An Empirical Investigation of the Relationship Between Success in Mathematics and Visual Programming Courses

Garry White
Marcos Sivitanides
Department of Computer Information Systems
College of Business Administration
Southwest Texas State University
601 University Drive
San Marcos, TX 78666
GW06@business.swt.edu ms06@business.swt.edu

ABSTRACT

Many universities do not have prerequisites for the introductory computer visual programming course. Therefore, faculty and students do not have any means of predicting the student's performance in this course. This research addresses this issue. Past research and accepted theory are presented to show the cognitive requirements for success in a first procedural programming course to be similar to those required for success in a mathematics course. Such research is lacking for visual programming. This research shows similar correlations between math courses and visual programming courses. Significant positive correlations were found between grades from Freshmen mathematics courses, ACT math scores, SAT math scores and grades from a Sophomore introductory visual programming course. This indicates that students who perform well in Freshman level Math courses, possess the cognitive characteristics required to perform equally well in Sophomore level visual programming classes. We can predict that students who perform well in math courses will perform equally well in a visual programming course.

Keywords: cognitive development, prerequisites, programming languages, procedural programming, visual languages, mathematics, business mathematics.

1. INTRODUCTION

There is a need to have prerequisites for programming courses to ensure that those who enroll have the necessary cognitive skills to be successful. A strong mathematics background predicts success in procedural programming (Alspaugh, 1970; Ricardo, 1983; Ignatuk, 1986). Studies have shown that math scores on the Scholastic Aptitude Test (SAT-M) and the American College Testing program (ACT) correlate with procedural programming course grades (Renk,1987; Ott, 1989). Several other studies have shown a relationship between mathematics proficiency and success in procedural programming (Taylor and Mounfield, 1991). These studies support the practice of mathematics prerequisites for computer courses (Ralston, 1984; Saiedian, 1992).

However, there is no research to show whether this is true or not with visual programming. The purpose of this study is to investigate whether, like with procedural programming, there is a relationship between mathematics proficiency and success in visual programming. The Null hypothesis used in this research is: "there is no relationship (predictability) between success in Math courses and success in a Visual Programming course."

1.2 Definitions of Procedural and Visual Programming

A procedural programming language is characterized by three properties: the sequential execution of instructions, the use of variables representing memory locations, and the use of assignment to change the values of variables (Louden, 1993). An example of such a language is COBOL. The instructions consist of three structure types: sequential, decision, and iteration. The instructions are placed in modules or subroutines with the data declarations kept separately from the procedure code.

Visual programming, such as Visual Basic, consists of visual objects that contain procedural code. An object can be loosely described as a collection of memory locations together with all the operations that can change the values of these memory locations (Louden, 1993). Data declarations, data definitions and program instructions are all under one identifier, which is known as an object. The language characteristic of Visual programming is the manipulation of visual objects on a computer screen.

Visual Basic evolved from and is an enhancement of regular procedural BASIC (Pietromonaco, 2002; Shelly & Cashman & Quasney, 2003). Visual Basic has the code for the procedural structures of sequence, iteration, and selection with the added features of visual object-oriented components. The visual components, such as a button, are known as objects. They have properties and event procedures (Nelson, 1993). Visual Basic "public" and "private" procedures are like OOP public and private methods. Visual objects encapsulate properties and event-procedures (Schneider, 1999). Such characteristics are lacking in procedural languages, therefore making visual programming different from procedural programming..

The literature supports the idea that Visual Basic is a type of Visual Programming language, different from procedural. (Buchner, 1999; Grehan, 1996a, 1996b; Llewellyn & Stanton & Roberts, 2002; Oz, 2002; Potter, 2003; Spain, 1996; Stair and Reynolds, 2001). One academic text book describes Visual Basic as an OOP language, rather than a third generation procedural language like BASIC, C, COBOL, Pascal (O'Brien, 2004). Visual Basic supports a syntax that looks a little object-oriented (Holtzman, 1996; Bradley & Millspaugh, 2003). A report describes the extent to which object-oriented (OO) programming can be performed in Visual Basic (Kai & McKim, 1998).

