

THE FREQUENCY OF OCCURRENCE OF
SYNTAX ERRORS FOR BEGINNING PROGRAMMING STUDENTS

by

LINDA MARIE RUST

B. S., Kansas State University, 1968

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

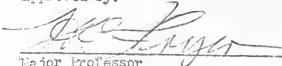
MASTER OF SCIENCE

Department of Statistics and Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1969

Approved by:


Major Professor

LD
2668
R4
1969
R68

TABLE OF CONTENTS

c. 2 CHAPTER

	PAGE
I. INTRODUCTION	1
Definition of Terms	2
Statement of the Problem	3
Importance of the Study	3
Objectives of the Study	3
Description of the Population	4
Chi-Square	4
II. METHOD OF RESEARCH	5
III. RESULTS OF THE STUDY	6
Frequencies and Percentages	6
Generalized Means Statistics	14
Chi-Square Tests of Significance	15
IV. SUMMARY	21
V. CONCLUSION	22
VI. RECOMMENDATIONS	23
BIBLIOGRAPHY	24
APPENDICES	
Appendix A. Questionnaire Completed by Students for Each Laboratory Problem	26
Appendix B. Listing of the Errors that Occurred, Their Frequency and Explanation	27

LIST OF TABLES

TABLE	PAGE
I. Percent of students by major and percent of errors by students.	7
II. Listing of the thirteen most frequently occurring single errors.	8
III. List of textbook topics, number of pages per topic, and number of errors related to topic.	11
IV. Listing of error types for five problems according to frequency of occurrence.	13
V. Mean number of runs for each laboratory problem within each instructor.	15
VI. Mean number of runs for each laboratory problem within each classification.	16
VII. Chi Square values for five variables tested against single errors.	17
VIII. Chi Square values for five variables tested against types of errors.	20

CHAPTER I

INTRODUCTION

In recent years technology has increased almost exponentially. Fundamental to the progress of our advancing technology is the capability to acquire, process, and interpret information, or data, and to make logical decisions based on such interpretations. The device which contributes largely to the rapid processing of data is the electronic digital computer.

This advancing technology requires highly skilled and well educated personnel to fill today's existing positions and tomorrow's new and diverse positions. Our educational system has been extremely influenced by the applications of digital computers.

Computer usage in education can be divided into four categories. The first category is computer-assisted instruction which Fishman et al. (1968) refer to as an "instructional procedure which utilizes a computer to control part, or all of the selection, sequencing, and evaluation of instructional materials." The second category is computer application in administration. Silvern and Silvern (1966) include class scheduling, attendance recording, record accounting, test scoring and analysis, and report generating in this area.

Thirdly, computers are used as a tool in conjunction with other courses, the most obvious ones being engineering, the sciences, mathematics, and business. The fourth category of computer usage is that the computer can be a subject in itself, involving uses, languages, programming, logic, and theory.

It was this last category, the computer as a subject itself, specifically programming, that was of major interest in this paper.

Chapman and Carpenter (1962) list eight principles of teaching which have evolved over the centuries. One of these is that misunderstandings should be detected and corrected immediately. It is obvious that when one is learning to program, mistakes or errors will be made. Beforehand knowledge as to what types of errors will be made and what factors influence the making of these errors could be of great benefit to those that teach programmers.

Such knowledge could improve the efficiency of learning to program the computer, an important part of our advancing technology. As stated by Goodman (1962), to simplify programming is one of the important tasks before us today in the field of computation, for unless ordinary people can use computers, and understand a fair amount about them, the contribution these machines can make to society will only be partially appreciated and partially realized.

I. DEFINITION OF TERMS

WATFOR Compiler. The WATFOR (from WATERLOO FORTRAN) compiler for the System 360 is an in-core, one-pass, load-and-go processor which accepts FORTRAN IV programs to be compiled and executed.

Error. An error was a diagnostic message code printed out by the WATFOR/360 compiler as listed in Appendix B of Blatt (1968).

Run Number. A run number was the sequential, student assigned number for each separate run of a specific problem.

Type of Error. Errors were grouped into the same categories or types as listed by Blatt. For example, DO LOOPS and DIMENSION statements were

two types of errors.

Classification. Classification referred to the school year of the students. Classifications were freshman, sophomore, junior, senior, and graduate students.

Category. A category was a subclass of a variable and was used in conjunction with a chi-square program. For example, the variable sex had two subclasses or two categories, male and female.

II. STATEMENT OF THE PROBLEM

There were two purposes for this study. The first purpose was to determine what kinds of errors were made by beginning programming students. The second purpose was to determine what effect, if any, the following factors had upon the errors that were made by students: (1) instructor, (2) laboratory problem number, (3) sex, (4) major, and (5) classification.

III. IMPORTANCE OF THE STUDY

To be able to predict what types of errors beginning programming students were most likely to make would be beneficial not only to educators, whether in the formal school system or in industry, but also to those developing the instructional material. More emphasis could be given to the areas that are producing the most errors, in hopes of significantly reducing their occurrence.

IV. OBJECTIVES OF THE STUDY

The objectives of this study stated as null hypotheses were:

- 1) Errors made by beginning programming students were independent of instructor.
- 2) Errors made by beginning programming students were independent

of laboratory problem.

3) Errors made by beginning programming students were independent of major.

4) Errors made by beginning programming students were independent of classification.

5) Errors made by beginning programming students were independent of sex.

V. DESCRIPTION OF THE SAMPLE

During the spring semester of the 1968-1969 school year a total of 194 students were enrolled in the course Fundamentals of Computer Programming at Kansas State University. These students were randomly assigned into six classes, taught by five instructors.

