AA-5269B-TK

Word Processing System
List Processing User's Manual

digital equipment corporation • maynard, massachusetts

Preliminary Edition, March 1977
1st Edition, March 1977
2nd Printing (Rev) October 1977

Printed in U.S.A

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

CONTENTS

CONTENTS (CONT)

FIGURES

FIGURES (CONT)

TABLES

CHAPTER 1
INTRODUCTION

The List Processing Package is a special feature of the Word Processing System. It allows you to produce the same kind of output over and over again – a letter, purchase requisition, baseball box score, row in a table – with just certain information varying from one piece of output to another. For example, each letter produced by the List Processing Package will contain the same body but will be addressed to a different recipient. Similarly, each purchase requisition produced by the List Processing Package will contain different part names, quantities, and costs but each will look like it was written on sheets ripped off a standard order pad.

1.1 APPLICATIONS

There is really no limit to the number of different list processing applications. You can tailor a particular combination of forms to be output and information to be plugged into the forms that satisfies your special needs.

There are two broad categories of list processing applications:

- Producing copies of the same text with just one or a few pieces of different text in each – a form letter to be sent to each recipient on a mailing list, for example. Figure 1-1 shows some year end thank you letters to customers of the Southport Book Store; all were produced by the List Processing Package.

- Producing a table where each line is typed in the same format, contains the same kind of information, but has different actual values – a telephone directory, for example, where each line has the same structure as every other but has a different name, address, and telephone number. The following example, produced by the List Processing Package, shows overdue accounts at the Southport Book Store.

Outstanding Accounts Payable

Name	Amount Due in $	Overdue Code in months
Hartford, Alice	8.95	2
Mahoney, Charles	84.50	1
McDonald, Joseph	107.16	4

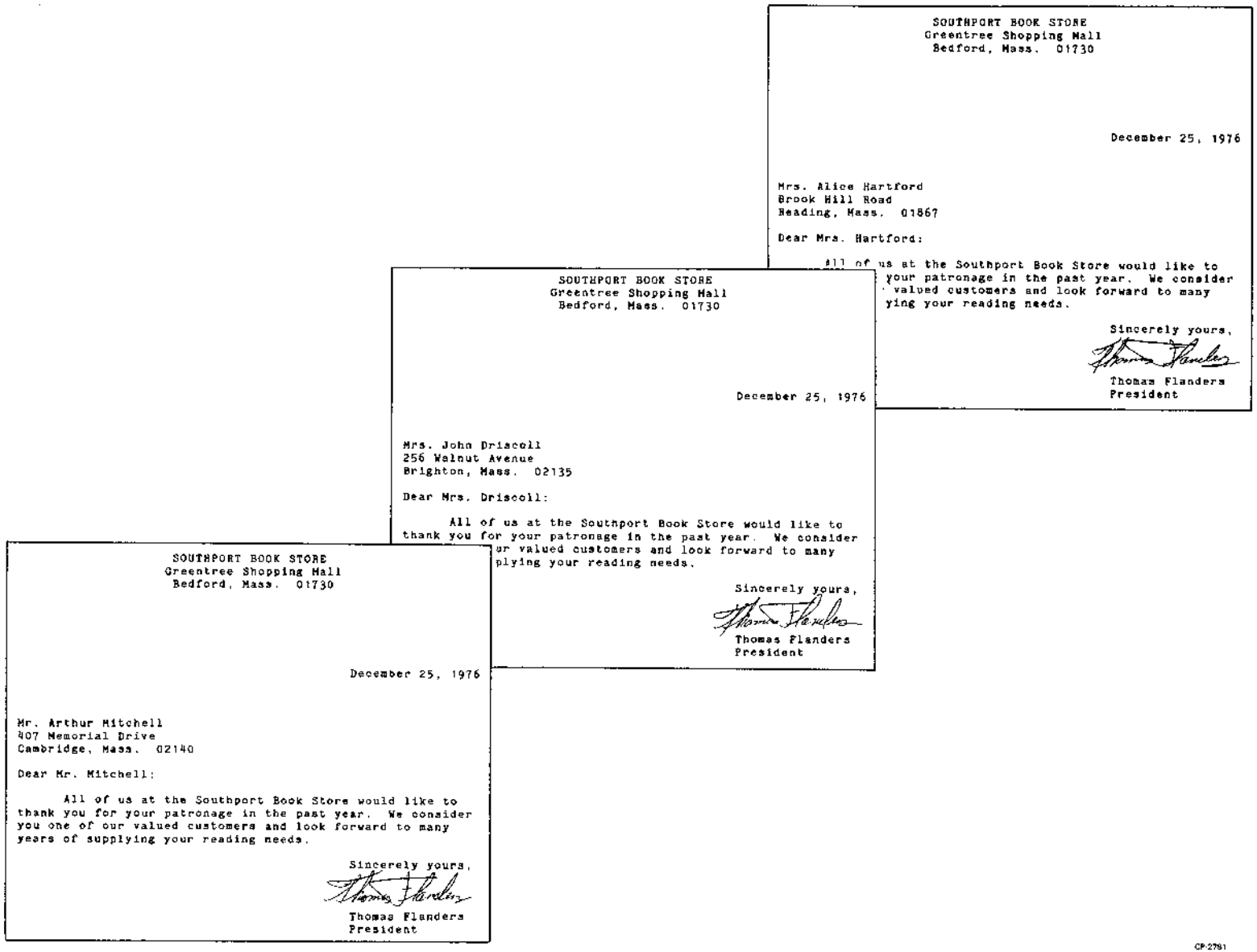


Figure 1-1 Form Letters Produced by the List Processing Package

1.2 SUMMARY OF FEATURES

The List Processing Package has many features that make the preparation of output from various forms and from different list entries an easy job.

- **Efficiency of use** – The List Processing Package saves a typist's time by automatically typing repetitious information. Form letters are a good example of this efficiency. All you have to do is type the names and addresses once, type a skeleton form that contains the body of the letter plus indicators showing where the names and addresses are to be plugged in, and tell the List Processing Package to go. It will then type out a separate letter to each recipient.

- **Adaptability to change** – The List Processing Package also saves time when it's necessary to change the wording or format of some output and completely do it over. If you decide you don't like the way your output looks, you can easily change its appearance and have the system type out a new copy. For example, if the columns in a table look too jammed together, you can change the tab settings and sit back while the List Processing Package retypes the table.

- **Variety of output** – You can produce many different kinds of output from the same collection of information. For example, you can type out letters addressed to people on a mailing list and also type out a table listing the letter recipients. In the Southport Book Store example just presented, two different kinds of output were produced: a letter to all entries in the list, and a table summarizing the overdue accounts. A third output, dunning letters (requests to pay up) to overdue accounts, could also be produced. Figure 1-2 shows how these might look.

- **Selectivity in processing** – You can be selective in deciding what to type out: you can produce output for every entry in the list (the Southport Book Store's thank you letters), or you can be choosy and process only a select group (the dunning letters to overdue accounts).

- **Customization of output (1)** – By incorporating the same piece of information several times in the same piece of output you can get a customized product. For example, you can produce a letter that looks like (an excerpt from a longer letter) this.

Miss Mary Canfield
101 North Chelsea Avenue
Warwick, Rhode Island 02889

Dear Mary,

We are sure you will agree, Mary, that reading a
newspaper helps keep you aware of what's happening.
Many of your neighbors on North Chelsea Avenue
already subscribe to the Daily Mercury and . . . . .

SOUTHPORT BOOK STORE
Greentree Shopping Mall
Bedford, Mass. 01730

December 25, 1976

Mrs. Alice Hartford
Brook Hill Road
Reading, Mass. 01867

Dear Mrs. Hartford:

Our records indicate that you have owed the Southport Book Store $8.95 for more than 2 months. We would appreciate your attention to this matter with no further delay.

Sincerely yours,

Ralph McKelvey
Treasurer

P.S. If you have recently paid this small amount, please accept our apologies for this letter.

SOUTHPORT BOOK STORE
Greentree Shopping Mall
Bedford, Mass. 01730

December 25, 1976

Mr. Joseph McDonald
23 Hampshire Street
Waltham, Mass. 02154

Dear Mr. McDonald:

Our records indicate that you have owed the Southport Book Store $107.16 for more than 4 months. We would appreciate your attention to this matter with no further delay.

Sincerely yours,

Ralph McKelvey
Treasurer

P.S. Unless this balance is paid within ten days, we will be forced to turn your account over to our lawyers.

CP-2779

Figure 1-2 Dunning Letters Produced by the List Processing Package

The letter to Miss Canfield has been customized to include the recipient's name and address at several different places. You can produce a similar customized letter to another recipient that might look like this.

Mr. Nathaniel Whitely
49 Old Road to Four Acres Corners
Concord, Massachusetts 01742

Dear Nathaniel,

We are sure you will agree, Nathaniel, that reading a newspaper helps keep you aware of what's happening. Many of your neighbors on Old Road to Four Acres Corners already subscribe to the Daily Mercury and

The above letter includes the recipient's name and address, just like the previous one. But notice that it has been further customized by having the system stretch the text to fit around the longer name and the longer location.

- **Customization of output (2)** – You can also produce a custom product by adding some special information to a particular piece of output. For example, Southport Book Store's thank you letter to Mr. Mitchell could very easily have been customized by adding the following postscript at the bottom.

  P.S. Art, hope your golf game gets under 77 in '77.

- **Flexibility in preparation** – The pieces of information that make up a list entry can be arranged in any order. They don't have to be in the same order as they will appear in the output. For example, if you want to produce a table of employees with the last name of each followed by the first, the entries in your list could nevertheless all have first name followed by last name. What's more, you can arrange the information in one list entry in a different order from that in another entry. Some of the entries could have first name followed by last name, and some others could have last name followed by first name.

1.3 PROCESSING

In order for the List Processing Package to type some output, you must tell the system to merge certain pieces of information from one selected list entry into a form, to output the merged form, and to repeat the merge/output process for other selected list entries. This process is shown in Figure 1-3.