There is a distinction between procedural languages and Visual Basic. "In procedure-oriented languages, the emphasis of a program is on how to accomplish a task. The programmer must instruct the computer every step of the way. The programmer determines and controls the order in which the computer should process the instructions. Object-oriented/event-driven programming languages emphasis is on the objects included in the user interface (such as buttons) and the events (such as clicking) that occur when those objects are used. Visual Basic is an object-oriented/event-driven programming language." (Zak, 1999).

"To stress that Visual Basic is fundamentally different from traditional programming languages, Microsoft uses the term project, rather than program, to refer to the combination of programming instructions and user interface that makes a Visual Basic application possible" (Schneider, 1999). With its object-oriented methods and procedures, visual basic and other "visual" programs require a different mindset from the common in-line programming languages (Shirer, 2000).

2. LITERATURE REVIEW

Procedural programming, math skills and several cognitive abilities such as general reasoning and analytic processing have a positive correlation (Fletcher, 1984). Three studies have shown relationships between success in procedural programming, mathematics proficiency, and Piaget's cognitive development (Cafolla, 1987; Azzedine, 1987; and Werth, 1985). These relationships may be due to the usage of the same area of the brain. Studies have shown both procedural programming performance and math ability correlating to the left hemisphere of the brain (Losh, 1984; Ott, 1989; Rotejnberg and Arshavsky, 1997).

2.1 Piaget's Cognitive Development Theory

Piaget's cognitive theory consists of three development levels (Piaget, 1972; Epstein, 1990): pre-operational, concrete, and formal operations. The first cognitive level, pre-operational, is a very low level of thinking. Such a person can use symbols from visual and body sensation to represent objects but has problems with reversing actions mentally (Biehler and Snowman, 1986, p. 62). For example, that person fails to failure to recognize that the amount of water remains the same when poured from a tall thin glass to a short wide glass. At the next level, concrete, a person can understand conservation of matter and classification/generalization; i.e. conclude that all dogs are animals and not all animals are dogs. However, such a person is unable to comprehend mathematical ratios (Barker and Unger, 1983). The final and highest cognitive development level defined by Piaget is formal operation. The ability to deal with abstractions, form hypotheses, solve problems systematically, and engage in mental manipulations characterizes this cognitive level (Biehler and Snowman, 1986, p. 63). Biconditional reasoning, such as "if and only if" logic, is a precondition to formal operational reasoning (Lawson, 1983). Procedural programming logic uses biconditional reasoning.

Piaget's theory indicates that formal operational thinking abilities develop around age 12 (Chiapetta, 1976). It is at this age that some students begin to move from concrete thinking to logic/abstract thinking. Several studies have shown that formal operations, such as abstractions and logical thinking, develop at different ages or not at all in people (Griffiths, 1973; Schwebel, 1975; Pallrand, 1979; Bastian et al., 1973; Epstein, 1980). Many high school and college students fail to attain full formal operational thinking (Griffiths, 1973; Renner and Lawson, 1973; Renner et al, 1978; Schwebel, 1972, 1975). This also applies to adults. Research has shown that a majority of adults fail at many formal operational tasks (Petrushka, 1984; Sund, 1976).

2.2 Cognitive characteristics of Computer Programming

Research suggests that procedural programming deals with high cognitive abilities such as problem solving and Piaget's cognitive formal operations (Dalbey and Linn, 1985; Hudak and Anderson, 1990). Many other studies have shown that formal operational reasoning ability is necessary for success in procedural computer programming/logic (Cafolla, 1987; Fletcher, 1984; Little, 1984; Ricardo 1983; Azzedine, 1987; Barker and Unger, 1983; Barker, 1985).

Since procedural programming skills are related to logical reasoning (Cafolla, 1987; Flok, 1973; Foreman, 1988, 1990), low cognitive development thinkers are unable to do programming in light of Piaget's theory of cognitive development. This is consistent with Little's (1984) study. That study showed students who tested high in formal operations, scoring higher on programming and logical thinking measures than students who were concrete thinkers (a Piaget's lower level of cognition).

Cafolla (1987) found, "... some people of college age have difficulty learning procedural programming. This suggests that the cognitive skills needed to learn procedural programming develop later or perhaps never, in some." There are those who lack or have limited cognitive skills to learn procedural programming (Becker, 1982).

