A COMPARISON OF EXAMINATION TECHNIQUES FOR INTRODUCTORY COMPUTER PROGRAMMING COURSES

Gretchen L. Van Meer
Central Michigan University
and
William H. Dodrill
West Virginia University

Introduction

The relative merits of various examination techniques is a critical issue when designing introductory computer programming courses. Two methods which are used extensively are multiple choice and programming exercises. Over the past several years, the examinations given in the introductory FORTRAN programming course at West Virginia University have included both multiple choice and programming exercise portions. Since for each examination, both portions were taken by the same student, the accumulated grades provide data for comparison of these techniques.

The accumulated data summarized in this comparison is for five consecutive semesters with three examinations and approximately 360 students per semester. The statistical methods used were the split-plot technique and the t-test. The primary objective was to identify any difference in student performance evaluation attributable to the examination procedures used.

While the results obtained are not necessarily conclusive, there is considerable evidence that a well-designed examination using either technique represents a fair approach to evaluating student performance.

Course Description

This is an introductory FORTRAN programming course taken primarily by students majoring in Business and Economics. Administrative procedures for the course have been described previously (1). Each student attends one of three lecture sections of approximately 120 each, which meet three times a week for a 50-minute lecture. In addition, each student attends one of nine laboratory sections of approximately 40 students each, which meets for one hour per week. The lecture sections are taught by faculty members and the laboratory sections are taught by graduate teaching assistants.

The lecture instructors cover the basics of programming in general, the FORTRAN language in particular, and general algorithms. The laboratory instructors assign and grade specific programming exercises, which are run batch on a WATFIV-5 compiler.

On the examinations, the multiple choice questions are designed to test the material covered in the lecture part of the course, while the programming exercise is designed to test the material covered in the laboratory sections.

Examination Procedures

Examinations are administered jointly to all three sections, with each student taking one version of a common examination. Examinations consist of a multiple choice section and a programming exercise. The answers for the multiple choice section are entered by the student onto a machine readable form, and the programming exercise is turned in separately. The multiple choice portion is machine graded; the programming exercise is hand graded by graduate teaching assistants. Each portion is graded on a scale of 0-100 and, therefore, the grades on each of the two portions could be compared for each student against him/herself.

Four versions of each examination are given. The variations in the multiple choice questions from one version to another include varying the questions, varying the choices for answers, and varying the order in which the questions are asked. The programming problems for a specific examination include four variations of a basic algorithm. The same basic programming concepts are covered in all four programming problems for a specific exam.

For the purpose of assigning a grade to the student, the programming score is entered by the graduate teaching assistant onto a reserved section of the

machine-readable form. This value is entered into the data base at the same time the multiple choice portion is machine graded. The program permits flexibility in weighting the two portions of the examination. For the examination data used in this paper, the multiple choice section was weighted at 75% and the programming exercise at 25% of the overall examination grade.

Data

The data consist of examination results over a period of five semesters, from fall semester of 1980 to fall semester of 1982. Each semester includes the results of three one-hour examinations. (A two-hour final examination was given each semester, but the results are not included here. The final examination consisted of multiple choice questions only, as the deadline for turning in grades did not permit time for grading a programming exercise.)

The number of students enrolled each semester was approximately 350. However, only the students taking all three hour examinations were included in the data base for this study. The number of students included per semester ranges from 224 to 345.

The policy in this course, as described in detail in an earlier paper (1), is to not permit students to make up any missed examination. However, when grades are computed, the lowest test score is dropped, so that a student may miss one examination without penalty. Because of the relatively large number of missing values, and because of the large size of the data base, it became expedient to eliminate those students who had missed one or more examinations. (Otherwie, asking the SAS program, referred to in the next section, to handle the missing data would have exceeded the capability of our equipment.) It has been our experience that the primary reason for missing examinations under these conditions is illness. Since viruses are no respectors of ability, it was felt that limiting the data to those students who took all three examinations does not bias the result.

Statitical Procedures

The statistical procedure used was the "split-plot" technique (2) using the Statistical Analysis System (SAS) package (3). This procedure was used with the three tests per semester as the "main units," examination type (multiple choice or program) as the "subunits," individual students as the "blocks," and semesters as "replicates."