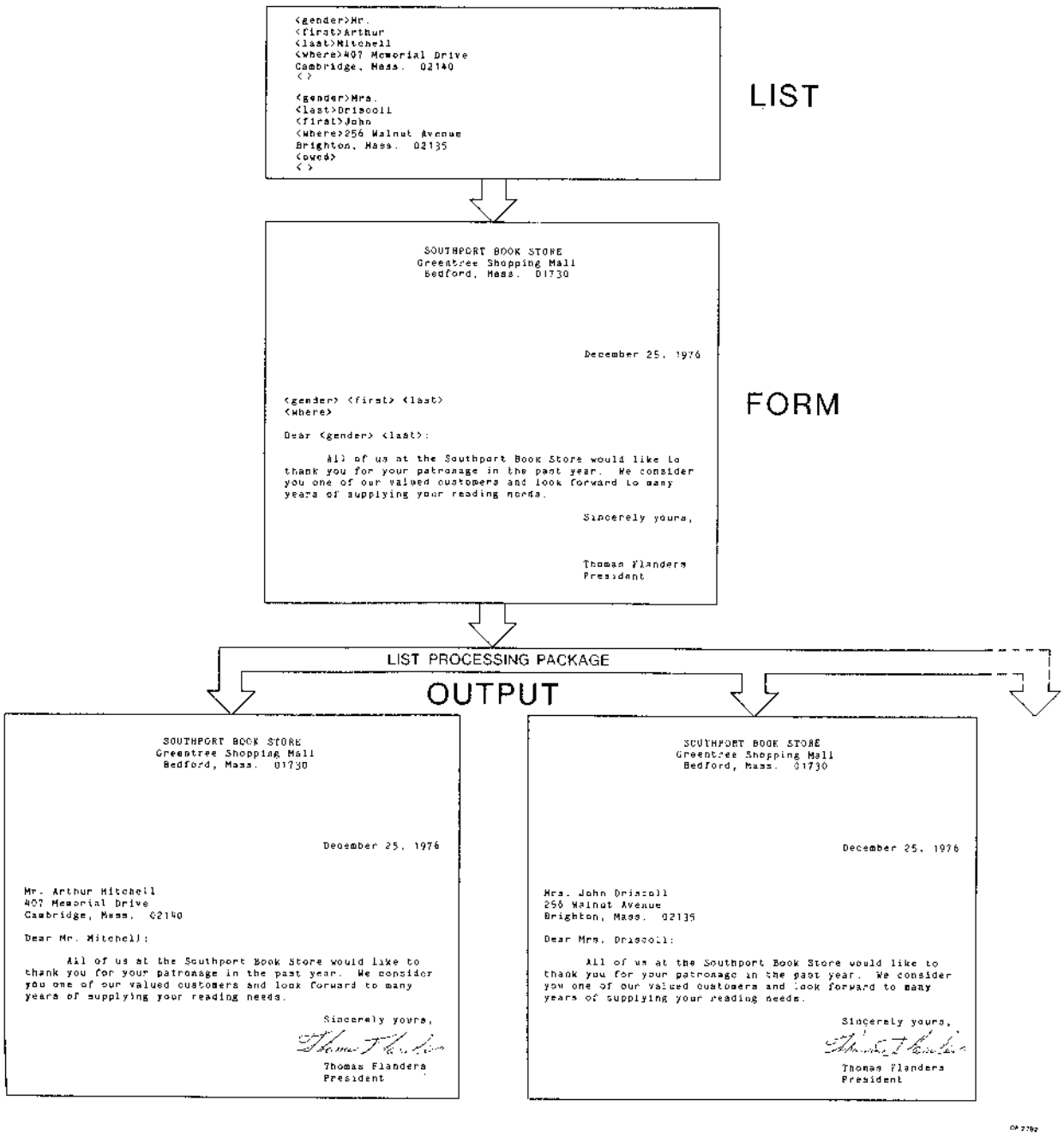


Figure 1-3 Producing Output with the List Processing Package

CHAPTER 2
MECHANICS OF USE

The mechanics of list processing involves four basic steps:

1. Building a list (database),
2. Creating a form,
3. Deciding and specifying which list entries should be processed,
4. Telling the List Processing Package to produce output.

You don't have to follow the above steps in the order listed (except for the last one). Sometimes you might need to develop a large database and one or two forms to be used with it. In this case it's generally a good idea to build the list before the form, since it's easier to change one form than many entries in a list. Othertimes you might have some forms that must be produced in a very prescribed way. In this case you would probably create the forms before the list.

The way your list is built influences how you create your form, and vice versa. It's a little like the old chicken and egg question as to what order you will follow. Your application, what you need to produce, and what information you have to work with all determine the order of the mechanics you will go through in using list processing.

2.1 BUILDING A LIST

A list, or database, contains information that is to be plugged into a form so that the merged results can be typed out. For each piece of output there must be a separate entry in the list. To build a list, do the following:

1. Design a "pattern" or "template" having the names of different pieces of information (fields) you wish to maintain for your entries. Continuing the example presented earlier in this section, the Southport Book Store maintains gender, first, last (for a person's name), and where (for an address) for all entries in its list and the amount owed, an overdue code, and a postscript (to be added) to a bill for just some entries. Make your pattern as complete as possible.

2. Create a document on your diskette to hold the list.

3. For each particular entry, type an identification of the piece of information (same as in the pattern) followed by the value. Here's the first line of the first entry in the Southport Book Store's database.

   <gender>Mr.

4. Continue typing field names and values until you have given all possible information for the entry. The complete first entry in the Southport Book Store's database looks like this.

```
<gender>Mr.
<first>Arthur
<last>Mitchell
<where>407 Memorial Drive
Cambridge, Mass.  02140
<>
```

You can be very flexible in creating an entry:

- All the field names need not be present and accounted for in each record.
- A field name doesn't have to be followed by a value.

Here's another entry from the Southport Book Store's database.

```
<gender>Mrs.
<first>Alice
<last>Hartford
<where>Brook Hill Road
Reading, Mass.  01867
<owed>8.95
<overdue>2
<ps>P.S.
If you have recently paid this small amount, please
accept our apologies for this letter.
<>
```

5. Repeat step 4 for each entry.

Chapter 3 gives details on building a list.

2.2 CREATING A FORM

A form is a document that contains the text you want output plus indicators of where information is to be plugged in. To create a form, do the following:

1. Create a document on your diskette to hold the form.
2. Write a "skeleton" form that has the text you want output. Indicate every place in the skeleton where information is to be plugged in. Type the completed form into the document. You can be very flexible in creating a form:
   - The same field can be used one or more times in the same piece of output.
   - Every single field does not have to be plugged into your output.

Here's the form that was used to produce the Southport Book Store's thank you letters. Notice that the fields called gender and last are used twice.

Chapter 4 gives details on creating a form.

```
                    SOUTHPORT BOOK STORE
                   Greentree Shopping Mall
                    Bedford, Mass.  01730



                                              December 25, 1976


<gender> <first> <last>
<where>

Dear <gender> <last>:

      All of us at the Southport Book Store would like to
thank you for your patronage in the past year.  We consider
you one of our valued customers and look forward to many
years of supplying your reading needs.

                                              Sincerely yours,


                                              Thomas Flanders
                                              President
```

Figure 2-1 Example of a Form Document

2.3 SPECIFYING WHAT TO PROCESS

Once the list (database) and the form(s) are prepared, you have the option of merging each and every list entry with the form for output, or being choosy by using only certain entries while ignoring others. To tell the system which entries to process, do the following:

1. Decide which particular entries from the database are to be processed.
2. Create a document to hold your decision rules (selection specification).
3. Write your selection criteria in a special format (described in a later section) and type it into the document.

You can merge different groups of database records with the same or a different form simply by creating different selection specifications. For example, the Southport Book Store used two different selection specifications with the same database to produce two different kinds of output: a thank you letter to all list entries, and a dunning letter to selected entries. Here's the selection specification used to produce the dunning letters.

```
if <overdue> =<2> or more
then process record
```

Chapter 5 gives details on creating a selection specification.

2.4 PRODUCING OUTPUT

Once the list, form, and selection specification are prepared, you are ready to produce output. To tell the system you want to merge list entries with a form according to a selection specification and output the results, do the following:

1. Call up the List Processing Package.
2. Tell the system whether you want to print the output now or save it (in a document) to be printed at some other time.
3. Identify your list, form, and selection specification documents.
4. If you don't want the system to apply the selection specification to each entry in the list, tell it the entry number (counting goes from 1 up to the end) where processing is to start and/or stop.
5. If you intend to print the output now, check that there is an adequate supply of paper in the printer.
6. Tell the system to go.
7. Collect the printed output, if produced.

Chapter 6 gives details on using the List Processing Package.

CHAPTER 3
HOW TO CREATE A LIST DOCUMENT

A list document, or database, contains information that is to be merged with a form to produce output. It is organized into units called records. Each record contains information that relates to one particular entity – an individual employee in the case of personnel data, a separate part in the case of inventory data, or a single patient in the case of hospital data. Figure 3-1 shows the organization of a list document (database of the Southport Book Store's customers, to continue the example in Chapter 2).

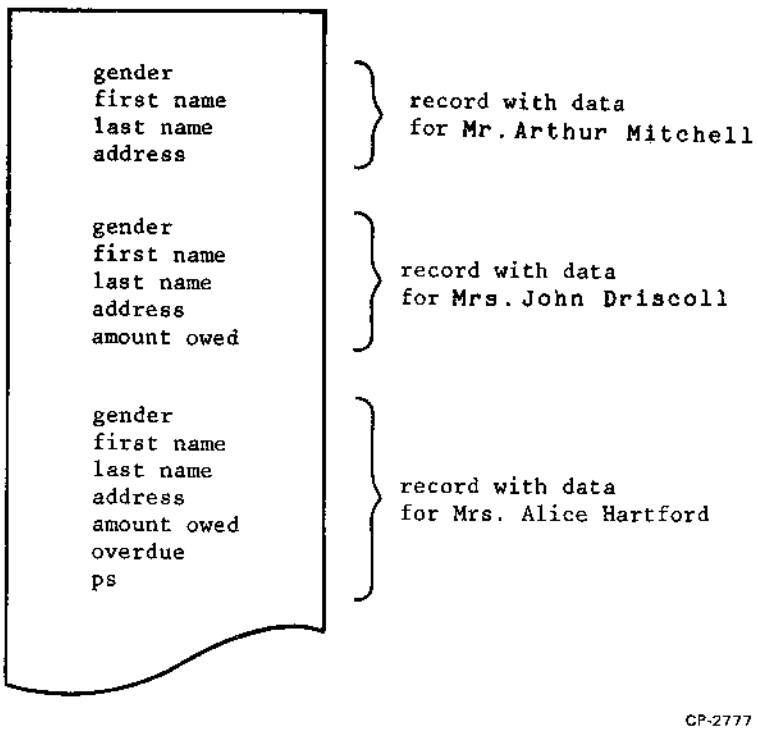


Figure 3-1 General Organization of a List Document

A record is made up of one or more pieces of information called fields. Each field is made up of a name and a value. The format for creating a list record is shown in Figure 3-2. The field name is typed inside a pair of angle brackets, <>, and the value is typed immediately after the >.