Cognitive development is a factor in determining one's ability to learn procedural programming (Folk, 1973). Those who reach Piaget's formal operational stage, have the mental tools needed to understand programming. They have an abstract learning style that helps them learn programming (Hudak and Anderson, 1990).

Two resent studies have shown that object-oriented programming also requires formal operational reasoning ability (White, 2001; 2002). Is this also true for visual programming? Does visual programming success require formal operational cognitive development just like procedural and object-oriented programming do?

2.3 Math as an Indicator

The learning of complex, abstract concepts found in mathematics appears to require Piaget's formal operation cognitive level (Pallrand, 1979; Parrino, 1981; Niaz, 1989; Nasser, 1993; Wolfe, 1999). Therefore, math is a good indicator of having the required cognitive development level to learn procedural programming. However, math grades from high school or college courses may be less accurate due to different instructors, different books, different tests, and different grading standards (White, 2003). The grades may not be comparable.

3. METHOD

To do research that will verify a math course to be beneficial, is difficult. It is infeasible to "randomly" assign students to different semester programming and math courses. Students have a set curriculum of courses to follow. The best way to research a math course prerequisite that indicates the required cognitive skills, is to correlate math course grades with programming course grades (White, 2003).

This study was done independent of instructors or math course locations. The intervening variables of different instructors and locations were not controlled or held constant. The math courses were taken with different instructors at various state universities and community colleges.

3.1 Data

A request to a state university Registrar's Office was made for all students who took the first Computer Information Systems (CIS1) programming course for the past 3 years. Each record contained the CIS1 grade, three Freshmen Mathematics, and ACT/SAT math scores. The study did not consider Math equivalent courses taken. For example: it is possible that students who took College Calculus did not have this math grade considered in the evaluation. The sample size was 837 records.

Grades were given the values of 4 for an "A," 3 for a "B," 2 for a "C," 1 for a "D," and 0 for an "F." Grades of "W" were treated as missing when performing correlations and step-wise regression. "W" grades were considered when evaluating grade distributions with math courses serving as a filter. If a course was repeated, the first grade was used. Using the second grade would have induced inflated grades due to familiarity of course content. However, many grades were either missing or were "Withdraw". These grades were dealt with as exclude cases pair wise in the statistical analysis.

The sequence in taking the courses was ignored. Research has shown that Math and Programming courses do not improve/change cognitive development nor ability (Kurland et al., 1986; Flores, 1985; Platt, 1990; Shaw, 1984; Ignatuk, 1986; Mains, 1997; Kim, 1995; Priebe, 1997)

3.2 Variables

The dependent variable was a Sophomore level CIS Introductory Programming course (CIS1) using Visual Basic.

The five independent variables were:

(1) a Freshmen College Algebra course (Math1). This course covered linear equations, inequalities, word problems, functions, and logarithms.
(2) a Freshmen Mathematics for Business and Economics I course (Math2). This course covered college algebra and finite mathematics. College Algebra (Math1) was a prerequisite.
(3) a Freshmen Mathematics for Business and Economics II course (Math3). This course covered college finite mathematics and elementary differential calculus. College Algebra (Math1) was a prerequisite.
(4) the SAT math score
(5) the ACT math score

3. 3 Statistics

Descriptive statistics and a correlation matrix of all variables were obtained. The independent variables were the three math courses and ACT/SAT scores. The math course grades (independent variable) of a grade of "2" (a grade of "C") or better, was used as a filter. This showed the changes in grade distribution of the visual programming course (dependent variable) when the math course was set as the criteria. Class GPA's were calculated before and after filtering.

4. RESULTS

Over the last three years the percentage of D's and F's assigned in the CIS1 course, was 23%. When "W's" were considered, the percentage of D's, F's, and W's jumped to 34%. Table 1 shows that as the level of the prerequisite math course increased, so did the CIS1 class GPA, while poor grades of D's, F's, and W's decreased. However, the number of students decreased since many had not yet taken the math courses. That is, out of the entire population of students enrolled in the programming course, 34% of them made a D, F or W in the programming class. Out of the students enrolled in the programming course who had already passed the Math3 course with grade C or better, only 25.3% of them made a D, F or W in the programming class. As shown in the literature for procedural programming (Taylor and Moundifled, 1991; Ott, 1988; Renk, 1987), this visual programming course had significant correlations with math course grades and ACT/SAT scores as shown in Table 2. The highest correlations were with Math2 and Math3. With only one exception (SAT_Math and Math1), all variables correlated at the .05 confidence level.

