Computer Aptitude Tests as Predictors of Novice Computer Programmer Performance

ASHOK SUBRAMANIAN

AND

KAILASH JOSHI

UNIVERSITY OF MISSOURI - ST. LOUIS

ABSTRACT

Information technologies such as distributed computing, object-oriented programming, and client server computing are rapidly changing the nature of computing in organizations. The availability of skilled personnel, however, has not kept pace with advances in the technology of information systems. In fact the lack of trained information systems (IS) personnel is widely recognized as a major impediment to the future development of IS in organizations. As a result, effective selection and training of IS personnel is an issue of utmost importance to the IS community. Computer aptitude tests may be useful tools to help organizations select suitable IS personnel.

This paper discusses the need for a suitable aptitude test to help predict the performance of individuals in computer programming. The effectiveness of the Computer Aptitude, Literacy and Interest Profile Test (CALIP) was empirically investigated using programming tasks in BASIC. The results indicated that CALIP, overall, is not a good predictor of programming performance. However, in both samples examined at least one dimension of the CALIP test was significantly associated with performance in programming tasks. Thus, it appears that the CALIP instrument could be modified and refined to function as an effective predictor of programming performance. The paper identifies the need for additional research in this area and makes some recommendations for future research.

INTRODUCTION

What qualities are typical of the ideal employee or student? Most employers or college administrators would agree that they prefer individuals who are dependable, motivated, goal-oriented, persistent, and teachable. They also may be looking for certain prerequisite work skills and abilities. The interview still seems to be the best single method for obtaining information necessary to evaluate personal characteristics and credentials of applicants. As useful as the interview may be, though, additional objective data may be needed.

In business, personnel selection is a common and necessary task but it can have serious consequences for both the organization and the individual. Organizations are often plagued by a high risk of project failure, strict information systems (IS) budget constraints, IS project backlogs, and poor programmer productivity [2,3,15,27]. A poor programmer in a company could consume 5 to 20 times more in time, effort, and equipment resources than a good one [18,25]. Persistent mistakes by incompetent programmers have delayed projects and even cost some MIS managers their jobs. IS experts have predicted that the future development of IS in organizations will be constrained by the availability of skilled and competent personnel to effectively utilize IT products. IS experts have identified the shortage of skilled IS personnel as a major cause of failures in the implementation of information systems [9,28]. Therefore, it is important for organizations to have an effective personnel selection and training processes so that individuals with the requisite aptitude and skills could be assigned to IS related tasks.

In colleges and universities, the opportunity costs can be high for those students who are denied admission to a computer course due to enrollment limitations, only to see those who enrolled subsequently drop the same course. Limited exposure can result in unrealistic vocational goals (either too limited or too expansive). For example, some students mistakenly interpret good experiences in playing computer games and using word processing packages as an indicator of a high programming aptitude.

The selection and career guidance process can be enhanced if reliable aptitude tests are available to assist in measuring an individual's potential and cognitive "fit." Indi-

vidual differences such as cognitive abilities and aptitudes have been identified in the literature as important determinants of job performance [16]. An entry-level applicant may have no meaningful previous work experience or training to review. The aptitude test scores can provide useful data to help an organization focus its selection, training, and guidance resources on candidates possessing potential in the mixture of attributes deemed suited for computer programming careers.

This paper presents the results of an empirical evaluation of a computer programming aptitude test called the Computer Aptitude, Literacy & Interest Profile test (CALIP). Our literature search revealed that CALIP is the only aptitude test designed for novice programmers. Even though the reliability and construct validity of the instrument has been established by Poplin et. al. [23], there is no significant research on the efficacy of CALIP as a predictor of programming performance. This research addresses that shortcoming. The test sample comprised high school and graduate students performing programming tasks using the BASIC programming language. The next section presents the rationale for utilizing aptitude tests. The following section reviews the literature relating to the interpretation of aptitude tests. The succeeding sections discuss the structure of the CALIP test, research methodology, sample, and the relevant variables of this study. The paper concludes with an analysis of the results, implications, and directions for future research. A glossary of terms used in this paper is included at the end of the paper.

