The purpose of a computer program or model documentation is to provide the details that will aid others to use the program, and, more importantly, aid those who desire to modify or revise the program. This pamphlet presents specifications for such documentation, first offering a complete outline of all the information that should appear. Each section of documentation—program identification and background, user documentation, and programmer documentation—is then described in detail. Examples are appended. (Author/SH)
Departmental Program and Model Documentation

PROGRAM AND MODEL DOCUMENTATION STANDARDS

Texas A&M University
The Texas Agricultural Experiment Station
H. O. Kunkel, Acting Director
AGRICULTURAL ECONOMICS PROGRAM
AND MODEL DOCUMENTATION

71-1

PROGRAM AND MODEL DOCUMENTATION
STANDARDS

by
Ray Billingsley
and
Stanley Wilson

February 1971
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Contents of Program and Model</td>
<td></td>
</tr>
<tr>
<td>Documentation</td>
<td>2</td>
</tr>
<tr>
<td>Documentation Elements</td>
<td></td>
</tr>
<tr>
<td>Part I PROGRAM IDENTIFICATION</td>
<td>3</td>
</tr>
<tr>
<td>Part II USER DOCUMENTATION</td>
<td>3</td>
</tr>
<tr>
<td>Part III PROGRAMMER DOCUMENTATION</td>
<td>4</td>
</tr>
<tr>
<td>Appendix A</td>
<td>7</td>
</tr>
<tr>
<td>Example of Program Documentation</td>
<td></td>
</tr>
<tr>
<td>Appendix B</td>
<td>15</td>
</tr>
<tr>
<td>Example of Model Documentation</td>
<td></td>
</tr>
</tbody>
</table>
INTRODUCTION

The purpose of a computer program or model documentation is to provide the details which will aid others to use the program but, more importantly, aid those who desire to modify or revise a program. Often a potential user cannot effectively utilize all the features of a program if he does not have an understanding of how the program operates. For more sophisticated programs, particularly those which simulate models of economic organizations and systems, any use at all is impossible without a substantial and detailed understanding of the program's algorithm, mathematical processes and assumptions. In any case, a simple explanation of where on the data card to enter the various values is seldom sufficient.

For these reasons, two types of documentation are useful. The first is for the person who wants to use the program as it exists. The second is for the person who would like to modify the program in order to meet his specific requirements or simply to improve the program. For these reasons, the program documentation specifications have been divided into USER DOCUMENTATION and PROGRAMMER DOCUMENTATION. Carefully written programs should also be generously endowed with comment cards which aid in program modification.

A number of programs or linear programming models have been written over the last several years which have not been documented. Substantial time, both professional and computer, has been expended to debug these programs and models, but after they were used for a particular project it was not possible for another person to use them again without virtually starting over. This results in a wasteful use of professional and computer time and incapacitates further use of the programs and models developed. Even if the existence of a similar program or model is known, its use may be impaired when the documentation does not exist or is so sketchy that modifying the existing program requires a great deal of effort. Often a program or model has wide possibilities for application by others and in some cases this may be more useful than its application to a particular problem.

The development of program and model documentation standards is intended to generate a class of uniform reports specifically devoted to program and model use and to provide a professionally recognized medium for displaying the work of those who expend efforts in building programs and computer simulations. The standards of documentation worked out in this document are designed so that the program or model can be understood both by the casual as well as the sophisticated user. Examples of program and model documentation are given in Appendices A and B respectively.