Four of these classes, each taught by a different instructor, were used for this study. The subjects for this study consisted of seventy-three males and eleven females.

VI. CHI-SQUARE

The chi-square distribution provides a means of comparing number of observed sampling occurrences with theoretical frequencies of occurrence of these same events under some hypotheses for k categories. The statistic is

$$\chi^2 = \sum_{i=1}^k \frac{(f_i - F_i)^2}{F_i}$$

where f_i represents the observed frequencies and F_i is some theoretical frequency. Chi-square critical values were obtained from Concepts and Methods of Experimental Statistics by Fryer (1966).

CHAPTER II
METHOD OF RESEARCH

All four instructors used the same textbook, Introduction to FORTRAN IV Programming: Using the WATFOR Compiler by Blatt. Four identical laboratory problems were assigned to each class on the same day. Problems were due from the students in two weeks. Two of the classes were also assigned a fifth problem. The results from these two classes are also included in this study.

Each student was required by his instructor to complete a questionnaire (Appendix A) for each laboratory problem. All Watfor message codes were to be listed according to sequential run number for each laboratory problem. The student was asked to check the appropriate category if he believed that the error occurred as a result of one of the following types: key punch error, 'dimension' statement, 'format!', data, use of 'common', 'logical if', 'arithmetic if', and 'go to' statement.

A program was written in FORTRAN to total the results of the error sheets. Chi-square statistics were gathered by using the Kansas State University Statistical Library Chi-square program. Mean statistics were provided by the Kansas State University Statistical Library Generalized Means program (1968).

CHAPTER III
RESULTS OF THE STUDY
FREQUENCIES AND PERCENTAGES

A total of 2,148 errors were recorded from the students in the four classes, 1,957 from male students, 191 from female students. Males accounted for slightly more errors than females. Males represented 87% of the total number of students and committed 91% of the total errors. Figures for the females were 13% and 9% respectively.

Class size ranged from sixteen to twenty-seven. Class one had twenty-seven students, twenty-five males, and two females. Class two had sixteen students with only one female. Class three had twenty-one students, sixteen males and five females. Class four had twenty with seventeen males and three females.

The students represented five classifications, freshman to graduate students. Freshmen were 20% of the total number of students and made 22% of all errors. Sophomores were 32% of the students but made only 27% of the errors. Juniors were 24% but made 27% of the errors. Graduate students were 4% and 3% respectively.

A total of twenty-eight majors were represented by these eighty-four students. These majors were easily grouped into six major categories. Percentage of total number of students for each major category and percentage of errors made by those students are listed in Table I.

TABLE I
PERCENT OF STUDENTS BY MAJORS AND PERCENT OF ERRORS BY STUDENTS

Major	% of Students by major	% of Errors by students
Business Administration and Accounting	22%	23%
Engineering	18%	19%
Mathematics	18%	16%
Social Science	18%	17%
Physical Sciences	14%	13%
Agriculture	10%	12%

There were 281 possible errors that could have occurred. One-half or 140 different errors did appear at least once. Of these, seventy-eight occurred five times.

The most frequently occurring error was NONE, or no error occurred. This was for two reasons. The first reason was that many students were able to successfully complete the first assigned problem and some the second problem on the first run. Therefore no errors were made. Since this was a beginning programming course, the first problem was deliberately designed to be easy. Its main purpose was to familiarize the students with the key-punch machines, procedure of assembling a card deck for running, etc.

The second reason is that in order to obtain the average number of runs for each laboratory problem, the error NONE was entered if no WATFOR error occurred. Many times it was necessary to rerun the problem because of job control errors, wrong answers, output spacing

problems, etc. These constituted valid runs; an error had to be recorded.

Thirteen out of a possible 281 errors accounted for 50% of the total 2,148 errors made. A listing, frequency and brief description of these thirteen errors appears in Table II. A complete listing of all errors that occurred along with their frequency, complete explanation and type grouping is found in Appendix B.

The error KO-0 meant that some error occurred during compile time. Even though WATFOR attempted execution after detecting errors during compilation, the erroneous source language statement could not be executed.

TABLE II

LISTING OF THE THIRTEEN MOST FREQUENTLY OCCURRING SINGLE ERRORS

Error Code	Frequency	Explanation
NONE	240	No error occurred.
KO-0	103	Compile time error.
SX-0	102	Missing operator.
PC-0	96	Unmatched parentheses.
SS-1	91	Subscript out of range.
UN-0	68	Control card encountered on card reader during execution.
ST-5	65	Undecodeable statement.
UV-2	61	Undefined variable-array number.
IF-3	54	Arithmetic or invalid expression in logical IF.
UV-0	52	Undefined variable-simple variable.
DO-5	51	Invalid DO-LOOP parameter.
SV-1	45	Array name or subprogramme name used without list.
EQ-8	44	Illegal use of equal sign.

To receive a SK-0 error, an operator such as '+', '-', '*', '/', '**', was left out of an expression. For example 'A=BC' could have been key punched instead of 'A=B+C'.

Since parentheses must be matched in FORTRAN programming, failure to do this would result in a PC-0 error. This error suggested that matching of parentheses should be heavily stressed when instructing a FORTRAN course.

The SS-1 error meant that a subscript exceeded the number of a corresponding DIMENSION statement. Either students are not sure how to use a DIMENSION statement or they do not fully understand the relationship of subscripted variables to the DIMENSION statement.