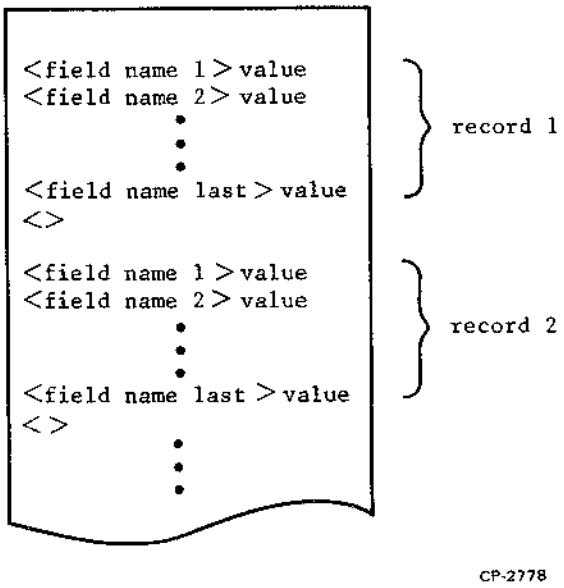


Figure 3-2 Format for Creating a List Document

Suppose you have a form letter that looks like this,

<gender> <first> <last>
<where>

Dear <gender> <last>:

All of us at the Southport Book Store would ....

and you want to output a merged copy to Mr. Arthur Mitchell. You would create a record in your list document that would look like this.

<gender>Mr.
<first>Arthur
<last>Mitchell
<where>407 Memorial Drive
Cambridge, Mass. 02140
<>

If you want to send the same letter to Mrs. John Driscoll, just add another record to your list document. It doesn't matter whether you put Mrs. Driscoll's record in front of Mr. Mitchell's, after it, or whether they're separated by many other records. The results will still be the same.

3.1 GUIDELINES FOR CREATING A RECORD

The different pieces of information that you put into a record generally depend on the kind of output you intend to produce. You should make the collection of different fields as complete as possible but not excessive. For example, if your form uses an entire address, there is no reason to have a record that looks like,

```
<street>34 Mill Street
<town>Bellows Falls
<state>Vermont
<zip> 05101
<>
```

when the following record would do the same job.

```
<where>34 Mill Street
Bellows Falls, Vermont  05101
<>
```

The complete collection of all possible fields to be used in constructing your list document is a "pattern" or "template." You can use it much in the same way that you would a tissue paper sewing pattern for a dress or a plastic template for drafting schematic diagrams.

While there are some rules that must be followed in creating a record, there are also some non-rules. As the name implies, they really aren't rules: you don't have to follow them, but if you do, your list will be easier to prepare and to modify.

3.1.1 Record End

Each record must have the characters <> after the last field. To make list editing easier, use two carriage returns after the <>. This makes your list more readable and also lets you move efficiently from record to record when editing by pressing the Blue PARA search key.

Remember to put <> at the end of the last record in the list.

3.1.2 Size

A record cannot have more than 2500 characters.

3.1.3 Field Name Restrictions

Any keyboard character except < and > may be used in a field name. It must not have more than 30 characters and must not start with !.

Use meaningful field names. For example, if you want to create records with people's names, call the fields first and last instead of (the single letters) f and l. However, if you use long field names, you will use up space on your diskette. A field named <ci-st-z> is less meaningful than <city-state-zip> but is certainly more economical in diskette storage.

3.1.4 Order of Fields

Fields may be arranged in different orders in different records. For example, you could have the following two records in a list.

```
<date>January 1977
<edition>Fourth
<printing>Second
<>

<edition>Fourth
<printing>First
<date>September 1975
<>
```

The List Processing Package will correctly process both records when you tell it to merge them with a form.

3.1.5 Presence/Absence of Fields

All possible field names do not have to be present and accounted for in every record. For example, you could have the following two records in a list. The first record contains fields gender, first, last, and where. The second record is missing the field called first.

```
<gender>Miss
<first>Mary
<last>Walker
<where>PK3-2/T12
<>

<gender>Mr.
<last>Harris
<where>MR1-2/E75
<>
```

When the List Processing Package cannot find a piece of information in a record to plug into a form, it just continues to output the form, skipping the missing field altogether (no blanks are inserted). A mailing label produced by the List Processing Package for the first record given above would look like

```
Miss Mary Walker
PK3-2/T12
```

while the second would look like this.

```
Mr. Harris
MR1-2/E75
```

3.1.6 Fields Without Values

Every field does not have to have a value. Nothing will be inserted in the output when the List Processing Package merges the record with the form. In the example just presented, the field called first was missing entirely. It could be present but without a value, as in the following record.

```
<gender>Mr.
<first>
<last>Harris
<where>MR1-2/E75
<>
```

Here the field called first has only a carriage return after it. The list processing results would be the same as before – a mailing label addressed to Mr. Harris.

3.1.7 Typing a Field Name

Type a record's field name exactly the way that it is spelled both in the form that the records will be merged with and in the selection specification. Upper and lowercase letters must match exactly. Use spaces between the field name and the brackets only if the field name in the form and in the selection specification appears the same way.

Suppose you have a form for producing mailing labels that looks like this,

```
<gender> <first> <last>
<where>
```

and you want to make a label for Mr. Arthur Mitchell. You would create a record in your list document that would look like,

```
<gender>Mr.
<first>Arthur
<last>Mitchell
<where>407 Memorial Drive
Cambridge, Mass.  02140
<>
```

Notice that all the field names are spelled exactly the same in the form and in the list record. Mr. Mitchell would never get his letter if the record had <Where> or < where >.

3.1.8 Field Value Restrictions (1)

Type a field's value exactly as you want it to appear in the output. This rule means you should not put any spaces (tabs, carriage returns, or other control characters) between the > and the field value unless you want those spaces to appear in your output. Continuing the previous form letter example, the salutation would look stretched out, like this,

Dear Mr. Mitchell:

if three spaces had been put in the list record between <last> and Mitchell.

NOTE

A field value is everything immediately after the > up to, but not including the last explicit carriage return, blank, tab, page marker, new page marker, or next <.

3.1.9 Field Value Restrictions (2)

Type a field's value exactly the way it appears in the selection specification. Upper and lowercase letters must match exactly. Use spaces after the > that ends the field name only if the selection specification has the same number of spaces right after the =.

For example, if a record looks like

```
<name>Jones
<town> Chicago
<>
```

it will not be merged with a form when the following selection is applied.

```
if <town> =Chicago
then process record
```

The List Processing Package will look for records whose town field has the value "Chicago". It will not process the above record because the value is " Chicago" with a space in front of the letter C.

3.1.10 Invisible Hyphens

When long words in a record are merged with a form, they might sometimes cause awkward line endings. To improve the appearance of your output, put invisible hyphens in such words in your list. For example, suppose you have a form that looks this.

```
The next meeting of the <club> will be held
on <when> in <where>. Please try to attend.
```

The output would present a neat appearance if used to issue a meeting of the Ski Club or the Coffee Clatch. But how about issuing a meeting notice for the International Benevolent Protective Order of Elks? Here the right margin might turn out jagged, as seen below.

> The next meeting of the International
> Benevolent Protective Order of Elks will be
> held on March 20 in room 2A of the meeting
> hall. Please try to attend.

If invisible hyphens were put in the record to mark syllables in the three big words, the meeting notice would look neater.

> The next meeting of the International Be-
> nevolent Protective Order of Elks will be
> held on March 20 in room 2A of the meeting
> hall. Please try to attend.

3.1.11 Format Control Characters and Rulers

Tabs and other format control characters may be placed in list records. When text with such control characters is merged with a form, the ruler from the form document and not from the list document is used to format the output. You can still place rulers in your list. Keep in mind, however, that rulers in list documents have no effect on how the record information appears in list processing output.

For example, the record in Figure 3-3 uses a tab stop right after the characters P.S. Because the list document contains a ruler with a word wrap at column 12, the postscript text is tabbed in at that column.

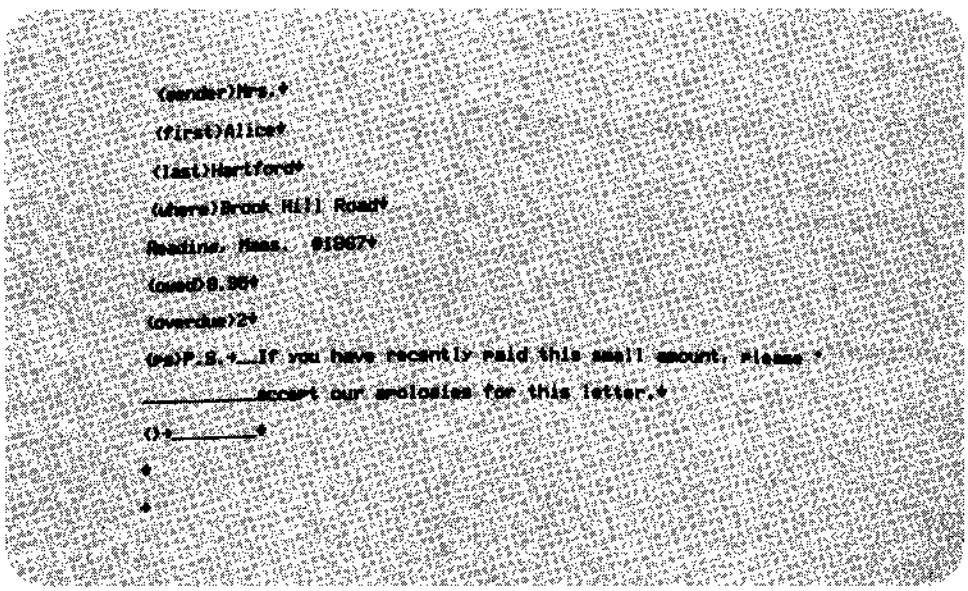


Figure 3-3 Example of a Record with Word Wrap Indent

Notice in Figure 3-4 that when this record is merged with a form (the Southport Book Store's dunning letter), the text of the postscript starts at column 7 with the continuation line justified at the left margin.