To achieve uniformity, an ordered listing of the material included in each documentation is required. This listing, called "CONTENTS OF PROGRAM AND MODEL DOCUMENTATION," is on the next page and should form the introduction and table of contents of the documentation. If a particular element in the documentation is not applicable, the author should state "none" after the element name in the body of the documentation and place "XXX" in the page reference space on the contents page.
CONTENTS OF PROGRAM AND MODEL DOCUMENTATION

I. PROGRAM IDENTIFICATION AND BACKGROUND

1. PROGRAM NAME

2. DOCUMENTATION NUMBER

3. AUTHOR OF DOCUMENTATION

4. PROGRAMMER

5. ORIGIN OF PROGRAM

6. LANGUAGE/COMPUTER

7. DATE

II. USER DOCUMENTATION

1. GENERAL ABSTRACT

2. OPERATIONAL USE

   Data Requirements/Options
   Input Specification/Format
   Jobstream

III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION

   Purpose
   Assumptions and Capabilities
   Algorithm
   Mathematical Description
   References

2. PROGRAM DESCRIPTION
   (Main and Subprograms)

   Program Abstract
   Variable List
   Subroutines/Functions called
   Flowchart

3. PROGRAM LISTING

4. SAMPLE PROGRAM RUN

   Input File/Sample Data
   Output File

* If a particular element is not contained in this documentation place XXX in this column.
I. PROGRAM IDENTIFICATION AND BACKGROUND

1. PROGRAM NAME
2. DOCUMENTATION NUMBER
3. AUTHOR OF DOCUMENTATION
4. PROGRAMMER
5. ORIGIN OF PROGRAM
6. LANGUAGE/COMPUTER
7. DATE

Each computer program or model should be given a different name. If the name is not indicative of the function of the program, a short statement should be included clarifying this. The documentation number identifies the year and the chronological order of the series for that year. The author of the documentation is usually a professional who initiates the program by having some process or model which he would like to see embodied in a program. He communicates this process or model in the form of an algorithm and/or flowchart to the programmer, who prepares the program, codes and debugs it. If the programmer produces part of the algorithm, he should be listed as co-author of the documentation. If an existing program written by another is modified, then the author should make reference to the original author and the original documentation, if known, in the section called "Origin of Program." If the program is original with the author, then the word "Original" should appear after the phrase "Origin of Program." If the author modifies his own program, he should make reference to the original program and documentation. The language refers to the computer language in which the program is written. When the documentation is separated from the program this information is not always obvious. The date refers to the date on which the program became operational.

II. USER DOCUMENTATION

1. GENERAL ABSTRACT: A short (not more than one page) description of the purpose of the program or model and how the program operates is required. This abstract should be directed to
professionals in the author's field and should indicate the functions which the program serves in that field. This abstract, to be suitable for indexing and use in information storage and retrieval systems, should be less than 200 words.

2. OPERATIONAL USE: The following information should be provided to facilitate use of a program or model.

Data requirements/options
Input specifications/format
Jobstream

A program may have several options and these options may determine which variables must be assigned values by inclusion as part of the data and which variables have their values calculated by the program. Data requirements, in turn, will determine the exact input specifications, that is the sequence in which the data cards are read in and whether a particular variable is assigned a numerical value in the input or is left blank. Frequently these specifications will consist of the data card formats alone. Particular attention should be given to situations in which disk files or tapes are to be utilized. The jobstream represents the sequence of control cards, program cards, data cards, etc. necessary to run the program.

III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION: The general description is intended to provide adequate information to convey what the program is intended to do and how it does it. References to more thorough treatments of the mathematical model than are included in the programming documentation are encouraged. The specification of the following items will constitute a reasonable general description of the program.

Purpose
Assumptions and capabilities
Explanation of algorithm
Mathematical description
References

The purpose consists of a short statement of what the program is supposed to do. This is followed by any special assumptions on which the model is based. These assumptions indicate any special capabilities or limitations of the program or model. The algorithm consists of an ordered series of steps by which the computer accomplishes the task or simulates the model.
Emphasis should be placed on the order in which the operations are carried out. Performing the task may depend more on the sequence in which the operations are done and less on each particular operation (which may be fairly simple). The contribution of the author may consist more in designing an algorithm than stating a new principle or a mathematical proof.

The mathematics of the program may be so simple or well known that this section is unnecessary. Alternately the program may be based on a mathematical proof or demonstration. In this case a clear, complete demonstration of the proof or reference to more thorough treatment of the mathematical model would be helpful.