If a control card was encountered (usually the end-of-job card) when attempting to read a data card, a UN-0 error occurred. Special programming techniques are needed to avoid making the above error. If these techniques were taught earlier in a beginning programming course, perhaps the occurrence of the UN-0 error could be reduced.

An undecodable statement error of ST-5 can mean anything. This error would be extremely difficult to reduce. For example, if the statement 'IF (A.GT.B) THEN A=B' was used this error would result. The word 'THEN' has no valid meaning in FORTRAN. The statement should have been 'IF (A.GT.B) A=B'.

The errors UV-2 and UV-0 are directly related. One has attempted to use a variable on the right of an equal sign before it has been assigned a value by appearing to the left of an equal sign. By combining the frequency of these two errors, this became the second most common error. More instruction might help to reduce these errors.

The errors IF-3 and EQ8 usually resulted for the same reason. To get the first error, an arithmetic expression appeared in a LOGICAL IF statement. The EQ-8 error resulted from illegal usage of an equal sign. For example, if the expression 'IF (A=B) S' was used the above errors would have occurred. The expression should have been 'IF (A.EQ.B) S!'. These errors were probably the result of a student not fully understanding the correct usage of arithmetic and logical operators.

DO-LOOP's are commonly used in FORTRAN programs. The DO-5 errors were the result of invalid usage of the DO-LOOP statement.

The SV-1 error resulted when either a subprogramme name appeared without legal use of its argument list or an array name appeared without its subscripts. If the error resulted because of the first reason, students probably need more explanation on how to use subprogrammes. If the error resulted because of the latter reason this meant that students did not understand the usage of subscripts. This closely tied in with the SS-1 error described earlier.

Since all students were instructed from the same textbook a few comments need to be made concerning the emphasis the author placed upon the instructional material. A list of topics along with the number of pages devoted to each topic can be found in Table III. The number of errors directly related to each topic is also given in this table.

Material concerning the topic 'Arithmetic and Logical Statements' was covered in eleven pages. However, more errors were directly related to this topic than to any other. A total of 537 errors resulted from arithmetic and logical statement usage.

Two other topics were explained briefly by the textbook and resulted in large numbers of errors. These were 'Card Deck and Control Cards', accounting for five pages and 118 errors, and 'Printed Output-Diagnostics and Termination', accounting for four pages and 130 errors. Because so many errors resulted from these three above mentioned topics, more emphasis should be given to these when teaching a beginning programming course.

On the other hand, perhaps less emphasis could be given to the topic 'FORTRAN Arithmetic'. Twelve pages were used to explain this topic but only thirty-eight related errors were made. Another topic

TABLE III

LIST OF TEXTBOOK TOPICS, NUMBER OF PAGES PER TOPIC,
AND NUMBER OF ERRORS RELATED TO TOPIC

TOPIC	PAGES	ERRORS
Arithmetic and Logical Statements	11	537
Go To Statements	2	6
Arithmetic and Logical IF	6	88
INPUT/OUTPUT	14	209
Card Deck and Control Cards	5	118
Printed Output-Diagnostics and Termination	4	130
FORTRAN Arithmetic	12	38
DO Statement	8	120
Arrays	29	301
Equivalence and Common	6	7
Format	16	245
Type Declaration and Initialization	10	55
Built-In and Arithmetic Statement Functions	10	19
Function Subprogram	16	21
Subprogrammes	7	34

needing less emphasis might be 'Type Declaration and Initialization'. Ten pages of textbook material and fifty-five errors were the statistics for this topic.

With reference to Table III, the first laboratory problem was assigned after the topic 'Printed Output' had been explained. This problem covered the usage of Logical IF statements. The second problem was assigned after the topic 'DO Statement'. It dealt with DO-LOOP usage.

The third laboratory problem, which covered Input/Output was assigned after the topic 'Format'. Following the topic 'Built-In and Arithmetic Statement Function' the fourth problem on functions was assigned. The fifth problem on subroutines was assigned after the topic 'Subprogrammes'.

It was expected that occurrence of errors would be closely related to the laboratory problem assigned and the material each covered. For example, errors that occurred in the first and second laboratory problems should not occur as often in the remaining problems, because students should have gained experience in how to correctly use the topics covered by this material.

However this was not the case for the following five types of errors: statements and statement numbers, syntax errors, job termination, equal signs, and card format and contents. Types of errors for each of the laboratory problems are listed according to total frequency of occurrence in Table IV. All of the five types of errors mentioned above were directly related to the topics that received a small amount of coverage in the textbook.

TABLE IV
 LISTING OF ERROR TYPES FOR FIVE PROBLEMS
 ACCORDING TO FREQUENCY OF OCCURRENCE

Error Type	Prob. One	Prob. Two	Prob. Three	Prob. Four	Prob. Five	Total Frequency
NONE	61	60	62	37	20	240
Statements and Statement Numbers	24	20	54	53	18	169
Syntax Errors	48	9	63	29	15	164
Job Termination	12	13	46	43	16	130
INPUT/OUTPUT	7	5	46	43	27	120
DO-LOOPS	0	6	43	44	27	120
Undefined Variables	7	14	40	45	9	115
Parentheses	3	4	71	22	15	115
Format	13	3	34	17	44	111
Subscripted Variables	0	5	76	7	17	105
Subscripts	0	5	35	50	7	97
Equal Signs	16	11	41	22	6	96
Card Format and Contents	23	12	33	12	10	90
I/O Operations	9	4	33	29	14	89
IF Statements	29	5	22	28	4	88
Variable Names	7	7	21	10	10	55
Data Statements	0	2	3	30	14	49
Subprogrammes	2	1	11	3	17	34
End Statement	7	16	0	1	3	27
Hollerith Constants	0	0	1	21	2	24
Functions and Subroutines	0	3	8	5	5	21
FORTRAN Type Constants	5	3	7	3	1	19
Powers and Exponentiation	0	0	14	0	2	16
Arithmetic & Logical St. Functions	0	4	5	3	1	13
DIMENSION Statements	0	1	8	2	1	12
Library Routines	0	0	6	0	0	6
Implicit Statement	0	0	2	4	0	6
GO TO Statements	2	0	0	3	1	6
Equivalence and/or Common	0	0	1	0	3	4
Mixed Mode	1	0	1	1	0	3
Common	0	0	0	0	2	2
Equivalence Statements	0	0	1	0	0	1
Job Control Cards	1	0	0	0	0	1
TOTAL	277	213	788	559	311	2,148

