTORY
0485 OK = TRUE ; * Set the SUCCESS indicator
0486 END ELSE
0487 OK = FALSE ; * Set the FAILURE indicator
0488 END
0489 RETURN
0490 *
0491 *
0492 *****
0493 *
0494 * The DELETE.RECORD subroutine confirms that the user intends to delete
0495 * the current record. If the action is confirmed, the CUST.DELETE
0496 * subroutine is called to perform the deletion. A separate subroutine is
0497 * used to handle updating indexes, etc.
0498 *
0499 DELETE.RECORD: *
0500 *
0501 IF CUST.ID NE '' THEN
0502 PRINT EL:'Are you sure you want to delete this customer? ':
0503 INPUT ANS:
0504 IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN
0505 * Deletion has been confirmed - do the delete
0506 OK = TRUE ; * Set the SUCCESS indicator
0507 CALL UI EX.CUST.DELETE(CUST.ID, FN.CUST, FN.CUST.XREF)
0508 DONE = TRUE ; * proceed to next record
0509 END ELSE
0510 OK = FALSE ; * Set the FAILURE indicator
0511 END
0512 END
0513 RETURN
0514 *
0515 *
0516 *****
0517 *
0518 * The SAVE.RECORD subroutine copies internal field data from the I DATA
0519 * array to the customer record array (CUST.REC). The CUST.UPDATE
0520 * subroutine is called to perform the update. A separate subroutine is
0521 * used to handle updating indexes, etc.
0522 *
0523 SAVE.RECORD: *
0524 *
0525 IF I DATA(1) NE '' THEN
0526 * Copy data from the internal field data array (I DATA) to the CUST.REC array
0527 CUST.ID = I DATA(1)
0528 CUST.CONTACT = I DATA(2)
0529 CUST.NAME = I DATA(3)
0530 CUST.ADDRESS1 = I DATA(4)
0531 CUST.ADDRESS2 = I DATA(5)
0532 CUST.CITY = I DATA(6)
0533 CUST.ST = I DATA(7)
0534 CUST.ZIP = I DATA(8)
0535 CUST.COUNTRY = I DATA(9)
0536 CUST.PHONE = I DATA(10)
0537 CUST.FAX = I DATA(11)
0538 CUST.HISTORY = I DATA(12)
0539 * Update the file
0540 CALL UI EX.CUST.UPDATE(CUST.ID, MAT CUST.REC, FN.CUST, FN.CUST.XREF)
0541 OK = TRUE ; * Set the SUCCESS indicator
0542 END ELSE
0543 OK = FALSE ; * Set the FAILURE indicator
0544 END
0545 DONE = TRUE ; * proceed to next record
0546 RETURN
0547 *
0548 *
0549 *****
0550 *
0551 * The RESTART subroutine prepares the internal field data array (I DATA),
0552 * customer record array (CUST.REC) and ID for a new customer record.
```

```

0553 *
0554 RESTART: *
0555 *
0556 CUST.ID = ''
0557 MAT CUST.REC = ''
0558 MAT IDATA = ''
0559 RETURN
0560 *
0561 *
0562 *****
0563 *
0564 * The CHECK.CHANGED subroutine checks if any internal field data has been
0565 * changed. The OK indicator variable is set to FALSE if any data is
0566 * changed, otherwise it is set to TRUE. The ID field is not checked, since
0567 * it is appropriately handled by the CHECK.ID subroutine.
0568 *
0569 CHECK.CHANGED: *
0570 *
0571 OK = FALSE
0572 IF CUST.CONTACT # IDATA(2) THEN RETURN
0573 IF CUST.NAME # IDATA(3) THEN RETURN
0574 IF CUST.ADDRESS1 # IDATA(4) THEN RETURN
0575 IF CUST.ADDRESS2 # IDATA(5) THEN RETURN
0576 IF CUST.CITY # IDATA(6) THEN RETURN
0577 IF CUST.ST # IDATA(7) THEN RETURN
0578 IF CUST.ZIP # IDATA(8) THEN RETURN
0579 IF CUST.COUNTRY # IDATA(9) THEN RETURN
0580 IF CUST.PHONE # IDATA(10) THEN RETURN
0581 IF CUST.FAX # IDATA(11) THEN RETURN
0582 IF CUST.HISTORY # IDATA(12) THEN RETURN
0583 OK = TRUE
0584 RETURN
0585 *
0586 *
0587 *****
0588 *
0589 * The CHECK.ABANDON subroutine calls CHECK.CHANGED. If any changes are
0590 * found, a message is displayed and the user is prompted to abandon the
0591 * changes. If there are no changes, or if the user decides to abandon
0592 * changes, the OK indicator variable is set to TRUE. Otherwise it is set
0593 * to FALSE.
0594 *
0595 CHECK.ABANDON: *
0596 *
0597 GOSUB CHECK.CHANGED
0598 IF NOT(OK) THEN
0599   PRINT EL: 'Do you want to abandon all your changes? ':BEL:
0600   INPUT ANS:
0601   IF ANS[1,1] EQ 'Y' OR ANS[1,1] EQ 'y' THEN OK = 1
0602   PRINT EL:
0603 END
0604 RETURN
0605 *
0606 *
0607 *****
0608 *
0609 * The DSPSCRN subroutine is used to refresh the entire screen.
0610 *
0611 DSPSCRN: *
0612 *
0613 * Clear the screen
0614 PRINT CLR:
0615 *
0616 * Display heading & labels
0617 PRINT @(5,0): 'Customer File Maintenance':
0618 FOR XLINE = 1 TO NUMFLDS
0619   LBWD = CONTROL(XLINE)<1,4>
0620   LBTX = (XLINE 'L#2 '): CONTROL(XLINE)<1,1>; * Prepend line number to label
0621   IF LBWD > 0 THEN LBTX = (LBTX : STR('.',LBWD))[1,LBWD]; * Pad label with dots

```

```
0622 PRINT @(CONTROL(XLINE)<1, 2>, CONTROL(XLINE)<1, 3>): LBTX:
0623 NEXT XLINE
0624 *
0625 * Display the field data
0626 GOSUB DSPDATA
0627 RETURN
0628 *
0629 *
0630 *****
0631 *
0632 * The DSPDATA subroutine is used to refresh the field data for all fields.
0633 *
0634 DSPDATA: *
0635 *
0636 FOR XLINE = 1 TO NUMFLDS
0637 GOSUB DSPLINE
0638 NEXT XLINE
0639 RETURN
0640 *
0641 *
0642 *****
0643 *
0644 * The DSPLINE subroutine is used to refresh the field data for one field.
0645 * The field to be refreshed is specified by the XLINE variable.
0646 *
0647 DSPLINE: *
0648 *
0649 MSK = 'L#': CONTROL(XLINE)<1, 7>
0650 PRINT @(CONTROL(XLINE)<1, 5>, CONTROL(XLINE)<1, 6>): I DATA(XLINE) MSK:
0651 RETURN
0652 *
0653
```