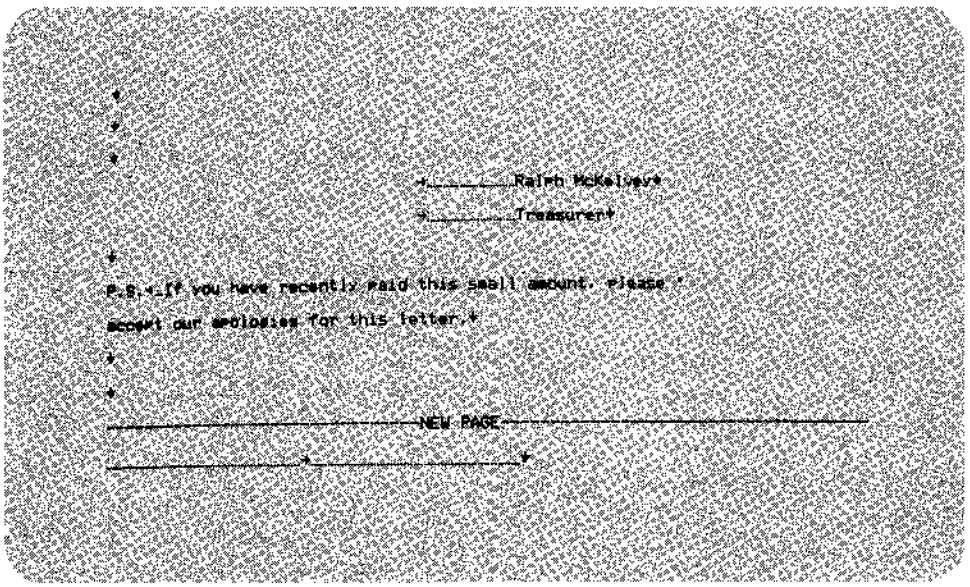


Figure 3-4 Using a Record with Word Wrap Indent in a Form Without Word Wrap

3.2 GUIDELINES FOR CREATING A LIST

3.2.1 Size

A list cannot have more than (approximately) 4000 records.

3.2.2 Shorthand Abbreviation Library

You can save typing time if you put the names of all fields that may possibly appear in a record into the Shorthand Abbreviation Library (document 2 on the System Diskette). Then every time you want to create a new record, just press the Gold and the ABBRV keys, type the two-character abbreviation name, and the entire set of field names will be placed in your list document. Here's what the abbreviation used to create the Southport Book Store's records looks like.

<gender>
<first>
<last>
<where>
<owed>
<overdue>
<ps>
<>

Once the entire set of field names, or template, is in your list document, you can move around with any of the edit keys. The Blue <> key should be very useful in adding field contents or deleting field names. When pressed, this key moves from the end of one field name to another (stops immediately after the > character).

3.2.3 Editing Records

If you place two carriage returns right after the <> ending each record, you then can move from the start of one record to the start of another by pressing the Blue PARA key.

3.2.4 Header Information

You may place some heading text in front of the first record in a list document. When the list is used for output, the List Processing Package will ignore the header information. Here's the beginning of a list document (the Southport Book Store's customer accounts) that has some heading text. The complete list document appears at the end of this chapter.

CONTENTS: Customer accounts. Each record has name (fields gender, first, and last). Some records have the outstanding account balance (field owed), given in dollars and cents. Some records with outstanding balances have the number of months overdue (field overdue). Some of these latter records also have a line or two of postscript to be added to their dunning letter (field ps).

<gender>Mr.
<first>Arthur
<last>Mitchell
<where>407 Memorial Drive
Cambridge, Mass. 02140
<>

As seen above, the header can be used to identify the contents of a list document.

NOTE
A header may contain any characters except < and >.

3.2.5 Sublists

A list can be broken down into one or more smaller lists by a technique explained in Section 4.1.10. The original or master list remains on your diskette no matter how many sublists are created. This capability could be used, for example, to make a separate mailing list database for residents of a particular geographic area.

3.3 EXAMPLE

The following list document shows the entire database for the Southport Book Store examples seen throughout this manual.

CONTENTS: Customer accounts. Each record has name (fields gender, first, and last). Some records have the outstanding account balance (field owed), given in dollars and cents. Some records with outstanding balances have the number of months overdue (field overdue). Some of these latter records also have a line or two of postscript to be added to their dunning letter (field ps).

```
<gender>Mr.
<first>Arthur
<last>Mitchell
<where>407 Memorial Drive
Cambridge, Mass.  02140
<>

<gender>Mrs.
<last>Driscoll
<first>John
<where>256 Walnut Avenue
Brighton, Mass.  02135
<owed>
<>

<gender>Mrs.
<first>Alice
<last>Hartford
<where>Brook Hill Road
Reading, Mass.  01867
<owed>8.95
<overdue>2
<ps>P.S.  If you have recently paid this small amount, please accept our apologies for this
          letter.
<>

<gender>Mr.
<first>Charles
<last>Mahoney
<where>257 Raven Road
Bedford, Mass.  01730
<owed>84.50
<overdue>1
<>

<gender>Mr.
<first>Joseph
<last>McDonald
<where>23 Hampshire Street
Waltham, Mass.  02154
<owed>107.16
<overdue>4
<ps>P.S.  Unless this balance is paid within ten days, we will be forced to turn your account
          over to our lawyers.
<>
```

CHAPTER 4
HOW TO PREPARE A FORM DOCUMENT

A form is a document that contains the text to be typed out plus indicators showing where and what kind of information is to be inserted. These indicators are the names of list record fields enclosed in angle brackets <>. Wherever the List Processing Package sees an indicator, it takes the value of the corresponding field from the record being processed and plugs it into the form. Figure 4-1 shows how a form document is used to produce output for a selected record.

4.1 GUIDELINES FOR CREATING A FORM

There are no rigid rules to follow in creating a form. It is a fairly simple, straightforward job. Just type the text you want merged for output into a document. At every place in the text where you want a piece of information plugged in, type the field name inside angle brackets.

4.1.1 Relationship to Database (1)

Just as you must consider your form when constructing your list document, you must also consider the pieces of information in your list when designing your form. The form should reflect the structure of the database (list document). Consider, for example, the following record.

```
<gender>Mr.
<first>Peter
<last>Adams, Jr.
<where>58 Foster Street
East Orange, New Jersey  07019
<>
```

If the form looks like this,

```
<gender> <first> <last>
<where>

Dear <first>:
```

the list processing output would be acceptable for mailing a letter to Mr. Peter Adams, Jr. However, suppose the form looks like this.

```
<gender> <first> <last>
<where>

Dear <gender> <last>:
```

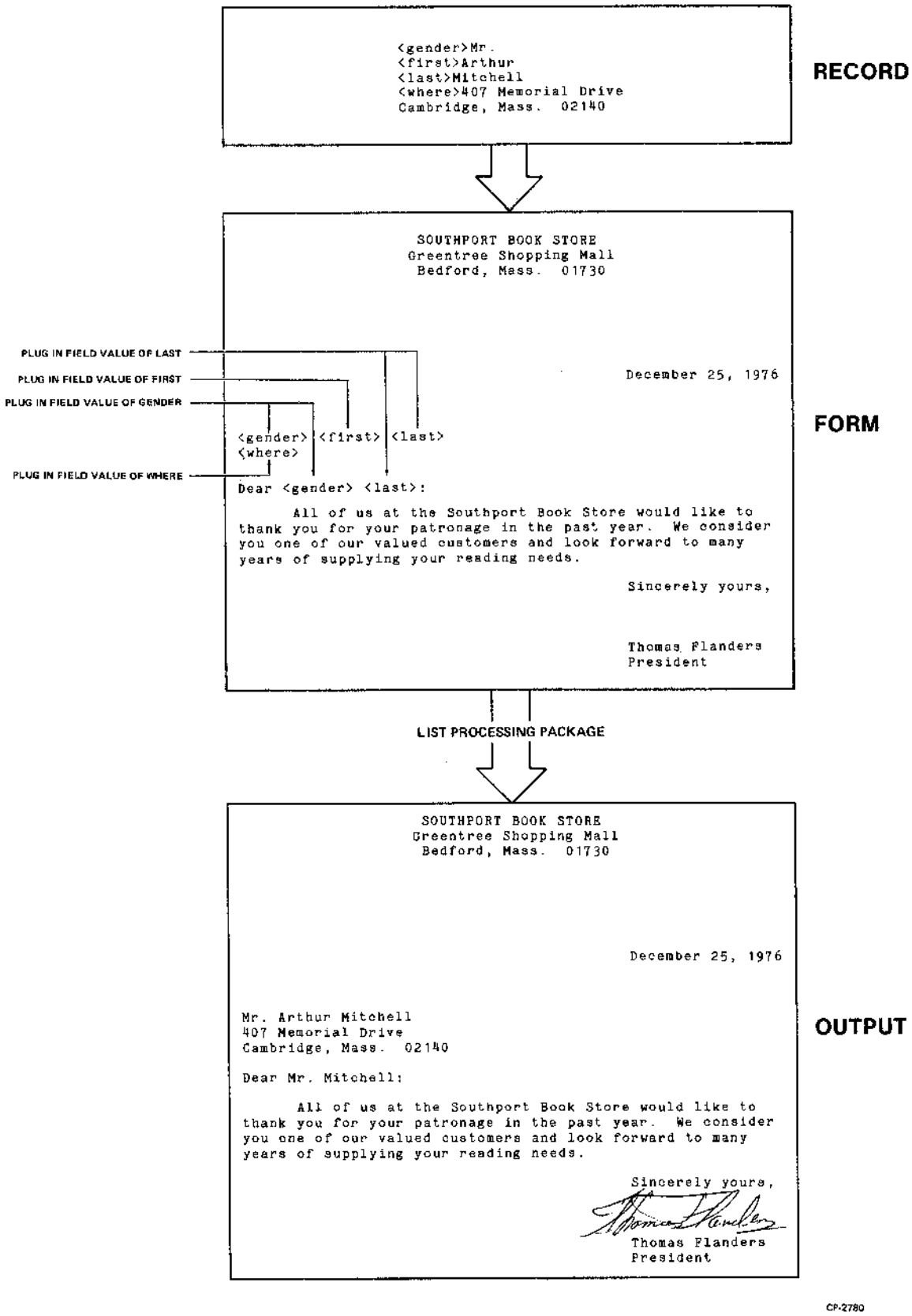


Figure 4-1 Merging Field Values into a Form for Output

Now the output produced by the List Processing Package would look like this.

Mr. Peter Adams, Jr.
58 Foster Street
East Orange, New Jersey 07019

Dear Mr. Adams, Jr.:

This output is not acceptable because saying "Dear Mr. Adams, Jr." is not correct English grammar. Here the form made no allowance for the possibility of Jr. or Sr. after a last name. Thus, it did not accurately reflect the database. The next guideline presents a solution to this problem.

4.1.2 Relationship to Database (2)

You can specify a field name in your form even though that field may not be present in every record in the list. Whenever the List Processing Package cannot find a field name in a record, or finds the name but no value, it just continues to output the merged form, skipping the missing field altogether (no blanks are inserted). For example, you could have a form that looks like this.