The split-plot technique permits the analysis of all the data simultaneously while taking into account a large number of sources of variability. The main units (tests) are not independent; some association exists because they are taken by the same student. The subunits are not independent because both types of problems are done by the student. Consequently, the variability due to these effects are accounted for in the test sum of squares. Other variability which is taken into consideration is the various interactions which arise in this design.

The results are summarized in Table 1. Not only are there significant differences ($P < .0001$) between test types, but also between semesters and between examinations within a semester. As is often the case with a large data base, relatively small differences translate into large statistical significance. The question which then needed to be addressed was whether the differences were meaningful from a practical standpoint; that is, whether the difference in examination techniques would result in a student being assigned a different letter grade if one examination technique or the other were used.

To address this question, a paired t-test was done on an examination-by-examination basis. For consistency, the same data were used; that is, only the students who took all three examinations. These results are summarized in Table 2.

Of the fifteen examinations evaluated, five had no signficant difference ($P > .05$) between test type. Of the ten examinations with significant differences between examination type, the average difference was less than 5 points in three of them. In another four cases the average difference was less than 10 points; however, 10 points is usually the difference from one letter grade to another. It is therefore probably appropriate to note those cases with differences of more than 5 points. In the seven such examinations, the differences ranged from 6.8 to 23.7. In five of those examinations the programming grade was higher than the multiple choice grade, with the reverse in the other two cases.

Conclusions and Recommendations

While there are some significant differences between the two examination procedures, the differences are often not large and may very well be due to the nature of this particular examination. The multiple choice questions are designed to cover material presented in the lecture portion of the course while the programming exercise is designed to cover

TABLE 1

ANALYSIS OF VARIANCE PROCEDURE

DEPENDENT VARIABLE: SCORE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F VALUE
MODEL	5545	3448757.00265763	621.95798064	3.59
ERROR	2758	478110.06273254	173.35390237	
CORRECTED TOTAL	8303	3926867.06539018		

SOURCE	DF	ANOVA SS	F VALUE	PR > F
SEMESTER	4	102446.76369976	147.74	0.0001
IDN(SEMESTER)	1379	1241739.13502229	5.19	0.0001
TEST	2	807644.33309248	2329.47	0.0001
SEMESTER*TEST	8	131275.28433161	94.66	0.0001
IDN*TEST(SEMESTER)	2758	645154.04924404	1.35	0.0001
PART	1	18795.16486031	108.42	0.0001
SEMESTER*PART	4	82662.24887232	119.21	0.0001
IDN*PART(SEMESTER)	1379	323303.75293403	1.35	0.0001
TEST*PART	2	56329.36584778	162.47	0.0001
SEMESTER*TEST*PART	8	39406.90475301	28.42	0.0001