It is interesting to note that the ACT/SAT scores also correlated at the confidence level of .05, yet the correlations were small when compared to the math courses. This may be due to a difference between students' ability, as indicated by the ACT/SAT scores, and a willingness to perform, as indicated by math course grades.

5. DISCUSSION

The statistical relationships shown in the data analysis are sufficient to allow us to draw several conclusions and inferences. First, it is reasonable to conclude that the Freshman Math course, while insufficient to improve a student's analytical and logical thinking skills, it is quite efficient and effective in assessing those skills. Second, from this and prior research it is clear that analytical and logical thinking skills are necessary to perform successfully in Math as well as in procedural, object-oriented, and visual programming courses. Third, it is clear that successes in the Freshman Math courses are a fairly good predictor of potential success in a Sophomore visual programming class.

The significance and practical usefulness of this research lies in the fact that we can predict the potential for success in a visual programming class from the student's performance in the Freshman Math class. This can be an invaluable tool in advising students as to whether they should pursue a visual programming class or not. We will be serving our students tremendously if we can advise them whether it will be fruitful for them to pursue computer programming courses or not, since their potential for success in computer programming can be predicted from their performance in their Freshman Math class. Most universities teach computer programming during the Sophomore year after the student has had a Freshman Math course. The performance in the Freshman Math course can be used as an advising tool. Students who failed or performed poorly in the Math class can be advised that based on the results of this study, they would be unlikely to perform well in a visual programming course. Therefore a fixed Math prerequisite to the programming class, will do the students a service by not allowing them to enroll in a Programming course that statistically they would be expected to perform very poorly in. A coursework that does not involve computer programming can be arranged for a student who performs badly in the Freshman Math class. The effect on the students will be that we can advise them better as to what courses they are expected to perform well in, therefore guiding them towards a degree plan that they are cognitively capable of. The effect on faculty and university administration will be that students who are enrolled in computer programming classes will be expected to perform well and the success rates in those classes will increase. This ability to predict the student's success will be a win-win situation for both the student and the academic institution.

Table 1. CIS1 Grades

	Math Prerequisite of grade "C" or better			
	No Prereq	Math1	Math2	Math3
Class GPA	2.29	2.25	2.38	2.50
Grades % D's & F's (no W's considered)	23%	23%	21%	16%
% of D's, F's, & W's	34%	33%	28.6%	25.3%
Total N of students	837	283	321	389

Table 2. Correlations

		CIS1	MATH1	MATH2	MATH3	ACT_MAT	SAT_MAT
CIS1	Pearson	1.000	.199**	.370**	.333**	.142*	.135*
	Sig. (2-tailed)	.	.000	.000	.000	.045	.011
	N	722	360	348	469	199	350
MATH1	Pearson	.199**	1.000	.318**	.209**	.317**	.133
	Sig. (2-tailed)	.000	.	.000	.000	.001	.059
	N	360	420	146	283	102	203
MATH2	Pearson	.370**	.318**	1.000	.435**	.299**	.340**
	Sig. (2-tailed)	.000	.000	.	.000	.001	.000
	N	348	146	397	287	125	217
MATH3	Pearson	.333**	.209**	.435**	1.000	.304**	.264**
	Sig. (2-tailed)	.000	.000	.000	.	.000	.000
	N	469	283	287	534	161	279
ACT_MAT	Pearson	.142*	.317**	.299**	.304**	1.000	.730**
	Sig. (2-tailed)	.045	.001	.001	.000	.	.000
	N	199	102	125	161	220	173
SAT_MAT	Pearson	.135*	.133	.340**	.264**	.730**	1.000
	Sig. (2-tailed)	.011	.059	.000	.000	.000	.
	N	350	203	217	279	173	399

**. Correlation is significant at the 0.01 level (2-tailed).

*. Correlation is significant at the 0.05 level (2-tailed).

6. REFERENCES

Alspaugh, C. A. (1970). "A Study of the Relationships between Student Characteristics and Proficiency in Symbolic and Algebraic Computer Programming." Dissertation Abstracts International, 31(4627B).