<gender> <first> <last><jr/sr>
<where>

Dear <gender> <last>:
:
:
:
:

If a record has no jr/sr field, nothing is inserted in that position in the output. The list record

<gender>Ms.
<first>Anne
<last>Willis
<where>Greycoat Lane
Ipswich, Mass. 01938
<>

would produce the following output when merged with the form.

Ms. Anne Willis
Greycoat Lane
Ipswich, Mass. 01938

Dear Ms. Willis:
:
:
:
:

Only when the List Processing Package finds a record with a value in the jr/sr field does it insert information in that position in the output. Recall the "Dear Mr. Adams, Jr." example. Suppose Adams' record looks like this.

<gender>Mr.
<first>Peter
<last>Adams
<jr/sr>, Jr.
<where>58 Foster Street
East Orange, New Jersey 07019
<>

Correct English grammar would now be produced by the List Processing Package. Merging this revised record with the above form would give the following.

Mr. Peter Adams, Jr.
58 Foster Street
East Orange, New Jersey 07019

Dear Mr. Adams:

.
.
.
.
.

4.1.3 Multiple Use of Field

There is no limit to the number of times a field may be repeatedly used in a form. In the examples above, fields gender and first were each used twice. In fact, repeating a piece of information can help to customize your output.

4.1.4 Typing a Field Name

Type a field name exactly the way it is spelled both in the list records and in the selection specification. Upper and lowercase letters must match exactly. Use spaces between the field name and the brackets only if the field name in the records and in the selection specification appears the same way.

4.1.5 Output Pagination

The List Processing Package does not advance to a new page with each merged form output. If you are producing lines in a table or telephone directory, for example, you don't want each line to start on a new page. In these cases, your output will be correctly formatted. But if you are producing letters or reports, you will want to have each start on a new page.

There must be a new page marker at the bottom of your form document to have each piece of output start on a new page. Do this when editing your form document. Just position the cursor at the bottom and press the GOLD and the NEW PAGE keys.

4.1.6 Headers

Heading information, or text that appears only once before the first merged form output, can be produced with the List Processing Package. Type your heading information at the beginning of your form document. Put the characters <!S> at the end of the header.

Here's a form that contains a header.

Outstanding Accounts Payable

Name	Amount Due in $	Overdue Code in months
<!S> <last>, <first>	<owed>	<overdue>

When the list document that appears at the end of Chapter 3 is merged with the form, the following output is produced.

Outstanding Accounts Payable

Name	Amount Due in $	Overdue Code in months
Hartford, Alice	8.95	2
Mahoney, Charles	84.50	1
McDonald, Joseph	107.16	4

Sometimes the specification may be so selective that no records are merged with a form for output. In this case no output whatsoever is produced, not even a header.

Only one <!S> may be placed in a form document. If you accidentally put more than one <!S> in your form document, only the first one takes effect. The extra ones (plus preceding text) will appear as part of each merged form that is output.

NOTE

If a form contains <!S> and there is a ruler change in the body of the form (after the <!S>), put a copy of the original ruler into the form immediately after the <!S>.

4.1.7 Trailers

Trailing information, or text that appears only once just after the last merged form output, can also be produced with the List Processing Package. Put the characters <!E> in your form and follow them with the body of the trailer.

Here's a procurement voucher form which uses both a header and a trailer. The header gives the title of the output and describes the column contents. The trailer gives instructions for ordering supplies.

```
                    PROCUREMENT VOUCHER

Description                 Number              Cost

<!S>
<what>                      <quan>              <cost>
<!E>

Purchase Requirements:
1.  Receipt(s) must be attached.
2.  Purchases over $10.00 require manager's signature.
```

Here's how the procurement voucher might look when used to order office supplies.

```
                    PROCUREMENT VOUCHER

Description                 Number              Cost

Gem paper clips               2 doz.            $ 12.60
8-1/2 x 11 ruled pads         4 pkg.            $ 20.00
Dixon No. 3 lead pencils      6 boxes           $  8.75

Purchase Requirements:
1.  Receipt(s) must be attached.
2.  Purchases over $10.00 require manager's signature.
```

If no records are selected for merging, no header or trailer will be output.

4.1.8 Merging More Than One Record with the Same Form

The List Processing Package merges a single record with a form, outputs the result, and repeats the merge/output operation for every selected record. It is possible, however, to merge two or more records into the same copy of form being output. Just put the characters <> into the form at the point where you want to advance to the next record.

This list processing feature can be used effectively to partition similar-looking lines into groups for easier reading. Telephone directories are a prime example of such grouping: it's easier for the eye to scan line after line of name, address, and telephone number when a blank line is inserted every so many lines.

Here's a form used to output lines of names and owed balances (continuing the Southport Book Store example) into groups of three.

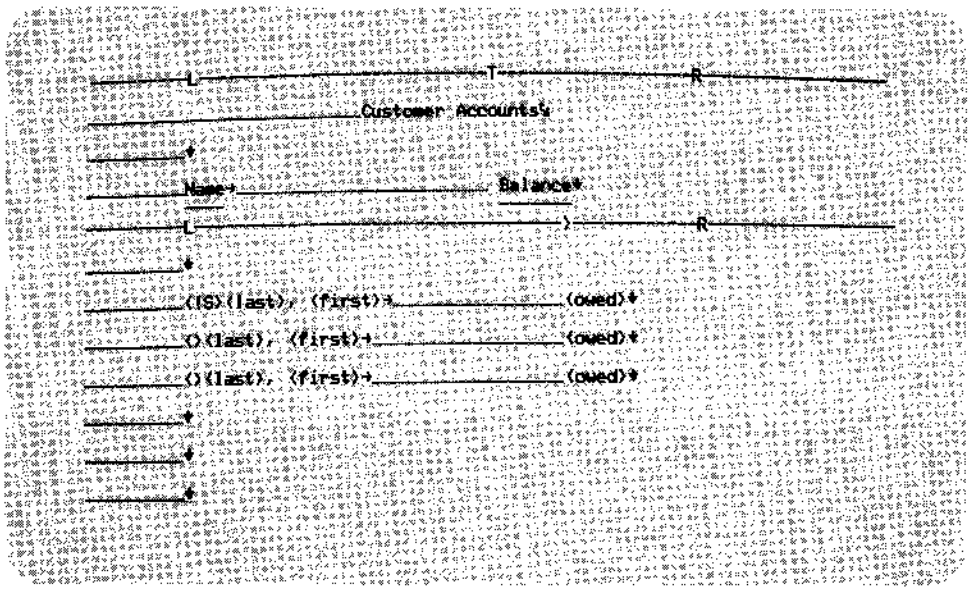


Figure 4-2 Example of a Form to Merge Three Records for Output

Notice that the characters <> appear twice, after the first and the second name/balance lines. No <> characters are used after the last line because the List Processing Package always goes to process the next list record when it reaches the end of the form. If <> had been placed at the end of the last line in the form, every fourth list record would be bypassed.

Here's the output produced by merging the Customer Accounts form with the list shown at the end of Chapter 3.

Customer Accounts

Name	Balance
Mitchell, Arthur	
Driscoll, John	
Hartford, Alice	8.95
Mahoney, Charles	84.50
McDonald, Joseph	107.16

4.1.9 Invisible Hyphens

Awkward line endings might possibly result if long words become positioned at the right margin of output. To improve the appearance of your output, put invisible hyphens in such words in your form.

This guideline is similar to the one presented in Chapter 3 for inserting invisible hyphens in list records. Suppose the general meeting notice form (Section 3.1.10) were changed. Now it is a form letter for reminding all members of the Elks about an upcoming meeting. It might look like the following.

> The next meeting of the International Benevolent Protective Order of Elks will be held on <when> in <where>. Please try to attend.

With invisible hyphens in the form to mark syllables in the three big words, the meeting notice would look neater if the margins were moved in. Here's how it might look with a narrower ruler.

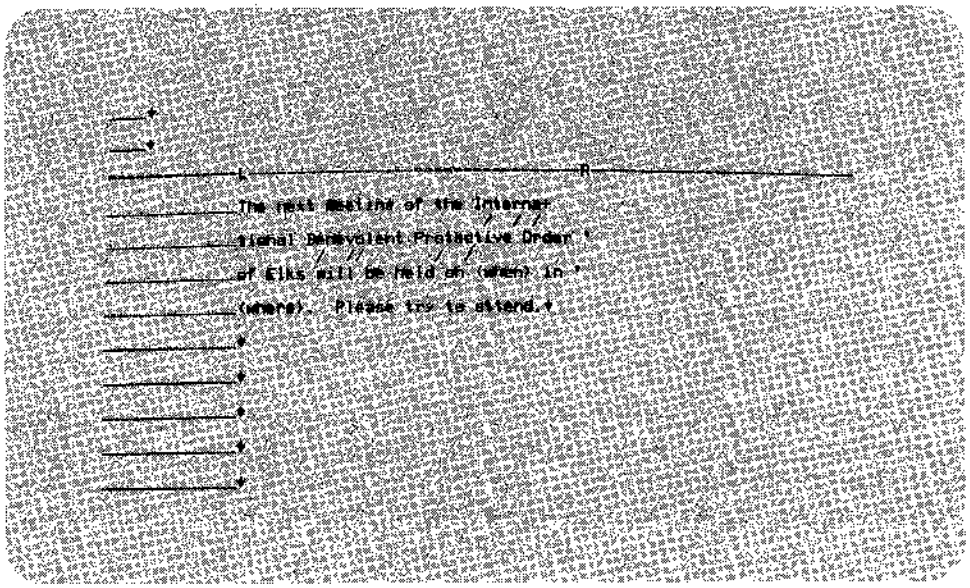


Figure 4-3 Example of Output Showing the Effect of Invisible Hyphens

The / character under letters in the preceding screen display indicates invisible hyphens. The Word Processing System will break a word at this point (adding a hyphen) if it gets positioned near the end of a line.

4.1.10 Sublists

You can select certain records from a list and directly copy them – field names, field values, and record ending <> – into a new list document without producing any merged output forms. This is accomplished with the list processing sublist creation feature. Many different sublists can be created from a single list without destroying the "master." The master list always remains the same no matter how many sublists are created.

To get a sublist, perform the following steps.

1. Create a specification document that selects only those records you want placed in the sublist.

2. Create a form document that contains the characters <!R>. Of course, the form may contain a header and/or trailer.

3. Call up the List Processing Package and identify your list, selection specification, form, and output documents.

When the List Processing Package is through running, your output document will contain the desired sublist.