As for types of errors, statements and statement numbers was the most frequently occurring type of error. Statements were either undecodeable or missing. Statement numbers were either missing, misplaced, or multiply-defined. Syntax errors, the second most frequently abundant type of error, were directly related to the first group. Syntax errors resulted when either an operator, constant, symbol, or statement member was missing or illegally used. If the amount of time specified for the program was exceeded or an error occurred during compilation, job termination errors occurred.

In general, variables caused the most errors. Either variables were undefined, improperly named or incorrectly subscripted. A combined total of 376 errors resulted from variables.

GENERALIZED MEANS STATISTICS

In order to obtain the average number of runs for each lab problem the Statistical Library's Generalized Means Program (1968) was used. The average number of runs for each instructor per laboratory problem is shown in Table V.

Laboratory problems one and two required the least number of runs for all instructors, approximately two runs. Problems three and four required the most number of runs for instructors, about five.

Only the runs for instructor one increased linearly. The others, especially the runs for instructor four, fluctuated. Instructor one had the highest average number of runs, followed by instructor two, then four, then three.

TABLE V

MEAN NUMBER OF RUNS FOR EACH LABORATORY PROBLEM WITHIN EACH INSTRUCTOR

Problem Number	Mean for Instructor 1	Mean for Instructor 2	Mean for Instructor 3	Mean for Instructor 4
1	2.33	3.07	1.76	2.40
2	2.96	2.21	1.53	1.40
3	5.50	5.64	5.33	6.22
4	6.12	5.43	5.77	3.53
Weighted Average	4.61	4.08	3.26	3.31

In Table VI, the average number of runs for each laboratory problem according to classification are shown. For all problems but the second, either juniors or seniors had the least number of runs. For all problems but the second, graduate students had the most number of runs. Both freshmen and graduate students required about two more runs for problem three than the other classes.

In increasing order, classifications, according to the average total number of runs are as follows: juniors, seniors, sophomores, freshmen, graduate students. However, the variations between these averages were small.

CHI-SQUARE TESTS OF SIGNIFICANCE

One of the purposes of this paper was to determine if the following five variables had any significant effect upon the errors that were made by beginning programming students: instructor, laboratory problem, major, classification, and sex. A chi-square test was used to determine if any dependencies existed.

TABLE VI
 MEAN NUMBER OF RUNS FOR EACH LABORATORY PROBLEM
 WITHIN EACH CLASSIFICATION

Problem Number	Mean for Freshmen	Mean for Sophomores	Mean for Juniors	Mean for Seniors	Mean for Graduates
1	2.19	2.56	2.16	2.00	4.00
2	1.93	2.43	1.85	2.33	1.33
3	7.43	5.53	4.53	5.31	7.50
4	4.75	6.00	5.39	5.00	6.50
Weighted Average	4.18	4.15	3.59	3.76	4.40

For the first chi-square test errors were sorted according to frequency of occurrence. The chi-square program was then run upon eleven categories of errors (the program used was limited to eleven categories at one time). Only those errors that had a frequency of occurrence of five or greater were used in order to obtain category sizes large enough to give meaningful statistics for the chi-square program.

The chi-square statistics for each of the five variables tested against occurrence of errors are given in Table VII. Values for degrees of freedom and chi-square critical values needed for significance at the .05% level are also listed. Eleven categories, beginning with the most frequently occurring error, were included in each group number. That is, group 1 consisted of the eleven most frequent errors, group 2 consisted of the next eleven most frequent errors, etc.

TABLE VII

CHI-SQUARE VALUES FOR FIVE VARIABLES TESTED AGAINST SINGLE ERRORS

Categories in Groups of 11	Instructor	Lab. Problem	Major	Classifi- cation	Sex
Group 1	75.32*	268.62*	79.28*	61.86*	11.72
Group 2	79.68*	268.59*	62.30	71.54*	12.84
Group 3	59.55*	139.26*	49.42	47.42	9.67
Group 4	65.92*	84.08*	68.94*	51.92	13.00
Group 5	57.27*	77.59*	45.96	31.59	9.33
Group 6	44.83*	98.32*	83.12*	63.21*	6.18
Group 7	42.92	70.37*	55.91	45.39	7.98
Degrees of Freedom	30	40	50	40	10
Chi-Square Value for Significance	43.77	55.76	67.93	55.76	18.31

*indicates significance at .05% level

Upon examination of the chi-square values it was found that the variable instructor had an effect upon the first six groups of error categories but no effect upon the last group. The variable laboratory problem significantly affected all groups of error categories.

Groups of error categories one, four and six were affected by the variable major whereas the other groups were not. The variable classification also affected only three groups, one, two, and six. The variable sex had no significant effect upon any of the groups.