APTITUDE TESTS

Organizations typically select individuals based on a personal interview. Although the interview process may reveal a candidate's verbal and communication skills, it may not be the best tool for assessing a candidate's logical reasoning abilities or analytical skills. A primary aim of assessing aptitudes is to locate strengths and weaknesses which are not apparent to an interviewer. Although proficiency tests may be more useful for evaluating experienced candidates, Bookman [4] believes that aptitude tests can also be a valuable tool for screening candidates for entry-level programming positions. This is because proficiency tests are based on the assumption that the subjects are experienced in the domain in which they are being tested. Therefore, the test is designed to measure differences in the level of efficiency or proficiency of the subjects. Aptitude tests, on the other hand, are based on the assumption that subjects have no experience in the domain in which they are being tested. The main purpose of the aptitude test is to measure if the subjects have the cognitive skills necessary to succeed in the domain of interest.

The diversity of computer-related occupations, as well as the range of skills within these occupations, have a direct bearing on how aptitudes should be assessed. No one test can adequately assess ability and interest in all occupational categories at all levels. On the other hand, a test of very specific abilities, such as logical programming skills, is likely to overlook information such as verbal and communication skills, needed by managers, teachers, or administrators to plan effective recruitment and training programs [23].

Historically, computer programming aptitude tests focused on potential skills of adult programmers [23]. The two best-known of these tests are IBM's revised Programmer Aptitude Test (PAT) and Science Research Associates' Computer Programmer Aptitude Battery (CPAB). The PAT is a one-hour test for adults consisting of number series, figure analysis, and arithmetic reasoning. The CPAB is a 79-minute timed battery consisting of sub-tests measuring adults' aptitudes for verbal meaning, mathematical reasoning, letter series reasoning, number ability, and flow chart diagramming.

The PAT and CPAB are well known aptitude tests; however these tests are more suited to assessing the aptitude of trained adult programmers [23]. This is because, as stated by Johnson [14], there is a close correspondence between tests such as the PAT and CPAB and the tasks performed by experienced adult programmers. This may be a desirable characteristic for testing experienced adult programmers, however, it is a distinct weakness when the aptitudes of inexperienced and novice programmers need to be assessed. Our literature search revealed that CALIP is the only instrument designed to measure the aptitudes of inexperienced, novice programmers [23]. In addition, CALIP includes a subtest to measure interest in computers which has been recommended by researchers such as Brillhart [5], and Reece & Gable [24]. We, therefore, examined the predictive efficacy of CALIP.

General Aptitude Test Results Interpretation

In an attempt to interpret the results from programmer aptitude surveys, Bush and Schkade [7] found a common thread running through the results. In their opinion, the lesson that can be learned from the otherwise diverse results is that no ideal personality or cognitive style for the Data Processing professional has emerged. They state, "Instead of the intuitive computer programmer, we found computer professionals to be thinking, judging, rational people who rely on their education, training, and experience to solve problems."

Middleton [21] states that programmers tend to exhibit a high degree of patience in their work and are not easily frustrated. Others list the ability to invent, concoct, conceptualize, and design. All of these sound like marvelous qualities but which are appropriate for a given task and how can they be measured objectively outside of a person's previous work experience?

According to Bush and Schkade [7], DP professionals in organizations may exhibit different sets of cognitive styles, or personality traits due to the "culture" of their organization. Reasons for the tendency toward homogeneity in organizations in general may be due to leadership assumptions on what traits produce success and the tendency to hire the same type of people for the same type of job. Couger and Zawacki [12,13] conducted studies to determine differences in the cognitive traits and personality makeup of DP professionals and other non-DP professionals. They concluded that DP personnel differ from the general population of employees quite significantly. Couger et. al., [11] conducted a ten year follow-up study and found persistent differences between IS personnel and non-IS personnel.

Since there is no single style or personality type that fits an individual for life in a given profession, the selection process is further complicated. More research is needed to provide organizations with testing instruments to simplify yet enhance the selection process [23].

In an academic environment such as universities, aptitude tests such as CALIP can be invaluable tools for university administrators, student counselors and students. As stated earlier, evidence from the literature suggests that successful professionals and individuals in computer-related professions possess very unique and specialized skills [7]. Tests such as CALIP can be used as tools to assess the aptitude of students who wish to pursue a career as MIS professionals. The value of such an aptitude test assumes even more importance when one considers the fact that at present the drop-out rate in introductory level programming courses in this mid-western university is around 40%. Perhaps, with appropriate counseling based on results of aptitude tests, students would be able to make better decisions about the choice of their careers.