Azzedine, A. (1987). "The relationship of cognitive development, cognitive style and experience to performance on selected computer programming tasks: An exploration." Dissertation Abstracts, *B48*(6), 1799.

Barker, R. J. and Unger, E. A. (1983). "A Predictor for Success in an Introductory Programming Class based upon Abstract Reasoning Development." Proceedings of the 14th SIGCSE Technical Symposium on Computer Science Education of the ACM, Orlando, Florida.

Barker, P. M. (1985). "Cognitive Correlates of Performance in Computer Programming by Children and Adolescents." Dissertation Abstracts, 46(7), 1872.

Bastian, S., Frees, J., Gruber, L., Johnson, J., Landes, B., Morton, L., Rozgony, S., and Stewart, J. (1973). "Are I.S.U. Freshman Students Operating at a Formal level of thought Processes?" Contemporary Education, 44, 358-362.

Becker, H. J. (1982). "Microcomputers in the Classroom -- Dreams or realities?" ERIC (ED217872).

Biehler, R. F. and Snowman, J. (1986). Psychology Applied to Teaching. Houghton Mifflin Company, Boston.

Bradley, J. and Millspaugh, A. (2003). Programming in Visual Basic.Net. McGraw-Hill, Boston, Mass.

Buchner, M. (1999). "Visual tools: Pros and cons." Midrange Systems, 12(6), 18.

Cafolla, R. (1987). "The Relationship of Piagetian formal Operations and other cognitive factors to computer programming ability (Development)." Dissertations Abstracts, A47(7), 2506.

Chiapetta, E. (1976). "A review of Piagetian studies relevant to science instruction at the secondary and college level." Science Education, *60*, 253-261.

Dalbey, J., and Linn, M. C. (1985). "The Demands and Requirements of Computer Programming: A Literature Review." Journal of Educational Computing Research, *1*(3), 253-74.

Epstein, H (1980). "Some biological bases of cognitive development." Bulletin of the Orton Society, 30, 46-52.

Epstein, H. (1990). "Stages in human mental growth." Journal of Educational Psychology, 82, 876-80

Fletcher, S. H. (1984). "Cognitive Abilities and Computer Programming." EDRS (ED259700).

Flores, A (1985). "Effects of Computer Programming on the Learning of Calculus Concepts." Dissertation Abstracts International, 46, 12A, p3640.

Folk, M. J. (1973). "Influences of Developmental Level on a Child's Ability to Learn Concepts of Computer

Programming." Dissertation Abstracts International, 34(3), 1125a.

Foreman, K. H. (1988). "Cognitive Style, Cognitive Ability, and the Acquisition of Inital Computer Programming Competence." EDRS (ED295638).

Foreman, K. H. (1990)."Cognitive Characteristics and Initial Acquisition of Computer Programming Competence." School of Education Review, 2, 55-61.

Grehan, R. (1996a). "A Complete Trilogy of Visual Basic tools: Code Complete from MicroHelp." Byte, 21, 32.

Grehan, R. (1996b). "Visual programming for Science: Visual Science from acroScience." Byte, 21, 208.

Griffiths, D. H. (1973). "The Study of the Cognitive Development of Science Students in Introductory Level Courses." ERIC (ED096108).

Holtzman, J. (1996). "Delphi and Visual Basic." Electronics Now, 67(10), 66-68.

Hudak, M. A. and Anderson, D. E. (1990). "Formal Operations and Learning Style Predict Success in Statistics and Computer Science Courses." Teaching of Psychology. 17(4) 231-234.

Ignatuk, N. (1986). "An Analysis of the Effects of Computer Programming on Analytical and Mathematical skills of high school students." Dissertation Abstracts, A47(3), 854.

Kai, J. and McKim, J. (1998). "Object-oriented capabilities of Visual Basic." Journal of Object-Oriented Programming 11(6), 46-57.

Kim, Y. (1995). "The Reasoning Ability and Achievement of College Level Students enrolled in a Logic Class in Computer Science." Unpublished Dissertation, University of Texas, Austin.

Kurland, D. M.; Pea, R. D.; Clement, C. A.; Mawby, R. (1986). "A study of the development of programming ability and thinking skills in high school students." Journal of Educational Computing Research, 2(4), 429-458.