The author should refer to any publications which will be useful to the reader or which can be used to go deeper into various aspects of the subject. If the program embodies techniques or principles discovered by another, the credit should be given to him. Both references to publications on techniques (such as books on linear programming) and subject area publications (such as books on economics) may be included. The form of reference should be standard bibliographical form and each entry should be assigned a number. In this way, referring to page 126 of the third entry in the bibliography, he can simply put [3, p. 126] in or after the sentence most closely associated with the reference. The references should certainly include any manuscripts prepared using this program or model.

2. PROGRAM DESCRIPTION: For the main program and each of the major sub-programs the following items are required:

   Abstract
   Variable list
   Subroutines and functions called
   Flowchart

The abstract should identify the purpose of the program. It should be directed to programmers and should indicate the role the main program and each sub-program plays in accomplishing the functions of the program or model. The variable list should include the names and interpretations of major variables used in the program. It is generally useful to distinguish between subscripted and simple variables. Listing the subroutines and functions called from the program generally provides useful reference information. A general flowchart of the logic of a program is useful. Simple, liberally commented descriptions may be preferable in some cases to detailed flowcharts. The flowchart should be coordinated with the listing in that the numbers of crucial statements in
the listing should appear at the corresponding point on the flowchart. A programme wishing to modify the original program can obtain a detailed flowchart by using a flowchart routine. Several are available.

3. PROGRAM LISTING: A reproduction of the computer listing is usually the best way to present the program listing. It not only eliminates the possibility of typographical error, but also conveys a better picture of the precise nature of the program.

4. SAMPLE PROGRAM RUN:

Input File/Sample Data
Output File

A sample run in which both the input file and the output file are listed may be useful. Additional annotation of these items may also be very useful. Many programs do not print out the input data before any results are calculated and/or printed. An image of this part of the input is often the easiest way to show the sample input data. This is particularly true if values of certain integers in the input determine which program options are used or how the input data is interpreted (for example which element in an array a given numerical value refers to). A sample output should be included when there is the possibility that it would be instructive to the user. A copy of the output pages alone may not be sufficient. Often some explanation of how to interpret various items in the output is vital. If the program does batch processing (more than one run of the program and introducing changes in a model's parameter or parameters), then a significant relationship may exist between a particular value in the input file and a particular value in the output file. For example, a given change in a certain model parameter may have a strong effect on a certain output value. If so, these relationships should be pointed out in describing the output file. Similarly, the formats of output files should be described carefully, particularly where information is sorted on tapes for future use.
APPENDIX A
CONTENTS OF PROGRAM AND MODEL DOCUMENTATION

I. PROGRAM IDENTIFICATION AND BACKGROUND

1. PROGRAM NAME .............. SEP-PRICE
2. DOCUMENTATION NUMBER ...... 70-0
3. AUTHOR OF DOCUMENTATION. Stanley Wilson and Ray Billingsley
4. PROGRAMMER .............. Stanley Wilson
5. ORIGIN OF PROGRAM ........ Original
6. LANGUAGE/COMPUTER ........ Fortran IV; IBM 360/65
7. DATE .............. October 15, 1970

II. USER DOCUMENTATION

1. GENERAL ABSTRACT .............. 9
2. OPERATIONAL USE .............. 9
   Data Requirements/Options ........ 9
   Input Specification/Format ........ 9
   Jobstream .............. 10

III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION .............. 10
   Purpose .............. 10
   Assumptions and Capabilities ........ 10
   Algorithm .............. XXX
   Mathematical Description ........ 11
   References .............. 11

2. PROGRAM DESCRIPTION
   (Main and Subprograms) .............. 11
   Program Abstract .............. 11
   Variable List .............. 11
   Subroutines/Functions called ........ XXX
   Flowchart .............. 12

3. PROGRAM LISTING .............. 13

4. SAMPLE PROGRAM RUN .............. 13
   Input File/Sample Data .............. 13
   Output File .............. 14

* If a particular element is not contained in this documentation place XXX in this column
II. USER DOCUMENTATION

1. GENERAL ABSTRACT

SEP PRICE is a computational device designed for use in NUPLEX planning and analysis. It generates a series of price and quantity changes given the price elasticity of demand and computes the SEP PRICE appropriate for each separable segment so that the average price received for the quantity of product sold is equal to the expected market price. SEP PRICE corresponds to the marginal revenue in the conventional imperfect competition model.