Tests such as the PAT and the CPAB have been utilized in the industry as tools for aiding data processing managers in selecting competent programmers. These tests have been given to experienced programmers in order to identify their aptitude for their work. It is believed that a person with little or no aptitude for computer programming tasks will, over the long haul, experience a great deal of job dissatisfaction and may eventually choose to enter another field to work [23]. Aptitude tests may also be used as a basis for making training decisions. Persons with a greater aptitude for computer-related work may benefit significantly more than individuals who do not have the aptitude and the potential to succeed in the field of information systems.

Thus, the results of aptitude tests can be used as valuable aids in making decisions about the allocation of scarce training and professional development resources within the information systems department. MIS programs in universities can also benefit greatly through the use of such tools. Students with a high aptitude for computers are more likely to successfully complete the program than those who do not possess the requisite aptitude. Thus, student drop-out rates can be reduced.

Description of CALIP

CALIP was developed by Poplin, Drew and Gable [23] to identify talented individuals who might want to specialize in a computer-related career. Their goal was to test aptitude apart from previous experience and complex verbal skills (e.g., reading comprehension). Based upon their review of the literature, the developers selected a broad sampling of item formats related to computer abilities, such as logical, sequential, spatial, and quantitative abilities. CALIP was designed to accomplish at least three purposes:

(1) to identify talented persons who might otherwise lack the opportunity to demonstrate aptitude for computer-related occupations.

(2) to broaden the range of realistic career options for people in the process of making career decisions.

(3) to provide an empirical basis for administrators, business managers, and teachers to allocate resources.

In general, past research indicates that programming success depends on an individual's ability to organize problems and their solutions into conceptual categories. Thus, a good programmer has an inventory of problem categories into which he/she can map any specific problem. Then the programmer applies a solution that is appropriate for that problem category [19,26]. Research in the field of aptitude testing has shown that broad-based tests that assess mathematical, logical, and mental organizational abilities are good predictors of success in programming [20]. Visual abilities such as pattern recognition have also been associated with programming success [6]. In order to successfully recognize patterns one must have a mental inventory of familiar patterns. The subtests in CALIP are based on this premise. They are designed to assess an individual's ability to detect patterns. These pattern recognition tests employ mathematical patterns (i.e., the series subtest), logical patterns (i.e., the logical structures subtest), visual patterns (i.e., the graphical subtest) etc. Thus, CALIP is based on generally accepted prior research. Further, it is designed, specifically, to measure the aptitude of inexperienced programmers.

Description of CALIP Sub-tests

The **estimation sub-test** consists of 24 diagrams where the person is asked to determine the number of blackened squares. The imposition of a two-minute time limit forces the individual to choose a strategy of answering fewer questions by counting the blocks or answering more by estimating.

The **graphics pattern sub-test** of 20 questions is designed to be language-free but uses test problem-solving ability. Each contains a set of figures in which some or part of the figure is missing. By using implicit rules of patterning, the person is expected to select the correct figure following the pattern.

The **logical structures sub-test** measures the ability of a person to find similarities between apparently dissimilar pairs or groups of items. The 20 items are numbers, letters, or words typically with an implicit pattern.

The **series sub-test** requires the person to complete a number and/or letter sequence for 24 items. In addition to encouraging diverse thinking , as the pervious sub-test does, this sub-test also rewards persons who can detect internal and highly structured order.

The **interest sub-test** attempts to measure factors related to long-term motivation and direction of effort but not to assure competence in programming. The 20 items are related to four categories of intellectual endeavor: people-oriented, things-oriented, numeric, and qualitative/emotional.

RESEARCH METHODOLOGY

Two samples were used in this research. One sample comprised graduate students, and the other comprised high school students. The students in both samples were administered CALIP tests at the beginning of the semester. At the end of the semester, the scores on programming assignments and exams were used as measures of programming performance. Given below is a more detailed description of the nature of the sample and the variables used in the analyses.

Sample

There were two samples consisting of high school students (n=39), and graduate students (n=46). The notable characteristic of the samples is that all individuals had *no previous programming experience.* The students in both samples were enrolled in an introductory computer programming course. The programming language used in both courses was BASIC.