An example of this feature is creating a sublist having active Southport Book Store accounts (any outstanding balance, not necessarily overdue). The form document might look like this.

```
Active Customer Accounts

<!S>
<!R>
```

Two carriage returns after the <!R> characters were used to separate the records in the sublist. Though not required, they make the sublist easier to read and edit.

The selection specification might look like this.

```
if <owed>=<*>
then process record
```

(The notation <*> means process the record if it has any value in the specified field. It is explained in detail in Chapter 5.)

Whatever document is identified for output will contain the header plus three records (Hartford, Mahoney, McDonald) after the List Processing Package has run.

4.1.11 Including Angle Brackets in Output

The List Processing Package outputs all text in a form until it encounters a < character. Then it stops the text output, discards the <, and expects to take some action depending on what follows (either field name>, >, !S>, !E>, or !R>). This means if you don't want the < discarded, you must tell the List Processing Package it should appear in your output text. To do this, precede the first < with an extra one. Then the List Processing Package will place the < and all information that follows it, up to and including the >, in the output.

Suppose you want to produce operating instructions for different models of the same piece of equipment, an electric broiler for example. And suppose the broiler has an ON button and a START switch. You would probably want one copy of the instructions to look like the following.

> Operating Instructions for Broiler Model #3276-B
>
> 1) Check that the <ON> button is lit up red.
> 2) Press the <START> switch.

The form used to produce the instructions should look like this.

> Operating Instructions for Broiler Model #<mod>
>
> 1) Check that the <<ON> button is lit up red.
> 2) Press the <<START> switch.

The extra < technique can also be used to change the spelling of a field name in a list document. Suppose you have so many records with <name>, <location>, and <telephone> fields that you are running out of space on your diskette. If you shortened the field names, you would gain space for expansion. To do this, create a form that looks like the following.

```
<<name><name>
<<loc><location>
<<fone><telephone>
<<>
```

Now run the List Processing Package with the old list and the above form. All records in the output document will contain field names <name>, <loc>, and <fone>.

CHAPTER 5
HOW TO CREATE A SELECTION SPECIFICATION DOCUMENT

A selection specification is the link between the records in the list document and the form document. It tells the List Processing Package which particular records are to be merged with the form and output. Figure 5-1 shows how a selection specification influences list processing output.

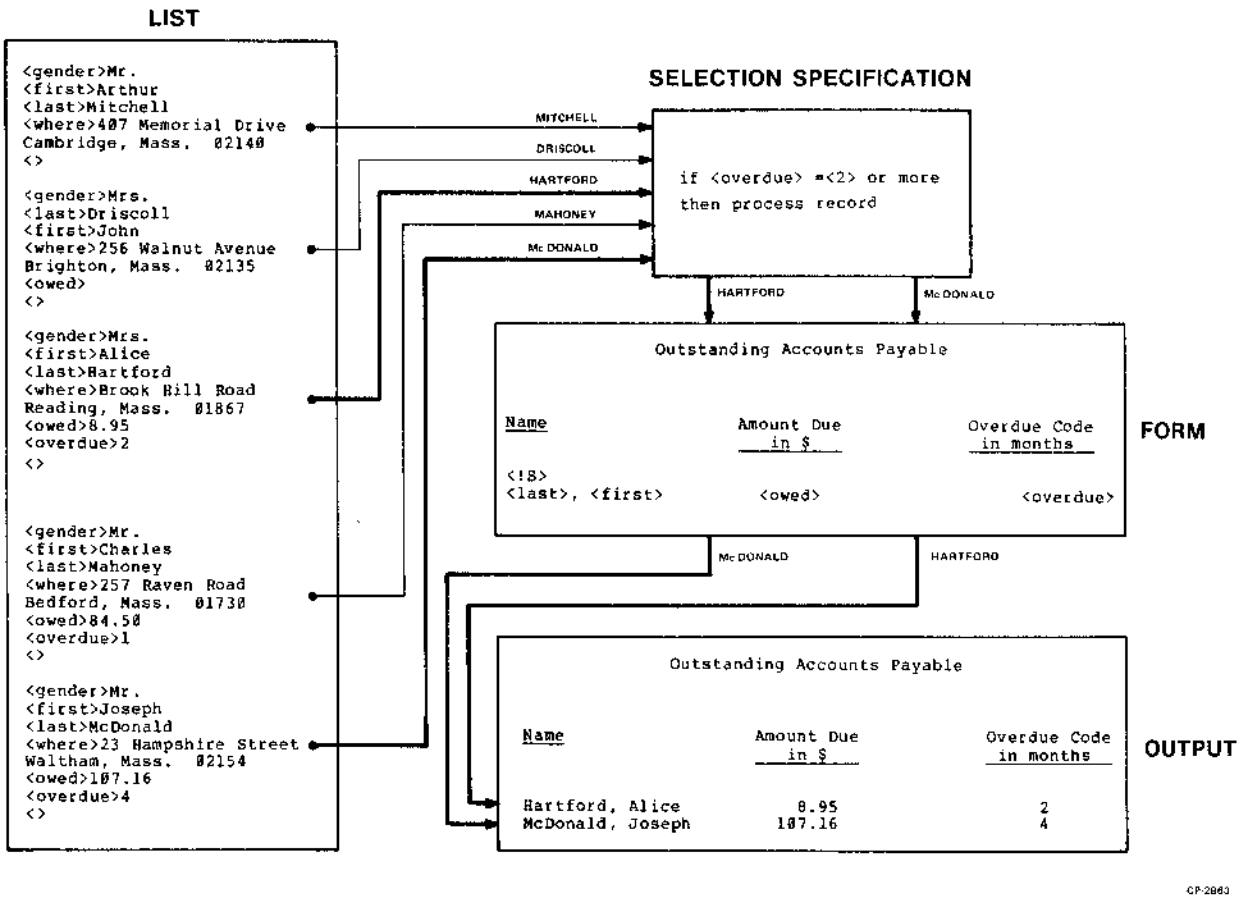


Figure 5-1 Role of Selection Specification in Producing Output

The most simple form of list processing is using all records and omitting none. The selection specification to merge all records looks like this.

```
process record
```

When the List Processing Package sees this basic selection specification, it does no testing. It simply merges each record with the form and outputs the result.

You can select some records for merging while bypassing others. This requires a selection specification that is written in the following format. You must supply your own information for words in boldface.

```
if <field name>=value
then process record
```

When the List Processing Package sees this selection specification, it tests each record. Only those records whose fields have the specified value are merged for output.

Suppose the records in your list contain a field called code, and code can have the possible values good and bad. If you want to process only those records whose code field is good, use this selection specification.

```
if <code>=good
then process record
```

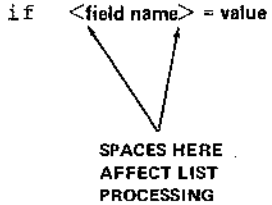
5.1 QUALIFICATIONS

5.1.1 Formatting

The words "then process record" must start on a new line.

5.1.2 Field Name Spelling

When a selection specification is applied to a record, the record is processed only if the field name is spelled exactly the same in the record and the selection specification. This means lowercase letters must match lowercase letters, uppercase letters must match uppercase letters, and spaces must match spaces.



For example, the following selection specification

```
if <last> =Smith
then process record
```

will examine these two records.

```
<last>Jones
<>

<last>Baker
<>
```

It will not cause either of the following records to be examined.

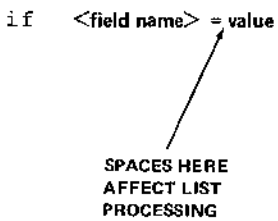
```
<Last>Smith
<>

< last>Adams
<>
```

The first record is skipped because the field called last starts with an uppercase L, the second because there is a space between < and last.

NOTE
Spell a selection specification field name exactly as it is spelled in the list records (considering spaces, uppercase letters, and lowercase letters).

5.1.3 Field Value Spelling

The value to be tested for should be spelled exactly the same in the selection specification as in the records. It should start right after the equal sign. This means that if you put a space between the equal sign and the value, the List Processing Package will test for a space followed by the value and not just the value itself.




```
then process record
```

The following selection specification will match only those records that have a space between <code> and good.

```
if <code>= good
then process record
```

It will not match a record that looks like

```
<code>good
```

because there is no space between the field name and its value. Similarly, it will not match

```
<code>    good
```

because more than one space is used.

NOTE

Use spaces between the equal sign and field value in a selection specification only if the spaces are to be tested for.

Spaces may, however, be safely used before the equal sign. These spaces will not affect list processing.

5.1.4 Size Restriction

A selection specification cannot be longer than approximately 20 lines. The maximum number of lines cannot be stated exactly as it depends on how long each one is.

5.1.5 Storage

Each selection specification must be placed in a separate document. If you should put two selection specifications in a document, the entire document will be worthless. This means you will not be able to use either specification for creating output.

An error message is displayed right after the start of the second selection specification when you try to run the List Processing Package with such a document.

5.2 KINDS OF TESTS

A selection specification can be used to:

1. Process every record.

2. Test whether a record has a field that matches a particular value(s).

   Example:
   ```
   if <month>=January
   or =February
   then process record
   ```

3. Test whether a record has several fields that match specified values.

   Example:
   ```
   if <month>=January
   and <day>=Monday
   then process record
   ```

4. Test whether a record meets several different conditions.

   Example:
   ```
   if <month>=January
   then process record
   or if <year>=bicentennial
   then process record
   ```

5. Test whether a record does not meet one or more specific conditions.

   Example:
   ```
   if <month>=January
   or =February
   but not if <month>=July
   then process record
   ```

5.2.1 Processing Every Record

Each record is merged with the form for output when the following selection specification is used.

```
process record
```

The List Processing Package does not test field(s) for value(s) when this selection specification is used.

5.2.2 Testing the Same Field

You can test the same field for several different values by using the word "or" to separate each possible value.

Format:
```
if <field> =value 1
or =value 2
      :
      :
      :
or =value last
then process record
```

Example:
```
if <state>=New Hampshire
or =N.H.
or =NH
then process record
```

To process all records belonging to New Hampshire residents, the above selection specification was designed to recognize the three most common spellings of the state name.

Each "or" test must start on a new line. If put on the same line as the previous test, the "or = value next" is considered part of the previous test. For example, the following selection specification tests the state field for value "N.H. or =NH."

```
if <state> =N.H. or =NH
then process record
```

5.2.3 Testing Several Fields

You can test several fields by using an "and" to separate the fields.