Based upon the above results, the null hypotheses stated on pages three and four for all variables, except sex, were rejected. A

dependency existed between each of the four variables, instructor, laboratory problem, major, and classification, and the variable errors.

At this point in the study the following questions became apparent. Was the variable major affecting the variable instructor? Would dependency still exist if the five variables were tested against errors grouped into categories according to type of error rather than categories according to frequency of error occurrence? Therefore the following new hypotheses were formulated:

1) Majors of beginning programming students were independent of instructors.

2) Types of errors made by beginning programming students were independent of instructors.

3) Types of errors made by beginning programming students were independent of laboratory problem number.

4) Types of errors made by beginning programming students were independent of major.

5) Types of errors made by beginning programming students were independent of classification.

6) Types of errors made by beginning programming students were independent of sex.

When the variable major was tested against the variable instructor, a chi-square value of 9.94 for fifteen degrees of freedom resulted. This value was not significant. A chi-square value of 25.00 was needed to reject the null hypothesis that majors of beginning programming students are independent of instructors.

In order to determine if the last five null hypotheses of independence stated above could be rejected or accepted, errors were grouped into categories according to types of errors. These error types are listed in Appendix B. The chi-square program was then used to test the variables instructor, laboratory problem, major, classification, and sex against types of errors. The results of this test are shown in Table VIII.

Because of the nature of the chi-square test only the types of errors that consisted of two or more errors that had a frequency of occurrence of five or greater were used in the above test. This was necessary to obtain meaningful values from the chi-square test. A total of twenty types of errors were tested.

The variable instructor significantly influenced three types of errors, format (first grouping), functions and subroutines, and subscripted variables. Chi-square values for the variable laboratory problem were significant for thirteen types of errors. This variable influenced more types of errors than did any of the other variables. Similar results were also shown for the variable laboratory problem by the chi-square test for single errors.

Five types of errors had significant chi-square values for the variable major. Those five types were data statement, format (first grouping), job termination, undefined variables, and subscripts.

The variable classification was significant for the following four types of errors: format (first grouping), job termination, undefined variables, and if statements. Only three types of errors showed significance when tested against the variable sex. These types were

TABLE VIII
CHI-SQUARE VALUES FOR FIVE VARIABLES TESTED AGAINST TYPES OF ERRORS

Type of Error	Instructor	D.F.	Laboratory Problem	D.F.	Major	D.F.	Classification	D.F.	Sex	D.F.
Card Format	17.58	15	38.16*	20	23.08	25	22.53	20	4.97	5
Data Statement	20.08	12	31.99*	12	34.49*	20	25.40	16	1.84	4
DO LOOP	17.89	15	35.74*	15	21.44	25	15.87	20	2.97	5
Format	49.54*	12	27.98*	16	31.53*	20	51.07*	12	3.49	4
Format	3.93	3	2.38	3	2.66	5	4.78	4	.02	1
Input/Output	21.03	18	24.49	24	27.94	30	26.53	24	3.55	6
Input/Output	6.34	3	5.82	4	5.76	5	7.73	4	5.26*	1
Job Termination	5.32	6	15.07	8	24.55*	10	23.83*	8	6.09*	2
Statement Functions	23.42*	12	24.20	16	26.75	20	23.89	16	3.55	4
Statements and St. Numbers	21.96	15	42.84*	20	9.46	25	19.55	20	7.44	5
Subscripted Variables	26.93*	9	18.45*	9	14.86	15	9.90	12	4.67	3
Syntax	10.49	12	56.44*	16	29.54	20	14.93	16	1.45	4
I/O Operations	7.01	6	19.16*	8	11.29	10	4.22	8	1.69	2
Undefined Variables	1.00	3	26.11*	4	14.21*	5	9.60*	4	.01	1
Variable Names	8.80	6	25.09*	8	9.65	10	12.29	8	1.49	2
Subscripts	.91	3	8.23*	3	12.68*	5	7.49	3	.41	1
Equal Signs	5.28	6	23.66*	8	5.34	10	7.87	8	7.77*	2
IF Statements	11.33	6	60.36*	8	11.73	10	25.68*	8	1.17	2
Parentheses	2.35	3	1.81	4	4.75	5	2.64	4	.00	1
Hollerith Constants	.04	2	1.30	2	1.59	5	.49	3	xxxx	x

* significance at .05% level

D.F. = Degree of Freedom

x = incomplete data

Input/Output (second grouping), job termination, and equal signs.

Table IV listed the error types in order of frequency of occurrence. Of the sixteen most frequently occurring error types, three of these, format (second grouping), Input/Output (first grouping), and parentheses were independent of any of the five variables.

Six of the error types were affected by one variable, five by two variables, and three by three variables. Or expressed another way, 50% of the sixteen most frequently occurring error types were significantly influenced by either zero or one variable and 50% by either two or three variables.

Statements and statement numbers, syntax, Input/Output, DO-LOOPS, parentheses, card format and contents, I/O operations, format, and variables names were the error types affected by one or zero variables. Job termination, undefined variables, format, subscripted variables, subscripts, equal signs, IF statements, and DATA statements were the error types affected by two or three variables.

Based upon the above results, the hypothesis that the types of errors made by beginning programming students are independent of laboratory problem number was rejected. The other four hypotheses concerning types of errors were accepted.

SUMMARY

In summary, male students made slightly more errors than female students. Sophomores made the fewest percentage of errors, compared to total representation, while juniors made the most errors. Mathematics majors made slightly fewer errors and agriculture majors made slightly

more errors when the percentage of errors made by students was compared to the percentage of students within a major.