Data Collection

Each of the students was administered the CALIP test during the first week of classes. During the course of the semester their performance was evaluated based on the quality of their programs. A weighted average of scores on programming assignments and exams was used as the dependent variable in the analyses.

Dependent variable

Programming performance was used as the dependent variable. A weighted mean score was computed for each student based on their scores on an exam (weighted 60%) and two programming assignments (weighted 20% each). Identical programming assignments and exams were used in the two samples. These weighted mean scores were used as measures of the programming performance of students. These programs were designed to test students on the fundamental concepts of procedural programming, namely, sequence, selection and iteration (see appendix). The first assignment involved the modification and extension of an existing program. The second assignment required students to write a complete program. Together these programs tested students on concepts such input-output using a variety of statements, looping concepts, mathematical expressions, print formatting and report writing, etc. The exam tested students' ability to write programs and to report the output of a given program. The programs used in the exam included sequential structure, branching, and looping structure construction.

Independent variables

Scores on the sub-tests of the CALIP instrument were used as independent variables in the analyses. Thus, there were five independent variables corresponding to the five sub-tests of CALIP i.e., Series (SER), Graphical Patterns (GRP), Logical structures (LOG), Interest (INT), and Estimation (EST).

RESULTS

The data in this study were analyzed using multiple regression models. The dependent variable in the regression model was the weighted mean score earned by individuals on their programs and exam. The scores on each of the sub-tests of the CALIP instrument were used as independent variables. Regression models were tested for each of the two samples i.e., the high school students (n=39), and the graduates (n=46). The results of the regressions are presented in Tables 1 through 4.

Table 1 shows that for the high school student sample (BASIC programming environment) the independent variables explain 31% of the variance in the dependent variable ($R^2 = 0.31$). The model is statistically significant (p=0.02). An examination of the coefficients of the independent variables shows that the series sub-test (SER) was positively and significantly associated with the dependent variable (p=0.005). None of the other independent variables exhibited any statistically significant association with the dependent variable. Further, since the adjusted R^2 value was 10% lower than the R^2 value, there were indications that the inclusion of these nonsignificant independent variables was merely decreasing the available degrees of freedom, without significantly increasing the amount of variation explained by the independent variables. Consequently a series of regression

TABLE 1

High School Students

Source	DF	Sum of Squares	Mean Square	F Value	Prob>F
Model	5	551.60793	110.32159	3.003	0.0242
Error	33	1212.13566	36.73138		
C Total	38	1763.74359			

Root MSE	6.06064	R-square	0.3127
Dep Mean	86.48718	Adj R-square	0.2086
C.V.	7.00756		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	T for H0: Parameter=0	Prob>{T}
INTERCEPT	1	34.342858	20.20456126	1.700	0.0986
SER	1	1.088871	0.36327104	2.997	0.0051
LOG	1	0.277977	0.37321047	0.745	0.4616
EST	1	0.154967	0.36732257	0.422	0.6758
GRP	1	–0.035472	0.36033282	–0.098	0.9222
INT	1	0.255712	0.34923533	0.732	0.4692

TABLE 2

High School Students

Source	DF	Sum of Squares	Mean Square	F Value	Prob>F
Model	1	519.90362	519.90362	15.731	0.0003
Error	37	1255.87138	33.04925		
C Total	38	1775.77500			

Root MSE	5.74885	R-square	0.2928
Dep Mean	86.57500	Adj R-square	0.2742
C.V.	6.64031		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	T for H0: Parameter=0	Prob>{T}
INTERCEPT	1	50.094214	9.24259687	5.420	0.0001
SER	1	1.216026	0.30659304	3.966	0.0003

TABLE 3

Graduate Students

Source	Sum of DF	Mean Squares	Square	F Value	Prob>F
Model	5	455.73859	91.14772	1.239	0.3094
Error	40	2943.73968	73.59349		
C Total	45	3399.47826			