Little, L. F. (1984). "The Influence of Structured Programming, Gender, Cognitive Development and Engagement on the Computer Programming Achievement and Logical Thinking Skills of Secondary Students." Dissertation Abstracts, A45(6), 1708.

Lawson, A. E. (1983). "The Acquisition of Formal Operational Schemata during Adolescence: The Role of the Biconditional." Journal of Research in Science Teaching, 20(4), 347-56.

Llewellyn, E. and Stanton, M. and Roberts, G. (2002). "Nine-step approach to designing successful visual programming applications." Computing & Control Engineering Journal, 13(2), 82-86.

Losh, C. L. (1984). "The relationship of student hemisphericity to performance in computer programming courses." Dissertation Abstracts A44(7), 2127.

Louden, K. C. (1993). Programming Languages, Principles and Practice. PWS Publishing Company, Boston.

Mains, M. G. (1997). "The Effects of Learning a Programming Language on Logical Thinking Skills."Unpublished Thesis, University of Nevada, Las Vegas, Nevada.

Nasser, R. and Carifio, J. (1993). "The Effects of Cognitive Style and Piagetian Logical Reasoning on solving Propositional Relation Algebra Word Problems." ERIC (ED364430).

Nelson, R. (1993). Running Visual Basic for Windows. Microsoft Press, Redmond, WA.

Niaz, M. (1989). "Translation of Algebraic Equations and its Relation to Formal Operational Reasoning". Journal of Research in Science Teaching. 26(9) 785-93

O'Brian, J. (2004). Management Information Systems: Managing Information Technology in the Business Enterprise, *6th Ed*. McGraw Hill, Boston.

Ott, C. F. P. (1989). "Predicting achievement in computer science through selected academic, cognitive and demographic variables." *Dissertation Abstracts, A49*(10), 2988.

Oz, E. (2002). Management Information Systems, 3rd Ed. Course Technology, Boston, MA.

Pallrand, G.J. (1979). "The transition to formal thought." Journal of Research in Science Teaching, *16, 445-451.*

Pietromonaco, P. (2002). "The Versatile Visual Basic." Poptronics, 3(7), 16-18.

Platt, D. (1990). "The Effect of a Second-Semester Computer Programming Course on Mathematical Problem-Solving Performance and the Relationship between selected Cognitive Factors and Mathematical Problem-Solving Performance." Dissertation Abstracts, *51*(10A), 3354.

Parrino, L. W. (1981). "The use of Cognitive Development Tasks as Predictors of Success in Developmental Mathematics Courses." Dissertation Abstracts Internatinoal 42, no. 04A, p. 1522.

Petrushka, D. (1984). "A Study of the effect of content on the ability to do syllogistic reasoning: An investigation of transferability and the effect of practice." Unpublished doctoral dissertation, Rutgers University, NJ.

Piaget, J. (1972). "Intellectual evolution from adolescence to adult." Human Development, *15*, 1-12.

Potter, T. (2003). Introduction to Information Technology, 2nd Ed. John Wiley & Sons, Inc., Hoboken, NJ.

Priebe, Roger L. (1997). "The Effects of Cooperative Learning on Content Comprehension and Logical Reasoning in a Second-Semester University Computer Science Course." Unpublished Doctoral Dissertation, University of Texas, Austin, Texas.

Ralston, A. (1984). "The first course in Computer Science needs a mathematics co requisite." Communications of the ACM, *27*(10), 1002-1005.

Renk, S. C. (1987). "Factors Affecting Academic Success in Introductory Computer Programming." Dissertation Abstracts International, A48(3), 579.

Renner, J. W., and Lawson, A. E. (1973). "Promoting Intellectual Development through Science Teaching." Physics Teacher, 11(5), 273-276.

Renner, J., Grand, R., and Sutherland, J. (1978). "Content and concrete thought." Science Education, *62*, 215-221.

Ricardo, C. M. (1983). "Identifying student entering characteristics desirable for a first course in computer programming." Dissertation Abstracts, A44(1), 96.

Rotenberg, V. S., and Arshavsky, V. V. (1997). "Right and left brain hemispheres activation in the representatives of two different cultures." Homeostasis in Health and Disease, *38*(2), 49-57.