Thirteen of a possible 281 errors accounted for 50% of the total 2,148 errors made. NONE was the most frequently occurring error.

Thirty-two different types of errors occurred. Statements and statement numbers was the most frequently occurring type of error.

Instructors somewhat influenced the average number of runs per lab problem. Graduate students and freshmen required the most number of runs, while juniors the least number of runs.

Chi-square values indicated that when the five variables instructor, laboratory problem, major, classification, and sex were tested against single errors, dependence existed for the first four variables and independence existed for the variable sex. The variable major was independent of the variable instructor. When the chi-square test was applied to types of errors, only the variable laboratory problem showed dependence. The other four variables were independent of types of errors and their corresponding hypotheses were accepted.

CONCLUSIONS

The results of this study showed that some errors occurred more frequently than other errors. Therefore it was possible to determine what kinds of errors were made by beginning programming students.

Based upon the chi-square values for the five variables tested against the occurrence of single errors, independence existed for the variable sex while dependence existed for the variables instructor, laboratory problem, major, and classification.

When errors were grouped into types of errors only the variable laboratory problem was dependent upon the variable errors. Therefore, all the hypotheses of the variables that showed dependence were rejected and the hypotheses of the variables that showed independence were accepted.

RECOMMENDATIONS

As a result of this study, a list of the most frequent errors and error types was obtained. In order to determine if the knowledge of these errors could have any effect upon reducing their occurrence, more similar studies will have to be conducted. Perhaps by having the instructors more thoroughly explain the usage of the most frequently occurring errors, their occurrence could be reduced.

Also there may be other factors that might significantly affect error and error type occurrence. For example, the amount of emphasis that the instructor put upon the students to accurately complete the questionnaire would affect the results. Another factor might be the amount of prior knowledge that a student might have upon entering such a course. There may be other meaningful ways to group the errors. All of these need to be checked for significant effect upon errors.

BIBLIOGRAPHY

- Blatt, John M. Introduction to FORTRAN IV Programming: Using the WATFOR Compiler. Pacific Palisades: Goodyear Publishing Co., 1968. 313pp.
- Chapman, R. L. and Carpenter, J. T. "Computer Techniques in Instruction." Programmed Learning and Computer-Based Instruction. Coulson, J. F., editor. New York: Wiley and Sons, 1962. pp. 242-43.
- Chi Square Program. Department of Statistics and Computer Science Statistical Library, Kansas State University.
- Fishman, Elizabeth Jane, Keller, Leo, and Atkinson, Richard C. "Massed Versus Distributed Practice in Computerized Spelling Drills." Journal of Educational Psychology. 59: 290-96, 4, August, 1968.
- Fryer, H. C. Concepts and Methods of Experimental Statistics. Boston: Allyn and Bacon, Inc., 1966. pp. 82-92, 569.
- Goodman, R. "Some Thoughts on Teaching Programming." Computers in Education. Hall, J. A. P., editor. New York: Macmillan Company, 1962. pp. 53-6.
- Generalized Means Program. Department of Statistics and Computer Science Statistical Library, Kansas State University.
- Silvern, Gloria M. and Silvern, Leonard C. "Programmed Instruction and Computer-Assisted Instruction--An Overview." Proceedings of the IEEE. 54:1648-55, 12, December, 1966.

APPENDIX

APPENDIX B

ERROR DESCRIPTIONS AND FREQUENCIES

In this Appendix, the diagnostic message codes printed out by the WATFOR/360 compiler are given. Next to each message code is its frequency of occurrence and its diagnostic message. Errors are grouped into types and the total frequency of occurrence for each error type is given.

CARD FORMAT AND CONTENTS

CC-0	32	Columns 1-5 of Continuation Card Not Blank.
CC-1	1	Too Many Continuation Cards (Maximum of 5).
CC-2	9	Invalid Character in Fortran Statement.
CC-3	7	First Card of a Programme is a Continuation Card. Probable Cause Statement Punched to Left of Column 7.
CC-5	4	Blank Card Encountered.
CC-6	4	Keypunch Used Differs from Keypunch Specified on the Job Card.
CC-7	22	First Character of Statement Not Alphabetic.
CC-8	2	Invalid Character(s) Concatenated with Fortran Keyword.
CC-9	9	Invalid Characters in Columns 2-5. Statement Number Ignored Probable Cause-Statement Punched to Left of Column 7.

90

COMMON

CM-3	2	Initializing of Common Should Be Done in a Block Data Sub-programme.
------	---	--

2

FORTRAN TYPE CONSTANTS

CN-6	19	Illegal Use of Decimal Point.
------	----	-------------------------------

19

DATA STATEMENT

- DA-1 7 Non-Constant in Data Statement.
- DA-2 9 More Variables than Constants in Data Statement.
- DA-5 1 Extended Data Statement not in /360 Fortran.
- DA-6 7 Non-Agreement between Type of Variable and Constant in Data Statement.
- DA-8 4 Variable Previously Initialized. Latest Value Used. Check Common/Equivalenced Variables.
- DA-9 2 Initializing Blank Common not Allowed in /360 Fortran.
- DA-A 11 Invalid Delimiter in Constant List Portion of Data Statement.
- DA-B 8 Truncation of Literal Constant has Occurred.

49

DIMENSION STATEMENTS

- DM-0 5 No Dimensions Specified for a Variable in a Dimension Statement.
- DM-2 1 Initialization in Dimension Statement is Illegal.
- DM-3 4 Attempt to Re-dimension a Variable.
- DM-4 2 Attempt to Dimension an Initialized Variable.