Format:

```
if <field 1> =value 1
and <field 2> =value 2
        :
        :
        :
and <field last> =value last
then process record
```

Example:

```
if <city>=Hudson
and <state>=New Hampshire
    or =N.H.
    or =NH
then process record
```

All records belonging to residents of Hudson, New Hampshire, and none belonging to residents of Hudson, Massachusetts, will be processed when the above selection specification is used.

5.2.4 Testing Several Conditions

You can test several conditions by using an "or" to separate the different "if tests."

Format:

```
if <field 1> =value 1
then process record
or if <field 2> =value 2
        :
        :
        :
or if <field last>=value last
then process record
```

Example:

```
if <last>=Smith
then process record
or if <maiden>=Smith
then process record
```

The above selection specification processes records for persons named Smith, now or in the past.

5.2.5 Excluding Conditions

You can exclude processing records with a specific condition by using a "but not" in front of an "if test."

Format:

```
if <field 1>=value 1
but not if <field 2>=value 2
then process record
```

Example:

```
if <country>=Canada
but not if <language>=English
then process record
```

Only those records that contain information for French-speaking Canadians will be processed when the above selection specification is used.

NOTE
Each "or," "and," and "but not" test must start on a new line.

5.3 TYPES OF VALUES THAT CAN BE TESTED

Three types of values can be tested for: a phrase, a number, a numeric range.

5.3.1 Testing for a Phrase

A phrase is a single letter (from A to Z), single special character (&, ', ", etc.), or any grouping of letters, special characters or digits (0, 1, ..., 9). A shorter way to describe a phrase is to say it is anything except a number (a grouping of digits only).

To test for a phrase, just put it on the right side of an equal sign in a selection specification. Most of the selection specifications shown in this manual have tested for phrases, as the following for example.

```
if <state>=New Hampshire
or =NH
or =N.H.
then process record
```

The complete state spelling and the two abbreviations are phrases.

You can indicate that *any character will be an acceptable match* by putting the characters <?> in the phrase being tested for.

```
Example:    if <word> =<?>ight
            then process record
```

A record whose word field contains night, a record whose word field contains right, and a record whose word field contains .ight will all be processed when the above selection specification is used.

```
Example:    if <part id>=MIL-<?>-1976
            then process record
```

A record whose part id is MIL-B-1976 will be processed as will one whose part id is MIL-5-1976 when the above selection specification is used.

You can indicate that a *particular number of any characters will be an acceptable match* by typing one question mark for each character to match in the phrase.

```
Example:    if <town> =<?????>boro
            then process record
```

A record containing data for the town of Northboro and one for the town of Southboro will both be processed, but a record containing data for the town of Westboro will not when the above selection specification is used.

You can indicate that *any number of any characters (including none) will be an acceptable match* by putting the characters <*> in the phrase being tested. (For those familiar with using other DIGITAL systems, this is the same as the wild card conventions.)

Example:
```
if <name> =<*>Jones
then process record
```

A record whose name field contains Mary Jones, one with John Paul Jones, and one with J. P. Jones will all be processed when the above selection specification is used.

In order that two records whose name fields contain J. P. Jones, Jr. and John Paul Jones, Sr. will be processed, the selection specification must be written like this.

```
if <name> =<*>Jones<*>
then process record
```

NOTE

No more than two sets of <*> characters may be used after an equal sign in a selection specification.

5.3.2 Testing for a Number

A number is one or more of the digits (0, 1, 2, ... 8, 9) written together. For example, your area code is a three-digit number, the year is a four-digit number, and six is a single-digit number. To test for a number in a selection specification, just put <> around it.

Example:
```
if <year>=<1776>
then process record
```

Any non-digits that appear inside the angle brackets are not really considered part of the number being tested for. This means that dollar signs, commas, decimal points, minus signs, and other special characters are ignored. Suppose a selection specification looks like this.

```
if <cost> =<$40,000>
then process record
```

A record that contains $40000, one which contains $40,000, one which contains 40000, and one which contains 40,000, will all be processed when the above selection specification is applied. However, a record with $400.00 will also be processed because, when the dollar sign and decimal point are removed, it contains the exact same digits as $40,000.

Example:
```
if <temperature> =<32>
then process record
```

With the previous selection specification, a record whose temperature field contains -32 will be processed just the same as a record with 32.

NOTE
The number being tested for cannot have more than 30 digits.

5.3.3 Testing For a Numeric Range

A field can be tested for being greater than a number, less than a number, or between two numbers by using one of the following formats.

Table 5-1 Format for Numeric Tests

Test	Format
Greater than	if <**field name**> = <**number**> or more then process record
Less than	if <**field name**> = <**number**> or less then process record
Between two numbers (1)	if <**field name**> = <**number 1**> through <**number 2**> then process record
Between two numbers (2)	if <**field name**> = <**number 1**> thru <**number 2**> then process record

Example:
if <balance>=<*>
and <overdue>=<2> or more
then process record

The above selection specification ensures that only customer accounts with active balances overdue at least two months will be processed.

NOTE
The <?> and <*> notations cannot be used to test for a number or a numeric range.

5.4 VERIFYING A SELECTION SPECIFICATION

In order to merge records with a form and produce output, your selection specification must be free of errors. You can verify a selection specification by calling up the List Processing Package, typing the letter T, and typing the name of the selection specification. If there are no errors, the following menu is displayed.

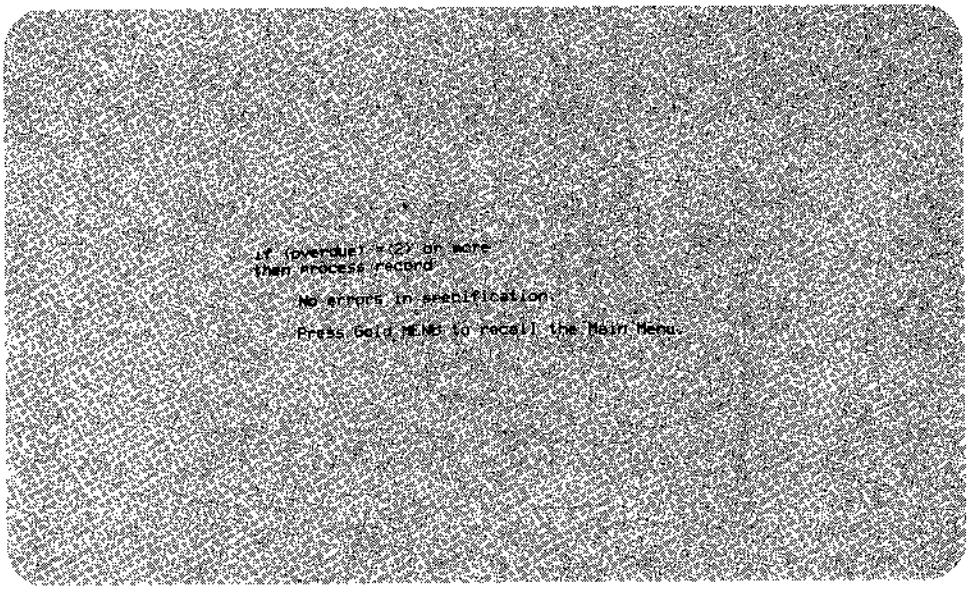


Figure 5-2 Screen View of a Successful Selection Test

If the selection specification contains an error, it is displayed on the screen with a ^ character under the first character of the first word that is incorrect. Here's what the screen looks like when a selection specification with an error is tested (the word record is misspelled as recard).

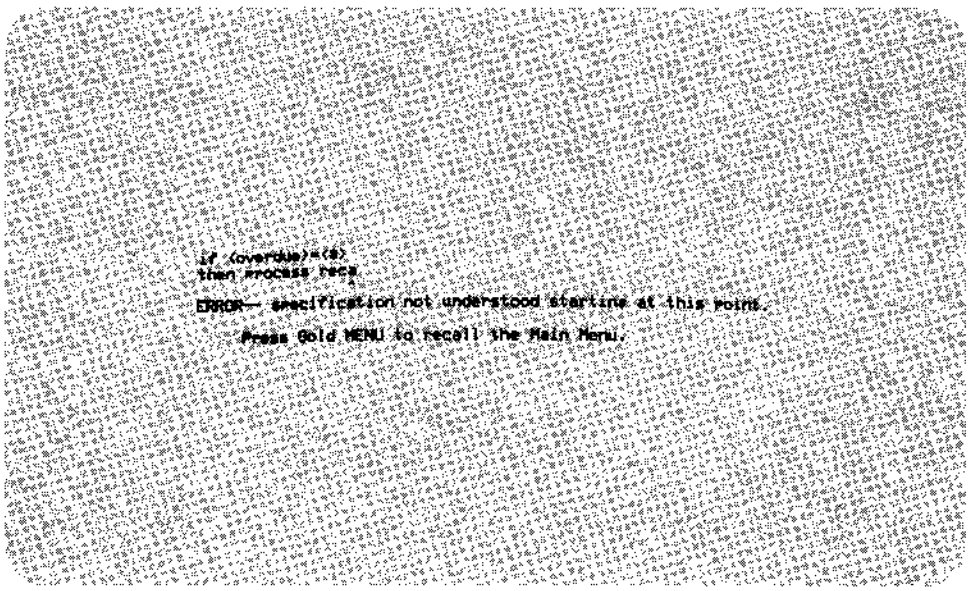


Figure 5-3 Screen View of an Unsuccessful Selection Specification Test

Correct the error and retest the selection specification.

NOTE

If you give the List Processing Package a selection specification that contains an error, no output will be produced (using option D or P). The selection specification will be displayed on the screen with a ^ character under the start of the error. An error message will also be displayed.

CHAPTER 6
HOW TO PRODUCE OUTPUT

6.1 RUNNING THE LIST PROCESSING PACKAGE

When you have prepared your list, form, and selection specification documents you are ready to run the List Processing Package to produce output. Follow these steps.

1. Call up the List Processing Package by typing LP (either upper or lowercase) at Main Menu level. The following menu will be displayed on the screen.