2. OPERATIONAL USE

Data Requirements: The program requires the elasticity of demand \( E \), a beginning quantity \( Q \), beginning average price \( P \), a change in quantity \( \text{DELQ} \), and the number of times \( Q \) is to be changed by \( \text{DELQ} \) \( N \).

Options: If the user wishes to move out along the demand curve, that is if the quantity is to be increased, then \( \text{DELQ} \) is entered with a plus sign or no sign. If the user wishes to move back along the demand curve, that is if the quantity is to be decreased, then \( \text{DELQ} \) is entered with a minus sign. In either case the program makes the sign of the change in Price the opposite of the sign of the change in Quantity.

Input Specifications/Format: The first card or cards in the input file are the description cards. The first four spaces on each description card should be left blank except for the last description card. On the last description card a "1" should be entered in Column 1. The user may have as many description cards as he wishes. After the description card comes the data card.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Spacing</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>( E )</td>
<td>1-7</td>
<td>V7.4</td>
</tr>
<tr>
<td>( \text{DELQ} )</td>
<td>10-24</td>
<td>F15.6</td>
</tr>
<tr>
<td>( Q )</td>
<td>25-44</td>
<td>F20.4</td>
</tr>
<tr>
<td>( P )</td>
<td>45-59</td>
<td>F15.4</td>
</tr>
<tr>
<td>( N )</td>
<td>78-80</td>
<td>I 3</td>
</tr>
</tbody>
</table>

Limitations: The program is limited to one elasticity value per run and one beginning quantity and price. Thus the user gets only one section of a demand curve on a run.
III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION

Purpose: The agricultural output of NUPLEX for any given crop could be such a large proportion of U.S. output that the output of NUPLEX would have a significant impact on the market price of the product and therefore on the average revenue. For that reason, sound economic analysis would indicate that this effect must be taken into consideration. This can be done using the separable programming feature of MPS/360.

The purpose of SEP-PRICE is to prepare input data for MPS/360 for separable programming. MPS/360 separable programming allows functions to be broken up into short ranges, and the slope of the line connecting the end points of each section is used to approximate the slope of the function over that section. The SEP-PRICE program breaks up the demand curve into small sections and calculates the SEP-PRICE appropriate for each separable segment so that the average price received for the quantity of product used is equal to the expected market price given the elasticity of demand, the size of the change in quantity, a beginning quantity and price.

Assumptions: The model assumes that each small section of the demand curve can be approximated accurately enough to be useful in deriving price changes from given changes in the quantity.

Algorithm: See Flowchart.
Mathematical Description: The formula used to calculate the change in price is:

\[ \text{DELP} = \frac{\text{DELQ} \times P}{Q \times E} \]

where,

- \( E \) = elasticity of demand,
- \( P \) = average price,
- \( Q \) = quantity, and
- \( \text{DELQ} \) = change in quantity,
- \( \text{DELP} \) = change in price

SEP-P is the separable price appropriate for each separable segment so that the average price received is equal to the expected market price.

References: (1) Mathematical Programming System/360, H-20-0476-1, IBM.

2. PROGRAM DESCRIPTION

Program Abstract: SEP-PRICE has no subroutines or functions and calls no system subroutines. The GENERAL ABSTRACT provides a sufficient abstract in this case.

Variable List: See mathematical description.

Subroutines/Functions called: None.