Saiedian. (1992). "Math of Computing." Computer Science Education, *3*(3), 203-221.

Schneider, D. (1999). An Introduction to Programming using Visual Basic 6.0, 4th Ed. Prentice Hall, Upper Saddle River, NJ.

Schwebel, M. (1972). "Logical Thinking in College Freshman: Final Report." ERIC (ED 110896).

Schwebel, M. (1975). "Formal operations in first year college students." Journal of Psychology, *91*, 133-141.

Shaw, D. G. (1984). "The Effects of Learning to Program a Computer in Basic or Logo on the Problem Solving Abilities of Fifth Grade Students." Dissertation Abstracts, A45(7), p1985.

Shelly, G. and Cashman, T. and Quasney, J. (2003). Microsoft Visual Basic.NET Complete Concepts and Techniques. Thomson Course Technology, Boston, Mass.

Shirer, D. (2000). "Basic: The Little Language that wouldn't Die." Computing in Science & Engineering, 2(2), 6-10.

Spain, W. (1996). "Visual Basic." Computerworld, 30(44), 102.

Stair, R. and Ryunolds, G. (2001). Fundamentals of Information Systems. Course Technology: Thomson Learning Publishers, Boston, MA.

Sund, R. B. (1976). Piaget for educators: a multimedia program (In Brooks, R (1978) The relationship between Piagetian Cognitive development and cerebral cognitive asymmetry. ERIC (ED160224, Trans.). Columbus: Charles E. Merrill.

Taylor, H. G. and Monnfield, L. (1991). "An Analysis of Success Factors in College Computer Science: High School Methodology is a Key Element." Journal of Research on Computing Education, 24(2), 240-245.

Werth, L. (1985). "Predicting Student Performance in a Beginning Computer Science Class (Piaget, Personality, Cognitive Style)." Dissertation Abstracts International 46, no. 09A, p 2489.

White, G. L. (2001). "Cognitive Characteristics for Learning Java, an Object Oriented Programming Language." Unpublished Dissertation, University of Texas at Austin, Texas.

White, G. L. (2002). "Cognitive Characteristics for Learning C++." Journal of Computer Information Systems, 42(3), 51-55.

White, G. L. (2003). "Standardized Mathematics Scores as a Prerequisite for a First Programming Course." Journal of Mathematics and Computer Education, 37(1), 96-104.

Wolfe, F. E. (1999). "Levels of Piagetian Development among adult Mathematics Students." Dissertation Abstracts International 60, no. 09A, p. 3239.

Zak, D. (1999). Programming with Microsoft Visual Basic 6.0. Course Technology, Boston, MA.

AUTHOR BIOGRAPHIES

Garry L. White is a faculty member in the Computer Information Systems department at Southwest Texas State University (SWT) in San Marcos Texas. He holds a MS in Computer Sciences from Texas A & M University – Corpus Christi and a PhD in Science Education, emphasis in Information Systems, from The University of Texas at Austin. Professional Certifications from the Institute of Certified Computer Professionals (ICCP) include C.D.P, C.C.P., and C.S.P. He has been on the SWT faculty since 1997. His teaching interests are in the areas of Computer Programming, Data Communications, Systems Analysis, and Computer Networks. His research interests and work are in the areas of Computer Education and the Internet. He has published papers and abstracts in journals such as the Journal of Computer Information Systems. Proceeding publications have been with the Decision Sciences Institute and the Information Systems Educational Conference.

Marcos P. Sivitanides is a tenured Associate Professor of Computer Information Systems at Southwest Texas State University (SWT) in San Marcos Texas. He holds a BA (Honors) in Computer Sciences, an MBA and a PhD in Management Information Systems, all from The University of Texas at Austin. He has been on the SWT faculty since 1989. His teaching interests are in the areas of Computer Programming, Systems Analysis and Design, Database Design and Management, and Computer Networks. His research interests and work are in the areas of Decision Theory, Computer Education, and Curriculum Development and Design. He has published papers and abstracts in journals such as Decision Sciences and the Journal of Information Systems Education and conferences such as the Decision

Sciences Institute, and the Information Systems Educational Conference.

STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.

ISSN 1055-3096