12

DO LOOPS

- DO-0 4 Illegal Statement Used as Object of DO.
- DO-1 16 Illegal Transfer into the Range of a DO LOOP.
- DO-2 9 Object of a DO Statement has Already Appeared.
- DO-3 11 Improperly nested DO-LOOPS.
- DO-4 7 Attempt to Redefine a DO-LOOP Parameter Within Range of Loop.
- DO-5 51 Invalid DO-LOOP Parameter.
- DO-7 18 DO-Parameter is Undefined or Outside Range.

- DO-8 3 This DO LOOP Will Terminate after First Time Through.
 DO-9 1 Attempt to Redefine a DO-Loop Parameter in an Input List.

110

EQUIVALENCE AND/OR COMMON

- EC-1 3 Common Block has a Different Length than in a Previous Subprogramme.
 EC-8 1 Variable Used with Non-Constant Subscript in Common/Equivalence List.

4

END STATEMENTS

- EN-0 24 No End Statement in Programme - End Statement Generated.
 EN-1 3 End Statement Used as Stop Statement at Execution.

27

EQUAL SIGNS

- EQ-6 29 Illegal Quantity on Left of Equal Sign.
 EQ-8 44 Illegal Use of Equal Sign.
 EQ-A 23 Multiple Assignment Statements not in /360 Fortran.

96

EQUIVALENCE STATEMENTS

- EV-2 1 Less than 2 Members in an Equivalence List.

1

POWERS AND EXPONENTIATION

- EK-9 16 X**Y Where X.LT.0.0, Y.NE.0.0.

16

FORMAT

- FM-0 33 Invalid Character in Input Data.
 FM-2 2 No Statement Number on a Format Statement.

FM-5	22	Format Specification and Data Type Do Not Match.
FM-6	1	Incorrect Sequence of Characters in Input Data.
FT-1	16	Invalid Character Encountered in Format.
FT-2	11	Invalid Form Following a Specification.
FT-3	2	Invalid Field or Group Count.
FT-4	4	A Field or Group Count Greater than 255.
FT-6	4	No Closing Quote in a Hollerith Field.
FT-7	3	Invalid Use of Comma.
FT-A	4	Character Follows Closing Right Bracket.
FT-D	1	Invalid Character Before a Right Bracket.
FT-E	4	Missing or Zero Length Hollerith Encountered.
FT-F	8	No Closing Right Bracket.

 111

FUNCTIONS AND SUBROUTINES

FN-3	1	Repeated Argument in Subprogramme or Statement Function Definition.
FN-4	11	Subscripts on Right Hand Side of Arithmetic Statement Function.
FN-7	8	Invalid Argument in Arithmetic or Logical Statement Function.
FN-8	1	Argument of Subprogramme is Same as Subprogramme Name.

 21

GO TO STATEMENTS

GO-0	3	Statement Transfers to Itself or to a Non-Executable Statement.
GO-2	1	Index of Computed 'Go To' is Negative or Undefined.
GO-3	2	Error in Variable of 'Go To' Statement.

 6

HOLLERITH CONSTANTS

- HO-1 1 Zero Length Quote-Type Hollerith.
 HO-2 1 No Closing Quote or Next Card not Continuation Card.
 HO-3 10 Hollerith Constant Should Appear only in Call Statement.
 HO-4 12 Unexpected Hollerith or Statement Number Constant.

 24

IF STATEMENTS (ARITHMETIC AND LOGICAL)

- IF-0 7 Statement Invalid after a Logical IF.
 IF-3 54 Arithmetic or Invalid Expression in Logical IF.
 IF-4 27 Logical, Complex, or Invalid Expression in Arithmetic IF.

 88

IMPLICIT STATEMENT

- IM-0 4 Invalid Mode Specified in an Implicit Statement.
 IM-1 1 Invalid Length Specified in an Implicit or Type Statement.
 IM-4 1 Specification Must Be Single Alphabetic Character, First Character Used.

 7

INPUT/OUTPUT

- IO-0 5 Missing Comma in I/O List of I/O or Data Statement.
 IO-2 13 Statement Number in I/O Statement not a Format Statement Number.
 IO-3 14 Buffer Overflow - Line Too Long for Device.
 IO-6 8 Variable Format not an Array Name.
 IO-8 16 Invalid Element in Input List or Data List.
 IO-9 6 Type of Variable Unit not Integer in I/O Statement.
 IO-C 26 Invalid Element in an Output List.
 IO-D 1 Missing or Invalid Unit in I/O Statement.

- IO-F 12 Invalid Delimiter in Specification Part of I/O Statement.
 IO-J 17 Invalid Delimiter in I/O List.
 IO-K 2 Invalid Delimiter in Stop, Pause, Data, or Tape Control Statement.

120

JOB CONTROL CARDS

- JB-1 1 Job Card Encountered During Compilation.

1

JOB TERMINATION

- KO-0 103 Job Terminated in Execution because of Compile Time Error.
 KO-1 1 Fixed Point Division by Zero.
 KO-2 3 Floating Point Division by Zero.
 KO-3 3 Too Many Exponent Overflows.
 KO-6 18 Job Time Exceeded.
 KO-8 2 Integer in Input Data is Too Large.

130

LIBRARY ROUTINES

- LI-C 6 Negative Argument for Sqrt or Dsqrt.

6

MIXED MODE

- MD-3 1 Relational Operator has a Complex Operand.
 MD-4 2 Mixed Mode - Logical with Arithmetic.

3

PARENTHESES

- PC-0 96 Unmatched Parenthesis
 PC-1 19 Invalid Bracket Nesting in I/O List.