Root MSE	8.57866	R-square	0.1341
Dep Mean	83.47826	Adj R-square	0.0258
C.V.	10.27653		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	T for H0: Parameter=0	Prob>{T}
INTERCEPT	1	51.448873	25.90034937	1.986	0.0539
SER	1	1.301078	0.55786666	2.332	0.0248
LOG	1	–0.337473	0.44464646	–0.759	0.4523
EST	1	–0.164650	0.53843055	–0.306	0.7613
GRP	1	0.106354	0.45965602	0.231	0.8182
INT	1	0.151751	0.43631589	0.348	0.7298

TABLE 4

Graduate Students

Source	Sum of DF	Mean Squares	Square	F Value	Prob>F
Model	1	403.67290	403.67290	5.955	0.0187
Error	45	3050.53986	67.78977		
C Total	46	3454.21277			

Root MSE	8.23345	R-square	0.1169
Dep Mean	83.31915	Adj R-square	0.0972
C.V.	9.88183		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	T for H0: Parameter=0	Prob>{T}
INTERCEPT	1	53.729471	12.18504920	4.409	0.0001
LOG	1	0.986353	0.40420326	2.440	0.0187

models was tested in a stepwise manner, with an independent variable being eliminated at each step. In each model, the least significant variable was identified and eliminated in the next nested model. Table 2 shows that the elimination of all independent variables except SER greatly increased the statistical significance of the model, at the same time the reduction in the R^2 value was only 2%. In other words, SER explained 29% of the variation in programming performance while the other independent variables collectively explained a mere 2% of the variance in the dependent variable.

Table 3 presents the results of the sample of graduate students. The model was not statistically significant (p=0.31). The R^2 value was 0.13, however, the adjusted R^2 was only 0.03. In addition, an examination of the independent variables revealed that the logical structures sub-test (LOG) was the only one significantly positively associated with the dependent variable (p=0.03). The difference between the R^2 and adjusted R^2 values suggested that some of the independent variables could be eliminated from the model without significantly reducing the amount of variation explained by the model. Therefore, a stepwise elimination process was adopted, and the results of these analyses are presented Table 4. It is evident that the elimination of all independent variables, except logical structures (LOG), did not significantly reduce the R^2 value, which decreased from 0.13 to 0.12. More importantly, the elimination of the nonsignificant independent variables caused the model to become statistically significant (p=0.02). Logical structures (LOG) was significantly correlated with the dependent variable (p=0.02).5.0

DISCUSSION AND CONCLUSION

In general, the results do not provide much support for CALIP's effectiveness as an instrument for predicting programming performance. In only one sample (out of the two examined) was the overall model statistically significant. In other words, in only 50% of the cases could the CALIP test dimensions together explain a statistically significant amount of variation in the programming performance of students. Although the tests of the overall models do not reflect well on CALIP, some of the sub-tests — notably the logical structures (LOG) and the series sub-test (SER) did indicate statistically significant associations with programming performance. The stepwise elimination process revealed that CALIP sub-tests such as graphical patterns (GRP), interest (INT), and estimation (EST) did not adequately explain variations in performance of individuals. However, sub-tests such as logical structures (LOG) and series (SER) did explain a significant amount of the variation in the programming performance of individuals. Further, SER was an effective predictor of performance for the high school students but not for the graduate students.

The SER sub-test tests an individual's ability to detect an internal order in a set of numbers or letters. Procedural programming requires programmers to specify a sequence of steps for solving a problem. The development of such a solution is not possible unless the programmer can detect an internal order within the problem. Thus, an individual who performs well in the SER sub-test has the necessary cognitive ability to develop effective procedural programs. The LOG sub-test tests an individual's ability to relate dissimilar objects. This requires creative and diverse thinking. It is logical to assume that individuals with a broad knowledge base may be able to perceive relationships between seemingly dissimilar objects more effectively than those who do not possess sufficient breadth of knowledge. Thus, a person who has this ability may be able to perceive relationships among the various components of a problem, thereby, imposing a structure on the problem. This, in turn, facilitates the development of a structured solution in the form of a computer program.

In light of the previous discussion, one could speculate that the differential effect of the LOG and SER sub-tests on the performance the graduates and high school students respectively, may be a result of the differing breadth and domains of knowledge of the graduates and high school students. However, based on the data collected in this study it is not possible to offer empirically valid explanations for these results. Issues such as these, however, need to be addressed in future research.