TESTS OF HYPOTHESES USING THE ANOVA MS FOR IDN(SEMESTER) AS AN ERROR TERM

SOURCE	DF	ANOVA SS	F VALUE	PR > F
SEMESTER	4	102446.76369976	28.44	0.0001

TESTS OF HYPOTHESES USING THE ANOVA MS FOR IDN*TEST(SEMESTER) AS AN ERROR TERM

SOURCE	DF	ANOVA SS	F VALUE	PR > F
TEST	2	807644.33309248	1726.32	0.0001
SEMESTER*TEST	8	131275.28433161	70.15	0.0001

TESTS OF HYPOTHESES USING THE ANOVA MS FOR IDN*PART(SEMESTER) AS AN ERROR TERM

SOURCE	DF	ANOVA SS	F VALUE	PR > F
PART	1	18795.16486031	80.17	0.0001
SEMESTER*PART	4	82662.24887232	88.15	0.0001

PR > F	R-SQUARE	C.V.
0.0001	0.878246	18.4029
ROOT MSE		SCORE MEAN
13.16639291		71.54515896

Table 2

	Fall 80	Spring 81	Fall 81	Spring 82	Fall 82
Test 1					
N of Students	345	274	298	224	243
MC Ave	83.4	65.6	79.6	87.7	85.4
Prog. Ave	90.2	89.3	83.0	92.5	90.1
Ave. Diff	6.8	23.7	3.4	4.8	4.7
Confidence Interval (95%)	5.4 to 8.2	22.0 to 25.4	1.4 to 5.5	3.0 to 6.7	2.7 to 6.6
P	.0001	.0007	.0001	.0001	.0001
Test 2					
N of Students	345	274	298	224	243
MC Ave	63.1	77.6	69.1	70.6	80.8
Prog. Ave	71.7	79.0	50.3	67.4	70.9
Ave. Diff	8.6	1.6	-18.8	-3.2	-9.9
Confidence Interval (95%)	6.3 to 10.9	-1.2 to 3.9	-21.6 to -16.0	-6.6 to 0.1	-12.4 to -7.3
P	.0001	.2826	.0001	.0553	.0001
Test 3					
N of Students	345	274	298	224	243
MC Ave	53.0	62.1	56.1	61.1	62.5
Prog. Ave	59.1	74.0	57.2	62.5	61.4
Ave. Diff	6.1	11.9	1.1	1.4	-1.1
Confidence Interval (95%)	4.1 to 8.2	9.6 to 14.3	-1.5 to 3.7	-1.8 to 4.6	-3.9 to 1.6
P	.0001	.0001	.3973	.3908	.4059

material presented in laboratory. The two portions of the examination do not necessarily cover exactly the same material.

These examination procedures were specifically designed to be complimentary, rather than two independent tests of the same material. Under these conditions, one third of the examinations showed no statistically significant difference and more than half showed no practical difference between the two examination techniques. We feel that the similarity is great enough to suggest that the results of well-designed examinations of both types on the same material would be similar.

It is our opinion that the ideal examination procedure for a programming course would include writing and running a program at a terminal under specified conditions. However, until a school has available adequate hardware and software, examinations are likely to continue to consist of questions such as ours. We hope that this evaluation of our data will be helpful in providing guidance to others who must deal with the problem of administering examinations to large-enrollment courses.

References

1. W. H. Dodrill, "Computer Support for Teaching Large-Enrollment Courses," The Papers of the Thirteenth SIGCSE Technical Symposium on Computer Science Education, ACM SIGCSE Bulletin, Vol. 14, No. 1, February 1982.

2. Shirley Dowdy and Stanley Wearden, "Statistics for Research," a volume in the Wiley Series in Probability and Mathematical Statistics, Wiley-Interscience, NY 1983.

3. "SAS User's Guide, 1979 Edition," SAS Institute Inc., Post Office Box 10066, Raleigh, NC 27605.

Acknowledgements

The authors would like to acknowledge the invaluable assistance of Dr. Shirley Dowdy and Professor Dan Chilko of the Department of Statistics and Computer Science of West Virginia University for their advice on the statistical procedures and SAS programs used in this paper.

****

Continued from page 23.

5.11 Are vectors, matrices, records before sets, files and pointers?

The types associated with sets, files and pointers are peculiar to Pascal. A "conservative" approach would be to teach these types after the traditional topics of vectors, matrices and records. The survey showed:

conservative:	8	
S VMR FP:	1	(Grogono)
VR SF M P:	1	(Koffman)
S R F VM P:	1	(Atkinson)

5.12 Is CONST introduced at the same time as VAR?

I believe that, for a beginner, the introduction of an identifier being synonymous with a constant at the same time as introducing identifiers to name variables is not a good idea. However, only three books (Conway, Cooper and Rohl) introduce CONST much later than VAR.

6. Postscript

The information presented here supplements the information provided by Moffat and Moffat. It is hoped that this paper will enable a Pascal teacher to find a text which adopts the order which is closest to the order that he prefers. It is interesting to note that on many crucial issues there seems to be little agreement between authors as to the order in which elements of Pascal should be taught.

REFERENCES

[1] to [18]
These texts are the same as references [1] to [18] of the paper by Moffat and Moffat [22].

[19] Editor's Notes, ACM SIGCSE Bulletin, vol. 14, no. 3, p. 2 (Sept. 1982).

[20] J. W. Atwood and E. Regener, "Teaching Subsets of Pascal", ACM SIGCSE Bulletin, vol. 13, no. 1, p. 96-103 (Feb. 1981).

[21] D. Cooper and M. Clancy, "Oh! Pascal!", W. W. Norton, New York, 1982.

[22] D. V. Moffat and P. B. Moffat, "Eighteen Pascal Texts: An Objective Comparison", ACM SIGCSE Bulletin, vol. 14, no. 2, p. 2-10 (June 1982).