115

ARITHMETIC AND LOGICAL STATEMENT FUNCTIONS

- SF-1 5 Previously Referenced Statement Number on Statement Function.
- SF-2 5 Statement Function is the Object of a Logical IF Statement.
- SF-3 3 Recursive Statement Function, Name Appears on Both Sides of =.

13

SUBPROGRAMMES

- SR-0 17 Missing Subprogramme.
- SR-2 3 Subprogramme Assigned Different Modes in Different Programme Segments.
- SR-4 6 Invalid Type of Argument in Subprogramme Reference.
- SR-7 1 Wrong Number of Arguments in Subprogramme Reference.
- SR-8 6 Subprogramme Name Previously Declined - First Reference Used.
- SR-9 1 No Main Programme.

34

SUBSCRIPTS

- SS-0 4 Zero Subscript of Dimension not Allowed.
- SS-1 91 Subscript out of Range.
- SS-2 2 Invalid Variable or Name Used for Dimension.

97

STATEMENTS AND STATEMENT NUMBERS

- ST-0 39 Missing Statement Number.
- ST-3 23 Multiply-Defined Statement Number.
- ST-4 19 No Statement Number on Statement Following Transfer Statement.
- ST-5 65 Undecodeable Statement.

- ST-7 2 Statement Number Specified in a Transfer is the Number of a Non-Executable Statement.
- ST-9 6 Statement Specified in a Transfer Statement is a Format Statement.
- ST-A 15 Missing Format Statement.

169

SUBSCRIPTED VARIABLELES

- SV-0 33 Wrong Number of Subscripts.
- SV-1 45 Array Name or Subprogramme Name Used Incorrectly without List.
- SV-4 15 Variable with Variable Dimensions is not a Subprogramme Parameter.
- SV-5 12 Variable Dimension Neither Simple Integer Name Nor Subprogramme Parameter.

105

SYNTAX ERRORS (ERRORS DETECTED IN ATTEMPTING TO DECODE ONE STATEMENT)

- SX-0 102 Missing Operator.
- SX-1 7 Syntax Error - Searching for Symbol, None Found.
- SX-3 3 Syntax Error - Searching for Symbol or Constant, None Found.
- SX-4 22 Syntax Error - Searching for Statement Number, None Found.
- SX-5 3 Syntax Error - Searching for Simple Integer Variable, None Found.
- SX-C 19 Illegal Sequence of Operators in Expression.
- SX-D 8 Missing Operand or Operator.

164

I/O OPERATIONS

- UN-0 68 Control Card Encountered on Card Reader During Execution.
- UN-1 2 End of File Encountered.
- UN-7 14 Too Many Pages of Output.

UN-9 5 You Have Attempted to Write Onto the Card Reader, or to Read from the Line Printer or Card Punch.

89

UNDEFINED VARIABLES, I.E., VARIABLES WHICH HAVE BEEN ASSIGNED NO VALUES

SO FAR

UV-0 52 Undefined Variable - Simple Variable.
 UV-1 1 Undefined Variable - Equivalenced, Commoned or Dummy Parameter.
 UV-2 61 Undefined Variable - Array Member.
 UV-6 1 Variable Format Contains Undefined Character(s).

115

VARIABLE NAMES

VA-0 2 Attempt to Redefine Type of a Variable Name.
 VA-2 29 Variable Name Longer than Six Characters. Truncated to Six.
 VA-4 2 Attempt to Redefine the Type of a Variable Name.
 VA-6 1 Illegal Use of a Subroutine Name.
 VA-8 20 Attempt to Use a Previously Defined Name as Function or Array.
 VA-C 1 Name Used as a Subprogramme Name was Previously Used as a Common Block Label.

55

THE FREQUENCY OF OCCURRENCE OF
SYNTAX ERRORS FOR BEGINNING PROGRAMMING STUDENTS

by

LINDA MARIE RUST

B. S., Kansas State University, 1968

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Statistics and Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1969

It was the purpose of this study (1) to determine what errors were made by beginning programming students and (2) to determine what effect the following variables had upon the errors that were made by students: instructor, laboratory problem, major, classification, and sex.

The objectives of this study stated as null hypotheses were:

- (1) Errors made by beginning programming students were independent of instructor.
- (2) Errors made by beginning programming students were independent of laboratory problem.
- (3) Errors made by beginning programming students were independent of major.
- (4) Errors made by beginning programming students were independent of classification.
- (5) Errors made by beginning programming students were independent of sex.
- (6) Majors of beginning programming students were independent of instructor.
- (7) Types of errors made by beginning programming students were independent of instructor.
- (8) Types of errors made by beginning programming students were independent of laboratory problem.
- (9) Types of errors made by beginning programming students were independent of major.
- (10) Types of errors made by beginning programming students were independent of classification.
- (11) Types of errors made by beginning programming students were independent of sex.

Data for the study were collected by use of a questionnaire. Data gathered from the questionnaire consisted of laboratory problem number, instructor, error code, type of error and run number. Eighty-four

students enrolled in a beginning programming course from Kansas State University, Manhattan, Kansas, participated in the study.

The results of the study showed what errors were made by beginning programming students. Based upon chi square values for the five variables tested against the occurrence of errors, independence existed for the variable sex while dependence existed for the variables instructor, laboratory problem, major, and classification.

The variable major was independent of the variable instructor. Chi square values for the five variables tested against types of errors showed dependence for the variable laboratory problem. Independence was indicated for the variables instructor, major, classification, and sex when tested against types of errors.