Flow Chart: See next page.
Flowchart

Start

Identity Card

Data, DELQ, Q, P, N

DELP
SEP-P
P

Card Image
DELP, Q, DELP
P, SEP-P

Stop
3. PROGRAM LISTING

```
DIMENSION DES(19), P(100)
WRITE(6,1,?)
120 FORMAT(1H1,2X,78H - PROBLEM IDENTIFICATION - )
   CONTINUE
READ(5,121) IFLAG, (DES(N),N = 1,19)
121 FORMAT(11,1X,10(A4))
WRITE(6,122) IFLAG, (DES(N),N = 1,19)
122 FORMAT(1X,11,2X,10(A4))
   IF( IFLAG .EQ. 0 ) GO TO 5
PEN(5,100) F,DELQ,0,P(I),N
100 FORMAT(F7.4,2X,F15.6,F20.4,F15.4,18X,13)
WRITE(6,123) F,DELQ,Q,P(I),N
123 FORMAT(1H0,F7.4,7X,F15.6,F20.4,F15.4,18X,13)
WRITE(6,115)
115 FORMAT(1H1,10X,5HDEL 0,22X,1HG,21X,5HDEL P,22X,1HP,14X,5HSEP-P)
   DO 10 I = 1,M
   ?I = I
10   M = N - 1
   SEPP = P(I) * P(I)
   DO 20 J = 1,M
   IF( (I - J) .LE. 0 ) GO TO 25
20   SEPP = SEPP - P(I - J)
25   CONTINUE
20   CONTINUE
   DELP =((DELQ * P(I)) / (Q * E)) * (-1.0)
WRITE(6,110) Q,P(I), DELQ, DELP, SEPP
27   Q = 0 + DELQ
28   P(I + 1) = P(I) + DELP
10   CONTINUE
```

4. SAMPLE PROGRAM RUN

Input File/Sample Data: E = 0.14, DELQ = 90, Q = 9000, P(I) = 4.65, N = 15 for this sample run. Data is not reproduced as part of the output. The description cards are reproduced but not shown here because their contents are self explanatory.
The above is an example of how the output looks. The first 9000 units can be sold at $4.65. If an additional 90 units are sold, the average price declines to $4.34. In order for the average price to be $4.34, the user of MPS/360 must have the first 9000 units sell for $4.65 and the additional 90 units sell for $4.03, the separable price. If the sales go up to 9180 units, then average price declines to $4.05. The first 9000 units must sell for $4.65, the next 90 sell for $4.03 and the next 90 sell for $3.17 in the input data for MPS/360.
CONTENTS OF PROGRAM AND MODEL DOCUMENTATION

I. PROGRAM IDENTIFICATION AND BACKGROUND

1. PROGRAM NAME . . . . . . . . . LP Farm Problem
2. DOCUMENTATION NUMBER . . 71-0
3. AUTHOR OF DOCUMENTATION. John Doe
4. PROGRAMMER . . . . . . Jane Doe
5. ORIGIN OF PROGRAM . . . Billy Freeman [1]
6. LANGUAGE/COMPUTER . . MPS/360; IBM 360/65
7. DATE . . . . . . . . January 1, 1971

II. USER DOCUMENTATION

1. GENERAL ABSTRACT . . . . . . . . . . . . . . . . . 17
2. OPERATIONAL USE . . . . . . . . . . . . . . . 17
   Data Requirements/Options . . . . . . . . . . . 17
   Input Specification/Format . . . . . . . . . . . 17
   Jobstream . . . . . . . . . . . . . . . . . . . . . 17

III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION . . . . . . . . . . . . . . 17
   Purpose . . . . . . . . . . . . . . . . . . . . . . 17
   Assumptions and Capabilities . . . . . . . . . . 17
   Algorithm . . . . . . . . . . . . . . . . . . . . 17
   Mathematical Description . . . . . . . . . . . 17
   References . . . . . . . . . . . . . . . . . . . . 17