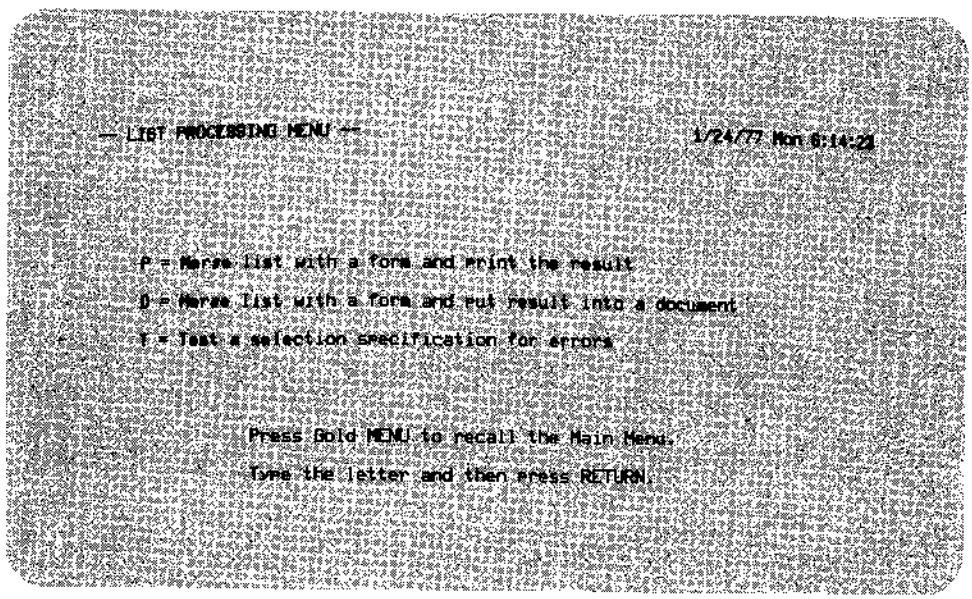


Figure 6-1 List Processing Menu

NOTE
Documents cannot be edited or printed when the List Processing Package is in use.

2. Select one of the three options by typing the appropriate character. If you want to print your output as it is produced and not save it on a diskette, type P. To save it for printing at a later time, type D. To test a selection specification, type T.

3. The system will now ask for the names of your list, selection specification, and form documents.

4. If you have selected option P, the Print Menu for the form document will now be displayed. You can make any changes to the printout format at this time. When ready, type YES to get back to the List Processing Package. Make certain that the printer has an adequate supply of paper.

5. If you have selected option D, the system will also ask for the name of the document where the output should be placed.

   If you specify a document that already exists, the following menu is displayed on the screen.

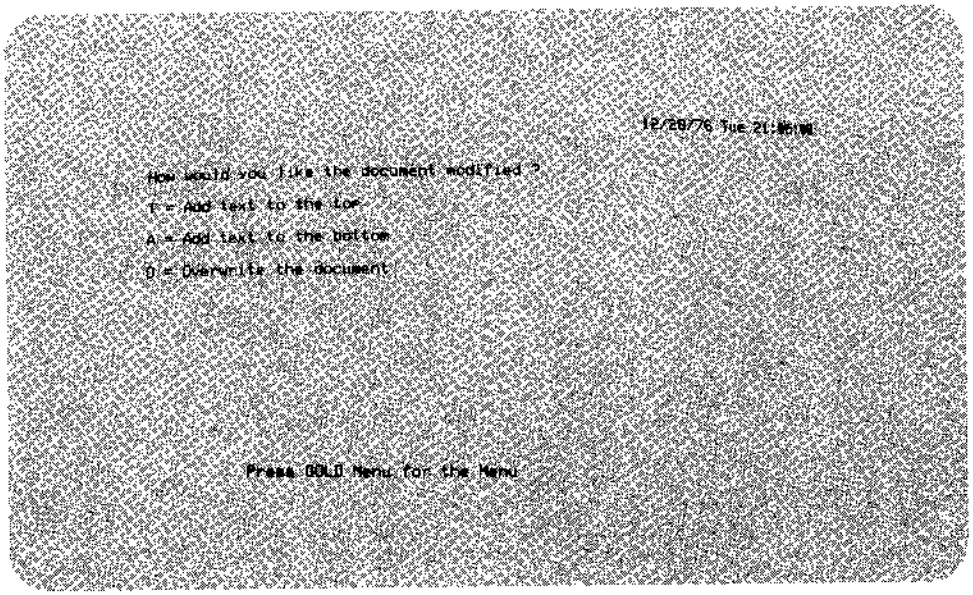


Figure 6-2 Menu for Adding Output to an Existing Document

You can add the List Processing output in front of what's currently in the document, after it, or you can overwrite it. If you want to use a different document, press the Gold and the MENU keys. The system will ask you for the name of the new output document.

If the document you specified does not exist, the system creates it.

6. Once all system prompts are answered, the List Processing Start Menu is displayed.

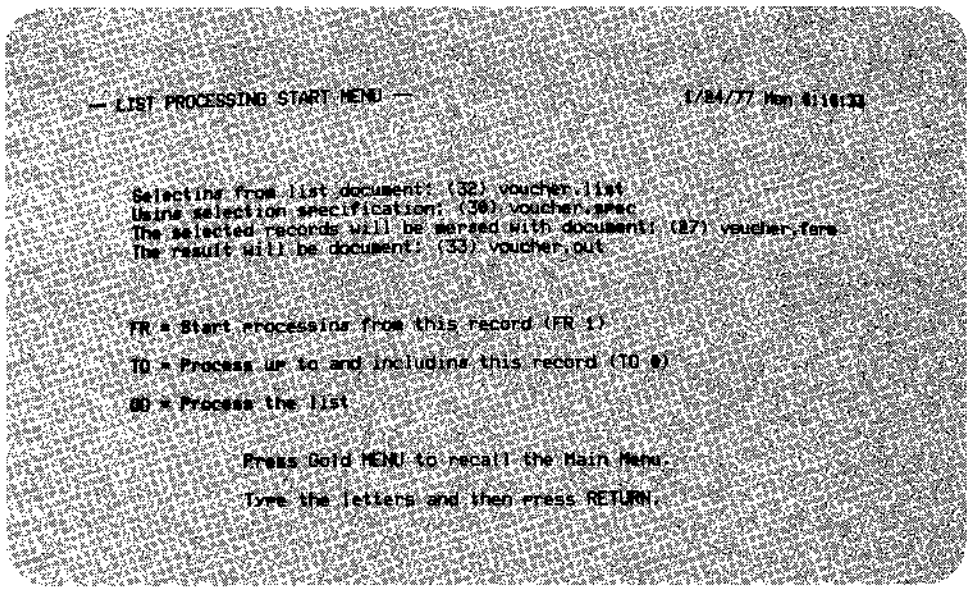


Figure 6-3 List Processing Start Menu

The value zero in the "TO" line means apply the selection specification to every record in the list document. If this is what you want, just type GO.

If you don't want to start at the beginning of the list document, type

FR **first record #**

and press RETURN. The value you give as **first record #** will be displayed in the List Processing Start Menu. If you don't want to go as far as the end of the list document, type

TO **last record #**

and press RETURN. The value you give as **last record #** will be displayed in the List Processing Start Menu.

NOTE
The number specified as either first record # or last record # must not be greater than 999.

Type GO when you are ready to run.

As the List Processing Package is running, it displays first the selection specification and then

Record being processed: record #

on the screen along with each merged form that is output. If your output is being printed as it is produced (option P), the actual printout will be a few records behind what is displayed on the screen.

When list processing is complete, the system tells you how many records were merged with the form for output and how many were examined. The screen display looks like this.

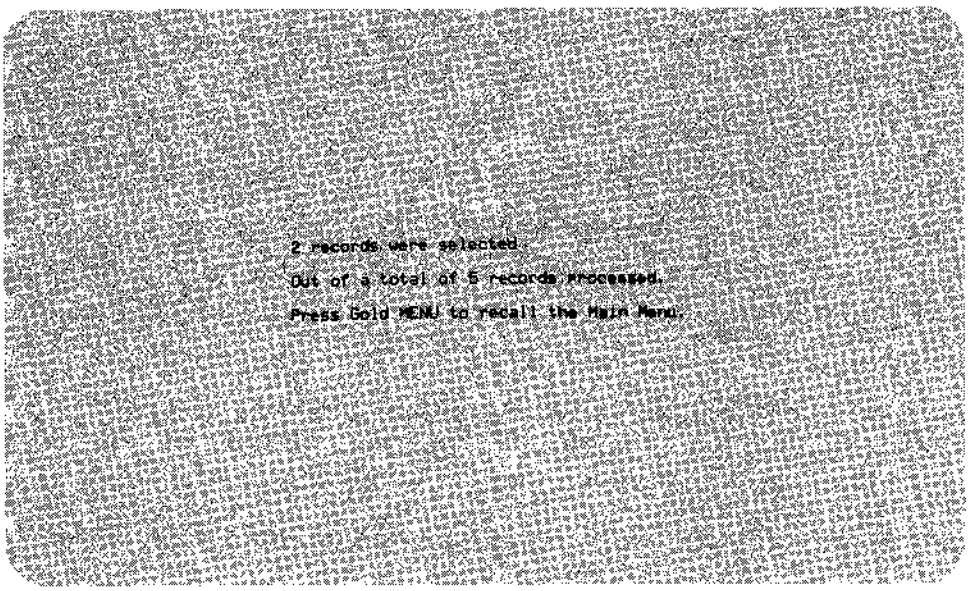


Figure 6-4 List Processing Completion Menu

Remember to pick up your printed output if any was produced.

6.2 STOPPING

If you want to stop the List Processing Package while it is running, press the Gold and the HALT keys. There may, however, be a slight delay before processing and/or output actually stops. Once list processing has halted, the number of records output and the number examined will be displayed on the screen. At this point there is no way to resume processing the list where you stopped. You must press the Gold and the MENU keys, type LP, and start over again (of course, you can tell the List Processing Start Menu to start with the last record processed when you halted).

6.3 SINGLE SHEET PRINTOUT

To print on single sheet paper, such as letterheads, do the following when the Print Menu for the form document is displayed.

1. Type NO and press RETURN.
2. Type PL and press RETURN.
3. Type SE YES and press RETURN.
4. Type OK and press RETURN.

When the List Processing Package goes to print the first piece of output, the information being displayed on the screen will come to a stop. Now follow these steps:

1. Load a single sheet of paper into the printer.
2. Type the letter R.

Each time the screen display stops, load another sheet and type the letter R again.

The printout lags behind what you see on the screen. This means that when the "--DOCUMENT FILING BEING COMPLETED--" message is displayed, there are still a few pages waiting to be printed.

NOTE

Continue to feed single sheets of paper and type R until the message giving the number of records processed and the number examined is displayed on the screen.

INDEX

WORD PROCESSING SYSTEM
LIST PROCESSING USER'S MANUAL
AA-5269B-TK

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? _____

What features are most useful? _____

What faults do you find with the manual? _____

Does this manual satisfy the need you think it was intended to satisfy? _____

Does it satisfy *your* needs? _____ Why? _____

Would you please indicate any factual errors you have found. _____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

Fold Here

Do Not Tear - Fold Here and Staple

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
Maynard, Massachusetts 01754