

# **A COMPARISON OF PROCEDURAL PROGRAMMING ACHIEVEMENT AND OBJECT-ORIENTED PROGRAMMING ACHIEVEMENT IN RELATION TO COGNITIVE SKILLS.**

by  
Terry Dwyer  
BAppSc, BEd, GDipEd.

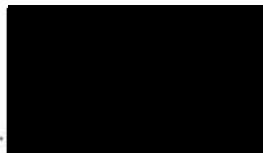
A thesis submitted in partial fulfilment  
of the requirements of the degree of  
Master of Education (Honours) of  
the University of New England

June, 1997

## Certificate of originality

I certify that the substance of this thesis has not already been submitted for any degree and is not being currently submitted for any other degrees.

I certify that to the best of my knowledge any help received in preparing this thesis, and all sources used, have been acknowledged in this thesis.

A solid black rectangular box used to redact the signature of the author.

\*\*\*  
Signature

## **Acknowledgments**

The completion of this effort would not have been possible without the help of a select number of people.

The Year 12 Information Processing and Technology students of Stanthorpe High School through 1994 and 1995 are thanked for being willing participants in this study. The Principals of the school, in 1994 and 1995, are also thanked not only for giving their permission to carry out this study but also for their strong support and interest in the study.

The guidance in design and statistical analysis provided by Dr Ken Vine contributed to the work.

I want to thank my wife, Julie, and son Mark who endured the long hours spent on this work. The support from Julie, not only morally, but also in correcting punctuation, sentence structure and grammatical errors was of immense and indispensable help.

To Dr Maurice Anker who inspired, encouraged, and challenged me, I owe a special debt of gratitude. I am especially indebted to Dr Anker and Dr Vine for their willingness to carefully examine the numerous chapter drafts and redrafts.

Finally I offer my thanks to the many people who inquired about the study and who quietly encouraged me to persevere with the study.

## Abstract

The recent advent of programming languages incorporating both procedural programming and object-oriented capabilities at a reasonable price for schools raises the question of whether an exposure of Year twelve information technology students to object-oriented programming is an appropriate pedagogical objective. The potential of object-oriented programming to inherit user interfaces formed the contextual justification for also considering the possibility that the addition of sophisticated user interfaces would enthuse students and provide more stimulating and exciting developmental work.

An information processing model based on cognitive skills and motivational factors provided the framework within which student achievement in traditional procedural programming was compared with student achievement in object-oriented programming. Forty-eight Year twelve students, randomly assigned to procedural programming and object-oriented programming classes, were administered the Learning Style Profile. Achievement was measured in three areas of programming competence: knowledge of syntax, program modification, and program composition. Attitude towards programming was measured by the use of a liking of programming subscale, a programming difficulty subscale, and a programming usefulness subscale.

The findings indicate that while there is no significant difference in student achievement between each instructional treatment, the cognitive demands of each programming environment do differ. Object-oriented programming achievement in the areas of program modification and program composition is significantly related to cognitive skill factors of simultaneous processing, persistence, and memory. The object-oriented feature of encapsulation will require instructional strategies which develop simultaneous processing skills. There is evidence to suggest that student skill in sensing an overall pattern from the relationships among components is amenable to change.

Despite the additional cognitive demands of object-oriented programming, it appears that students have no inherent difficulties in learning object-oriented programming. The study suggests that students would not feel that object-oriented programming is more difficult than procedural programming.

The majority of Queensland schools support the algorithms and programming topic, within the Year 11 and 12 Information Processing and Technology subject, with procedural programming (Clarke 1992, 4; King, Feltham and Nucifora 1994, 21). Within this context, a sequence in which students experience procedural

programming and then later use a sophisticated user interface within an object-oriented programming environment may be an acceptable transitional curriculum evolution. This intermediary position in which students study both procedural programming and object-oriented programming has some justification in that two programming languages enhance the problem solving approach and students' view of the use of computer systems for problem solving (Lawson 1985a, 541; Lawson 1985b, 542; McGrath 1988, 467-484).

The exploration of the relationships between cognitive skills and object-oriented programming achievement does provide some guideline for the design of instructional strategies and learning experiences. It also promises to improve student success because learning problems are more frequently related to the type and level of the cognitive processes required to learn the material rather than to the difficulty of the subject matter (Letteri 1988, 22).