2. PROGRAM DESCRIPTION
   (Main and Subprograms) . . . . . . . . . . . . . . 18
   Program Abstract . . . . . . . . . . . . . . . . . 18
   Variable List . . . . . . . . . . . . . . . . . . 18
   Subroutines/Functions called . . . . . . . . . . 18
   Flowchart . . . . . . . . . . . . . . . . . . . . . . . . XXX

3. PROGRAM LISTING . . . . . . . . . . . . . . . . . . . . . 19

4. SAMPLE PROGRAM RUN . . . . . . . . . . . . . . . . . . . 18
   Input File/Sample Data . . . . . . . . . . . . . . . . . 19
   Output File . . . . . . . . . . . . . . . . . . . . . . XXX

* If a particular element is not contained in this documentation place XXX in this column.
II. USER DOCUMENTATION

1. ABSTRACT: This model is designed for class room use to illustrate the basic elements of linear programming.

2. OPERATIONAL USE: Detailed information available in the MPS/360 Linear Program User's Manuals, [2].

III. PROGRAMMER DOCUMENTATION

1. GENERAL DESCRIPTION: The linear programming problem used here for illustration is based on a hypothetical farm situation. The objective is to maximize farm income.

   Purpose - To illustrate use of LP
   Assumption and Capabilities - Maximize farm income
   Algorithm - None

References


[2] "IBM Linear Program Users Manuals" (H20-0291) and (H20-0476-0).
2. MODEL DESCRIPTION

Abstract - See section II.1

Variable list -

<table>
<thead>
<tr>
<th>Row Name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJ</td>
<td>Objective Function 1</td>
</tr>
<tr>
<td>Obj</td>
<td>Objective Function 2</td>
</tr>
<tr>
<td>CHRow 1</td>
<td>Change Row</td>
</tr>
<tr>
<td>1CROP</td>
<td>Cropland</td>
</tr>
<tr>
<td>2PAST</td>
<td>Pasture</td>
</tr>
<tr>
<td>3CTAL</td>
<td>Cotton allotment</td>
</tr>
<tr>
<td>4FGB</td>
<td>Feed grain base</td>
</tr>
<tr>
<td>LBR 1</td>
<td>Labor 1</td>
</tr>
<tr>
<td>LBR 2</td>
<td>Labor 2</td>
</tr>
<tr>
<td>LBR 3</td>
<td>Labor 3</td>
</tr>
<tr>
<td>LBR 4</td>
<td>Labor 4</td>
</tr>
<tr>
<td>90TZ</td>
<td>Oats grazing transfer</td>
</tr>
<tr>
<td>100ST</td>
<td>Steer selling</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>10TG</td>
<td>Oats grain</td>
</tr>
<tr>
<td>20GF</td>
<td>Oats graze</td>
</tr>
<tr>
<td>3GSG</td>
<td>Grain sorghum</td>
</tr>
<tr>
<td>4COT</td>
<td>Cotton</td>
</tr>
<tr>
<td>5COW</td>
<td>Cow calf</td>
</tr>
<tr>
<td>6STS</td>
<td>Steers</td>
</tr>
<tr>
<td>7STS</td>
<td>Steers sell</td>
</tr>
<tr>
<td>8LAB</td>
<td>Labor Buying 1</td>
</tr>
<tr>
<td>9LAB</td>
<td>Labor Buying 2</td>
</tr>
<tr>
<td>10LAB</td>
<td>Labor Buying 3</td>
</tr>
<tr>
<td>11LAB</td>
<td>Labor Buying 4</td>
</tr>
</tbody>
</table>

Right Hand Side

PO1
PO2

Subroutine and Functions called - none

Flow chart - none

3. PROGRAM LISTING: See Page 19

4. SAMPLE PROGRAM RUN:

Input File/Sample Data: See page 20.