In summary, the implications of these results are fourfold:

1. There is a need to reassess if instruments such as CALIP do indeed measure computer programming aptitude — a similar study could be repeated with larger and more diverse samples.
2. It appears that only one or two dimensions of computer programming aptitude may be major determinants of actual programming performance. Thus, further studies could focus on identifying these dimensions, and more important, *determining why these dimensions are better predictors of programming performance than others.*
3. There may be a complex relationship between the structure and syntax of a programming language, programming aptitude (measured by instruments such as CALIP), and programming performance. Thus, predictors of computer programming performance may need to take into consideration relevant characteristics of the programming language.
4. Finally, it is quite possible that there may be other dimensions of computer programming aptitude. In addition, there may be mediating factors(such as the characteristics of the programming language). Researchers should consider other potential predictors of programming performance in future research.

This study has addressed a topic that is emerging as an

important area of research. The future development of IS in organizations will be constrained by the availability of skilled and competent personnel to effectively utilize IT products. The shortage of skilled manpower is already evidenced by rising consultant fees and IS personnel salaries in many state-of-the-art technologies such as CASE tools, Client Server computer systems, etc. [1,8]. IS experts have identified the shortage of skilled IS personnel as a major cause of failures in the implementation of information systems [9,28]. Therefore, it is important for organizations to have effective personnel selection and training processes so that individuals with the requisite aptitude and skills could be assigned to IS-related tasks. Organizations in the field of computer education such as schools, colleges and universities play an important role in providing the industry with skilled IS personnel. It is important for these organizations to understand the characteristics of individuals that are likely to succeed in computer related tasks so that individuals with an aptitude for computer programming could be identified and encouraged to select IS as a career. The acute shortage of skilled manpower have prompted some to suggest using psychological tests to assess the capabilities of IS personnel [22]. We believe that computer aptitude tests are tools that may help alleviate the personnel selection problem. Our study has shown that aptitude tests such as CALIP have the potential to discriminate between good and poor programmers, however, the instrument cannot be effectively used in its original form. Additional research is needed to develop and refine the CALIP instrument to effectively assess and predict programming ability.

GLOSSARY

Aptitude: A natural or acquired ability for learning and understanding.

Aptitude test: A standardized test to measure the ability of individuals to develop skill and acquire knowledge.

Cognitive style: The way in which knowledge is acquired by using different mental processes such as perception, reasoning, or intuition.

Experience: Active participation in an activity, or events leading to the accumulation of knowledge or skills.

Personality: The relatively stable organization of all personal characteristics. An enduring pattern of attributes that define the uniqueness of a person.

Skills: Proficiency, ability or dexterity acquired in a specific area.

REFERENCES

[1] Appleton, E.L., "Staffing Up? Here's What You'll Pay," *Datamation,* October 15, 1994, pp. 53–56.

[2] Benjamin, C. "Information Technology in the 1990s: A Long-Range Planning Scenario," *MIS Quarterly,* Vol. 6, Num. 2., 1982, p. 11–31.

[3] Blank, M., and Barratt, D., "Finding and Selecting Systems Analysts and Designers," *Journal of Systems Management,* Vol. 39, March 1988, p. 8–11.

[4] Bookman, Harvey, "Testing For the Best Programmer," *Datamation,* April 1, 1989, pp. 83–86.

[5] Brillhart, L., "Computers: An Answer to Engineering Student Learning Styles at the Community College," *Computers and Education,* 1980, vol. 4, pp. 247–253.

[6] Brown, L., Sherbenou, R.J., and Johnsen, S.J., *Test of Nonverbal Intelligence,* 1982, PRO-ED, Industrial Oaks Blvd., Austin, Texas 78735.

[7] Bush, Chandler M. and Schkade, Lawrence L., "In Search of the Perfect Programmer, *Datamation,* March 15, 1985, p. 128 (4).

[8] Caldwell, B., "In the Money," *Informationweek,* June 5, 1995, pp. 34–44.

[9] Chabrow, E.R., "The Training Payoff," *Information Week,* July 10, 1995, pp. 36–46.

[10] Cheney, P., and Lyons, N., "IS Skill Requirements: A Survey," *MIS Quarterly,* Vol. 4, Num. 1, 1980, p. 35–43.