## Contents

	Acknowledgments	i
	Abstract	ii
	List of tables	vi
	List of figures	vii
<b>1</b>	<b>Introduction</b>	<b>1</b>
	Theoretical framework	6
	Learning style	6
	Motivation	10
	The research problem	11
	Significance of the study	13
<b>2</b>	<b>Review of the related literature</b>	<b>14</b>
	The research domain	14
	Object-oriented programming	14
	Learning style	16
	Motivation	26
	Attitude	28
	Programming instruction	30
<b>3</b>	<b>Methodology</b>	<b>32</b>
	Rationale	32
	Research questions	33
	The overall objective of the thesis	33
	Definition of dependent and independent variables	33
	Hypotheses	34
	Experimental design	35
	Subjects	35
	Instructional resources	35
	Instructional sequence and data collection	36
	Threats to validity	40
	Internal threats to validity	40
	External threats to validity	41
	Data gathering instruments	44
	Learning style profile	44
	Attitude to(wards) programming scale	48
	Programming achievement test	54
	Ethical considerations	55
	Assumptions and limitations	57

<b>4</b>	<b>Analysis of data</b>	58
	Introduction	58
	Independent variables	58
	Dependent variables	58
	Overview of analyses	59
	Descriptive statistics	59
	Learning style instrument	59
	Attitude to(wards) programming scale	64
	Programming achievement instrument	67
	Analytical procedures	68
	Statistical hypothesis 1	68
	Statistical hypothesis 2	69
	Statistical hypothesis 3	74
	Statistical hypothesis 4	76
<b>5</b>	<b>Summary and conclusions</b>	79
	Restatement of the problem	79
	Summary description of procedures	81
	Major findings	82
	Conclusions	86
	Recommendations for further investigation	88
	<b>References</b>	90
	<b>Appendices</b>	
	Appendix A: Specific algorithms and programming objectives	103
	Appendix B: Normal end of unit achievement test applied prior to the study	106
	Appendix C: Procedural and object-oriented 'Athletics' code with an attached interface	110
	Appendix D: Programming achievement Instrument	123
	Appendix E: Attitude to(wards) programming instrument	127



## List of Tables

1	Some popular learning style instruments	19
2	Meta-analysis of correlational studies between programming achievement and field-independence	22
3	The relevant subscales of the Learning Style Profile	34
4	The algorithms and programming unit schedule	37
5	QCS comparison of school and state IPT students	42
6	Factor loadings on the hypothesised three attitude factors.	53
7	Learning style profile: Stem-and-leaf displays, kurtosis, and skewness of responses of the forty-eight Year 12 IPT students	60
8	Learning style profile: subscale means, standard deviations and Cronbach's alphas in Year twelve IPT students and Year twelve American students	61
9	Differences in the learning style profile means of Year 12 IPT students and Year 12 American normative data	61
10	Varimax factor loadings of the eight NASSP cognitive skill variables on two factors	62
11	Two factor cognitive skill profile: Stem-and-leaf displays, kurtosis, and skewness of scores of the forty-eight Year 12 IPT students. Standardised to a mean of 50 and a standard deviation of 15	63
12	Two factor cognitive skill profile: Skewness removed by a Box-Cox transformation and standardised to a mean of 50 and standard deviation of 15	64
13	Descriptive statistics, including stem-and-leaf display, of measurements obtained using the attitude to(wards) programming instrument (forty-eight responses)	65
14	Reliability indices (Cronbach's alpha) and student numbers for the attitude to(wards) programming subscales	66
15	Descriptive statistics, including stem-and-leaf display, of measurements obtained using programming achievement instrument (forty-eight responses)	67
16	Student's t statistics comparing achievement means of each instructional treatment ( $t_{0.05,46df,two-tail}$ ).	69
17	Pearson coefficients of correlation measuring the strength of the linear relationship between each of the cognitive skills and student programming achievement within each of the programming environments	70
18	Linear regression analysis quantifying the contribution of cognitive skill factors to programming achievement	71



19	Two-way ANOVA of student programming achievement. One factor being cognitive skill (factor A, factor B) and the other factor being programming environment (procedural, object-oriented)	73
20	Two-way ANOVA of student attitude to(wards) programming. The factors programming liking, difficulty, and usefulness each at two levels (pretest, posttest) and the factor programming environment at two levels (procedural programming, object-oriented programming)	75
21	Two-way ANOCOVA student posttest attitude to(wards) programming with pretest attitude as covariate. One factor being instructional treatment (procedural, object-oriented) and the other factor being cognitive skill (factor A, factor B).	78

## List of Figures

1	The evolution of programming languages	3
2	A 'Hello World' comparison	5
3	The major variables in Bloom's theory of school learning	6
4	Information processing, general operations Charles A. Letteri, 1982	8
5	Perceptions and inferences of the expectancy X value motivational theory	11
6	The experimental design	38
7	Plots of mean programming achievement (syntax (a) and (b), program modification c) and (d), and program composition (e) and (f)) in cognitive factors A and B (low , high).	72
8	Plots of mean attitude (pretest, posttest) of programming environment in liking programming (a), programming difficulty (b), and programming usefulness (c).	74
9	Plots of mean posttest attitude, adjusted for the pretest attitude covariate, (liking programming (a) and (b), programming difficulty (c) and (d), and programming usefulness (e) and (f)) in cognitive skill factors A and B (low, high).	77