Output File: None
Control Program

PROGRAM
INITIALZ
MOVE(XDATA, 'MAX')
MOVE(XPBNAM,'PBFILE')
MOVE(XOBJ,'2OBJ')
MOVE(XRHS,'P01')
CONVERT('SUMMARY')
BCDOUT
SETUP('MAX')
PICTURE
TRANCOL
PRIMAL
SOLUTION
EXIT
PEND

These statements are necessary for each problem. MAX, 2OBJ, and P01 are arbitrary names which are peculiar to this problem only.

CONVERT checks the input data and converts it to PROBFILE.

BCDOUT causes data to be printed.

SETUP is necessary, MAX is optimal (refer to discussion of objective function).

Causes optimal solution to be computed.

Causes solution to be printed.

EXIT and PEND signals the end of program.
<table>
<thead>
<tr>
<th>NAME</th>
<th>MAX</th>
</tr>
</thead>
<tbody>
<tr>
<td>N OBJF</td>
<td></td>
</tr>
<tr>
<td>CHROW1</td>
<td></td>
</tr>
<tr>
<td>L 1CROP</td>
<td></td>
</tr>
<tr>
<td>L 2PAST</td>
<td></td>
</tr>
<tr>
<td>L 3CTAL</td>
<td></td>
</tr>
<tr>
<td>L 4FGB</td>
<td></td>
</tr>
<tr>
<td>L 5LBR</td>
<td></td>
</tr>
<tr>
<td>L 6LBR</td>
<td></td>
</tr>
<tr>
<td>L 7LBR</td>
<td></td>
</tr>
<tr>
<td>L 8LBR</td>
<td></td>
</tr>
<tr>
<td>L 9OTZ</td>
<td></td>
</tr>
<tr>
<td>L 10ST</td>
<td></td>
</tr>
<tr>
<td>N 2OBJ</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COLUMNS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1OTG</td>
</tr>
<tr>
<td>1OTG</td>
</tr>
<tr>
<td>1OTG</td>
</tr>
<tr>
<td>2OGZ</td>
</tr>
<tr>
<td>2OGZ</td>
</tr>
<tr>
<td>2OGZ</td>
</tr>
<tr>
<td>3GSG</td>
</tr>
<tr>
<td>3GSG</td>
</tr>
<tr>
<td>3GSG</td>
</tr>
<tr>
<td>3GSG</td>
</tr>
<tr>
<td>4GOT</td>
</tr>
<tr>
<td>4GOT</td>
</tr>
<tr>
<td>4GOT</td>
</tr>
<tr>
<td>4GOT</td>
</tr>
<tr>
<td>5GOW</td>
</tr>
<tr>
<td>5GOW</td>
</tr>
<tr>
<td>5GOW</td>
</tr>
<tr>
<td>5GOW</td>
</tr>
<tr>
<td>6STR</td>
</tr>
<tr>
<td>6STR</td>
</tr>
<tr>
<td>6STR</td>
</tr>
<tr>
<td>7STS</td>
</tr>
<tr>
<td>7STS</td>
</tr>
<tr>
<td>8LAB</td>
</tr>
<tr>
<td>8LAB</td>
</tr>
<tr>
<td>9LAB</td>
</tr>
<tr>
<td>9LAB</td>
</tr>
<tr>
<td>10LA</td>
</tr>
<tr>
<td>10LA</td>
</tr>
<tr>
<td>11LA</td>
</tr>
<tr>
<td>11LA</td>
</tr>
</tbody>
</table>

RHS

| P01  | 1CROP 150.00000 | 2PAST 50.00000 |
| P01  | 3CTAL 50.00000  | 4FGB 50.00000  |
| P01  | 5LBR 1030.00000 | 6LBR 780.00000 |
| P01  | 7LBR 530.00000  | 8LBR 780.00000 |
| P02  | 1CROP 75.00000  | 2PAST 25.00000 |
| P02  | 3CTAL 25.00000  | 4FGB 25.00000  |
| P02  | 5LBR 25.75000  | 6LBR -19.50000 |
| P02  | 7LBR 13.25000  | 8LBR -19.50000 |

ENDATA