[11] Couger, D., Opperman, E., and Amoroso, D., "Motivating IS Managers in the 1990s," *Inside DPMA,* May, 1992, p. 6–9.

[12] Couger, D., and Zawacki, R., "What Motivates DP Professionals?" *Datamation,* Sept. 24, 1978, p. 116–123.

[13] Couger, D., and Zawacki, R., *Motivating and Managing Computer Personnel,* John Wiley & Sons, New York, 1980.

[14] Johnson, R.T., Computer Programmer Aptitude Battery, in O.K. Burros (Ed.), *The 7th Mental Measurements Yearbok,* 1972, Highland Park, NJ: Gryphon Press.

[15] Jones, C., "Measuring Programmer Quality and Productivity," *IBM Systems Journal,* Vol. 17, Num. 1, 1987, p. 39–63.

[16] Kanfer, R., "Motivation Theory and Industrial/Organizational Psychology," in Marvin Dunnette and Leaetta Hough (Eds.), *Handbook of Industiral and Organizational Psychology: Vol. 1: Theory in Industrial and Organizational Psychology,* Consulting Psychologists Press, Palo Alto, CA, 1990, p. 75–170.

[17] Konvalina, J., Wileman, S.A., and Stephens, L.J., "Math Proficiency: A Key to Success for Computer Science Students," *Communications of the ACM,* 1983, vol. 26, pp. 377–382.

[18] Laughery, R., and Laughery, K., "Human Factors in Software Engineering: A Review of the Literature," *Journal of Systems and Software,* 5, 1985, pp. 3–14.

[19] Mayer, R.E., "A Psychology of Learning BASIC," *Communications of the ACM,* 1979, vol. 22, pp. 589-593.

[20] McKeithen, K.B., and Reitman, J.S., "Knowledge Organization and Skill Differences in Computer Programmers," *Cognitive Psychology,* 1981, vol. 13, pp. 307–325.
[21] Middleton, Brett, "Programmers: Not A Breed Apart," *Computerworld,* April 4, 1988, p. 63 (3).
[22] Moad, J., "Psych Tests for MIS Staff: Is This Nuts?" *Datamation,* July 1, 1994, pp. 27–29.
[23] Poplin, Mary S., Drew, David E., and Gable, Robert S., *Computer Aptitude Literacy, and Interest Profile,* 1984, PRO-ED, Industrial Oaks Blvd., Austin, Texas 78735.
[24] Reece, M.J., and Gable, R.K., "The Development and Validation of a Measure of General Attitudes Toward Computers," *Educational and Psychological Measurement,* 1982, vol. 42, pp. 913–917.
[25] Sackman, H., *Computer, System Science, and Evolving Society,* 1967, New York: Wiley.
[26] Soloway, E., "Learning to Program = Learning to Construct Mechanisms and Explanations," *Communications of the ACM,* 1986, vol. 29, pp. 850–858.
[27] Stamps, D., "CASE: Cranking out Productivity," *Datamation,* July 1, 1987, pp. 55–58.
[28] Whiting, R., "From COBOL to Client/Server," *Client/Server Today,* April 1994, pp. 34–38.

APPENDIX

Assignment 1

Due: _____

The following BASIC program prints the average of a set of exam scores. In this assignment you are to modify the program so that it can be used to find the average scores on an exam that has been taken by several classes. Specifically, the program should output an average for each class, and the overall average (the average of the averages).

```
10  REM ************************************************************************
20  REM * THIS PROGRAM WILL DETERMINE THE AVERAGE OF A SET OF EXAM       *
30  REM * SCORES.  IT USES FOR INPUT THE NUMBER OF STUDENTS IN A CLASS   *
40  REM * AND THE INDIVIDUAL EXAM SCORES.  IT PRODUCES THE AVERAGE       *
50  REM * OF THE EXAM SCORES FOR OUTPUT.                                 *
60  REM ************************************************************************
70  LET TOTAL = 0
80  PRINT "HOW MANY STUDENTS IN THIS CLASS?"
90  INPUT NUMBER.OF.STUDENTS
100 FOR COUNT - 1 TO NUMBER.OF.STUDENTS
110 PRINT "PLEASE ENTER AN EXAM.SCORE"
120 INPUT EXAM.SCORE
130 LET TOTAL = TOTAL + EXAM.SCORE
140 NEXT COUNT
150 LET EXAM.AVERAGE = TOTAL / NUMBER.OF.STUDENTS
160 PRINT "THE AVERAGE SCORE IS"; EXAM.AVERAGE
170 END
```

Your modified program must work with <u>any</u> data of the following form:

1st data value	:	number of classes
2nd data value	:	number of students in the first class
3rd, 4th, ..., etc.	:	exam scores for the first class
next:	:	number of students in the second class
	:	exam scores for the second class
	:	the pattern of the above two lines repeats for each class

Submit a printed listing of your BASIC program and executive results.
Use the following data:

```
 4 - number of classes
 3 - number of students in the first class
78 - score of a student in this class
65 - do
90 - do -
 2 - number of students in this class
75 - a score
75 - do -
 1 - number of students
 0 - score
 4 - number of reports
75 - score
80 - score
95 - score
57 - score
```

Assignment 2

Due: _____

INPUT: A series of DATA statements will be used to provide input values for this assignment. Each DATA statement will have six data values. The data represents transactions on a credit card. The six values are:

1) A numeric code of 1, 2, 3, or 4. A code value of 1 means that the data represents the beginning balance. A value of 2 means the data represents a purchase transaction. 3 means the data represents a payment. A value of 4 means end of data for this account.
2) The account number
3) The amount of the transaction or beginning balance
4-6) The date

You may assume the data is sorted by account number and the first DATA statement for each account has a code of 1. The last data is all zeros.

OUTPUT: For each account, produce an account report of the following form:

```
BIGDEBT CREDIT CARE - MONEY STATEMENT
ACCOUNT NUMBER xx
DATE        PURCHASE      PAYMENT       INTEREST      BALANCE
mm/dd/yy                                              xxxxxx.xx
mm/dd/yy                  xxxxx.xx                    xxxxxx.xx
mm/dd/yy      xxxxx.xx                                xxxxxx.xx
mm/dd/yy     xxxxxx.xx    xxxxxx.xx      xxx.xx       xxxxxx.xx
```

Monthly interest charges are 1/12 of 18% of the beginning balance.
Use the following data: (Your program should accept any data like the above.)

```
1     34     100.00     5 01 87
2     34     87.53      5 08 87
2     34     10.00      5 20 87
3     34     100.00     5 20 87
4     0      0          0  0  0
1     52     1000.00    5 01 87
3     52     1.00       5 15 87
4     0      0          0  0  0
1     54     1.00       5 01 87
4     0      0          0  0  0
```

1	63	99999.99	5 01 87
3	63	10000.00	5 02 87
4	0	0	0 0 0
1	65	55.55	5 01 87
2	65	1.00	5 11 87
2	65	100.00	5 12 87
4	0	0	0 0 0
1	73	2.00	5 01 87
3	73	3.00	5 15 87
3	73	4.00	5 18 87
4	0	0	0 0 0
1	88	100.00	5 01 87
2	88	100.00	5 12 87
3	88	50.00	5 13 87
2	88	75.00	5 14 87
3	88	50.00	5 18 87
0	0	0	0 0 0

ABOUT THE AUTHORS

Ashok Subramanian *is an Assistant Professor of Management Science and Information Systems at the University of Missouri, St. Louis. Dr. Subramanian has a Ph.D. in Management Information Systems from the University of Houston. His research interests include Telecommunications technologies, applications, and policies; distributed computing technologies such as Client Server Computing; object-oriented systems development; and computer personnel issues such as productivity. He has worked as a consultant in the areas of open system architecture development; the development of coorporate standards, policies, and procedures for managing corporate computer networks.*

Kailash Joshi *is associate professor of MIS at the University of Missouri-St. Louis. He received his Ph.D. in management information systems from Indiana University in 1986. He has also worked for 10 years in industry in the areas of purchasing, materials, production, and systems. His current research interests include user behavior, user attitudes, management of MIS, user information satisfaction, the political perspective of MIS, information systems applications, computer applications in developing countries, and production and materials management. His other papers have appeared or are forthcoming in* MIS Quarterly, Decision Sciences, Information and Management, Omega: The International Journal of Management Science, Journal of Purchasing and Materials Management, Production and Inventory Management Journal, *and* Database.